Recent progress in unconstrained nonlinear optimization without derivatives

by A.R. Conn[1], K. Scheinberg[2] and Ph.L. Toint[3]

Report 97/12 April 24th, 1997

[1] IBM T.J. Watson Research Center, P.O.Box 218, Yorktown Heights, NY 10598, USA
Email: arconn@watson.ibm.com

[2] Industrial Engineering and Operations Research Department,
Columbia University, New York, NY 10027-6699, USA
Email: katya@ieor.columbia.edu

[3] Department of Mathematics, Facultés Universitaires ND de la Paix, 61, rue de Bruxelles,
B-5000 Namur, Belgium, EU
Email: pht@math.fundp.ac.be
Current reports available by anonymous ftp from the directory
"pub/reports" on thales.math.fundp.ac.be
WWW: http://www.fundp.ac.be/sciences/math/pht/publications.html

Invited semi-plenary presentation at the ISMP97, Lausanne, August 1997.

Recent progress in unconstrained nonlinear optimization without derivatives

A. R. Conn K. Scheinberg Ph. L. Toint

August 14, 1997

Abstract

We present an introduction to a new class of derivative free methods for unconstrained optimization. We start by discussing the motivation for such methods and why they are in high demand by practitioners. We then review the past developments in this field, before introducing the features that characterize the newer algorithms. In the context of a trust region framework, we focus on techniques that ensure a suitable "geometric quality" of the considered models. We then outline the class of algorithms based on these techniques, as well as their respective merits. We finally conclude the paper with a discussion of open questions and perspectives.

1 Motivation

In this paper, we consider the problem of minimizing a nonlinear smooth objective function of several variables when the derivatives of the objective function are unavailable and when no constraints are specified on the problem's variables. More formally, we consider the problem

$$\min_{x \in \mathbf{R}^n} f(x),$$

where we assume that f is a smooth nonlinear function from $\mathbf{R}^n$ into $\mathbf{R}$, and that $\nabla f(x)$ (and, a fortiori, $\nabla^2 f(x)$) cannot be computed for any x.

Our interest and motivation for examining possible algorithmic solutions to this problem is the high demand from practitioners for such tools. In the applications presented to the authors, computing the value $f(x)$ given a vector x is typically *very expensive*, and the values of the derivatives of f at x are not available either because $f(x)$ results from some physical, chemical or econometric measurements, or, more commonly, because it is the result of a possibly very large and complex computer simulation, for which the source code is effectively unavailable or unmodifiable. The occurrence of problems of this nature appears to be surprisingly frequent in the industrial setting. In particular, the wider use of highly specialized, powerful but proprietary simulation packages developed over a substantial period of time makes the second of these situations increasingly common.

Of course, one might consider turning to automatic differentiation tools (see Griewank and Corliss [15] or Griewank [14], for instance). However, such tools are unfortunately not applicable in the two typical cases mentioned above since they require $f(x)$ to be the result of a callable program that cannot be treated as a black box.

Another possibility is to resort to finite difference approximation of the derivatives (gradients and possibly Hessian matrices). A good introduction to these techniques can be found in the book of Dennis and Schnabel [12], for instance. In general, given the cost of evaluating the objective function, evaluating its Hessian or gradient by finite difference can be much too expensive. One can use quasi-Newton Hessian approximation techniques instead. In conjunction with the use of finite differences for computing gradients, when the resources are available or the evaluations are not too unreasonably expensive, this type of method has proved to be useful and sometimes surprisingly efficient.

When users are faced with problems for which function evaluations are very expensive and it is not appropriate to determine derivatives directly, there are a few strategies that can be considered. The first, and maybe simplest, is to apply existing "direct search" optimization methods, like the well-known and widely used simplex reflection algorithm of Nelder and Mead [18] or its modern variants, or the Parallel Direct Search algorithm of Dennis and Torczon [13], [32]. This first approach has the merit of requiring little additional effort from the user, but may require substantial computing resources: the inherent smoothness of the objective function is not very well exploited, and, as a result, the number of function evaluations is sometimes very large and in the case of Nelder and Mead, for example, the progress may be very poor, a major drawback when these evaluations are expensive. An advantage is that they can be suitable for non-smooth problems. We think it is useful to think of these approaches as sampling methods.

On the other hand all the other approaches mentioned below can be considered to be modeling methods in that they try to approximate the function over a region by some smooth model and, as one would expect, less readily lend themselves to the optimization of non-smooth functions.

In the case of smooth problems, this idea seems particularly attractive in that one can replace an expensive function evaluation by a much cheaper surrogate model and, especially for very complex problems, make considerable progress in obtaining improved solutions at moderate cost. The following interesting argument in favor of such techniques is for instance proposed by Powell [21], where he considers the relatively simple problem of solving a single nonlinear equation. The idea is to compare, in this context, the secant method with the Newton-Raphson method. The first requires one function evaluation per iteration and has a convergence rate of approximately 1.618 per function value while the second requires one function and one derivative value per iteration and has quadratic convergence. Therefore, if an extra function value is used to estimate the derivative in the Newton-Raphson iteration, the mean rate of convergence per function value is only equal to $\sqrt{2}$. This simple example indicates that it may not be optimal to use function values to compute explicit derivative approximations.

We will group under the name *derivative free optimization* all methods which do not attempt to *directly* compute approximations to the unavailable derivative information. Our purpose in this paper is not to provide a detailed investigation of an algorithm, but rather to introduce some basic concepts for derivative free methods, and to outline a whole class of algorithms in which these concepts are embodied.

There are two essential ingredients for derivative free methods. The first is to pick *better* points. In the case of sampling methods the intent is that the algorithm is designed to exploit, typically implicitly, the probabilities suggested by the samples to determine the next place to sample. In the case of modeling methods the expectation is that the surrogate model will predict suitable points. Of course, in both cases there is always the possibility of failure, which must then be corrected in a systematic way.

The other essential ingredient is that one searches appropriate subspaces since otherwise one may have no chance of finding an appropriate minimum. Indeed in the case of the Nelder and Mead method, it is not unusual for the simplices to degenerate and thereby stall progress. In modeling methods, for example using multivariant interpolation, such *geometry* considerations have to be an explicit component of the algorithm.

The rest of the paper is organized as follows. After a short survey of the history of derivative free optimization techniques in Section 2, we investigate, in Section 3, the basic algorithmic features that characterize the newer methods. We then discuss at some length two possible approaches for the realization of some of the concepts needed in the proposed class of algorithms. Just as in the univariant case, the simplest approach to consider first is interpolation based upon Lagrangian polynomials, but again analogous to the univariant case a major disadvantage of this choice of basis is that the entire set of fundamental polynomials needs to be reconstructed for every new interpolation. By contrast, Newton fundamental polynomials, amongst other advantages, can be updated. Section 4 presents an approach based on the Lagrange fundamental polynomials, while Section 5 investigates the use of multivariate Newton interpolation techniques. We conclude the presentation by discussing open questions and perspectives in Section 6.

2 A brief review of derivative free optimization methods

It is difficult to firmly state where the idea of derivative free methods for minimization was first introduced, but as Wright remarks in her interesting survey [36], direct search methods appear to have first been suggested in the 1950's. The first simplex based search was that of Spendley *et al.* [29], which like the work of Torczon [32], preserved the essential geometry. This was 'improved' on the grounds of efficiency by no longer insisting on maintaining the regularity of the simplex, first by Campey and Nickols [5], and then by Nelder and Mead [18]. They introduced, along with 'reflection away from the worst simplex vertex' the concept of contraction and expansion, which was carried out relative to a single axis. The essential modification of Torczon was that her reflections, expansions and contractions involved the entire pattern (simplex) so that good geometry is necessarily preserved. In addition an iteration succeeds when it finds a point of strict improvement over the best vertex, in contrast to Nelder and Mead's much weaker insistence on only requiring improvement compared with the worst vertex. Torczon observes and exploits the fact that the method can be readily and efficiently parallelized, which in some sense compensates for the additional function evaluations required to maintain the geometry. Because of the inherent preservation of the pattern geometry, along with the mechanism for step control that does not allow contraction on successful steps, Torczon [33] is able to provide a general convergence theory that applies to the early evolutionary design method of Box [2], Hooke and Jeeves [17] as well as Dennis and Torczon [13] and Torczon [32], but not to Nelder and Mead [18]. In [19], Powell described a method for solving the nonlinear unconstrained minimization problem based on the use of conjugate directions. The main idea of this proposal is that the minimum of a positive-definite quadratic form can be found by performing at most n successive line searches along mutually conjugate directions, where n is the number of variables. The same procedure may of course be applied to non-quadratic functions, adding a new composite direction at the end of each cycle of n line searches. Of course, finite termination is no longer expected in this case. This algorithm has enjoyed a lot of interest amongst both numerical analysts and practitioners. The properties of the method were analyzed by Brent [3], Callier and Toint [4], [31] and Toint [30]. The various computer programs based on this method have been widely used by a large number of practitioners[1].

Independently, Winfield developed a different idea in his Harvard thesis during the late 1960's. The main idea, expressed in [34] and [35], is to use the available objective function values $f(x_i)$ for building a quadratic model by interpolation. This model is assumed to be valid in a neighborhood of the current iterate, which is described as a trust region (a hypersphere centered at x_i), whose radius is iteratively adjusted. The model is then minimized within the trust region, hopefully yielding a point with a low function value. As the algorithm proceeds and more objective function values become available, the set of points defining the interpolation model are updated in such a way that its always contains the points closest to the current iterate. This contribution is remarkable, not only because this idea (admittedly with some crucial modifications) forms the basis of the methods we wish to study here, but also because it appears to be a very early statement of a trust-region method, even before the seminal paper of Powell [20]. It is somewhat surprising that it went mostly unnoticed.

Both the method of Winfield and that of conjugate directions have proved to be reasonably reliable[2]. In [35], Winfield recognizes the difficulty that interpolation points must have certain geometric properties, although he does not seem to consider what happens if these properties are not obtained. The conjugate directions method suffers from the relative problem of determining near-conjugate directions when the Hessian of the function is ill-conditioned. Recognizing this difficulty for this latter case, Powell [21] suggested using orthogonal transformations of sets of conjugate directions. Pursuing this idea, Powell [22] proposed to approximate the matrix of second derivatives itself, by modifying an initial estimate to ensure that it satisfies properties which would be satisfied if the objective function were quadratic. He also suggested the use of variational criteria, such as those used to derive quasi-Newton updates, in order that good information from an approximation can be inherited by its successors at subsequent iterations.

[1] Powell made his method available in the Harwell Subroutine Library under the name of VA04, but this original routine has been replaced by VA24, also written by Powell. Brent [3] gave an ALGOL W code named PRAXIS for a variant of the method. This latter code was subsequently translated to Fortran by R. Taylor, S. Pinski and J. Chandler. The Fortran version of PRAXIS is distributed in the public domain by J. Chandler, Computer Science Department, Oklahoma State University, Stillwater, Oklahoma 70078, USA (jpc@a.cs.okstate.edu). There is also an interface with the CUTE testing environment of Bongartz *et al.* [1].

[2] We base our statement for Winfield's method on the numerical experience reported in [35].

Twenty-five years later, Powell [24] proposed a method for constrained optimization, whose idea is close to that of Winfield. In his proposal, the objective function and constraints are approximated by linear multivariate interpolation[3]. Exploring the idea further, Powell [25] then described an algorithm for unconstrained optimization using a multivariate quadratic interpolation model of the objective function in a trust-region framework, an approach extremely similar to that of Winfield although seemingly independent. The crucial difference between Powell's and Winfield's proposals is that in the former the set of interpolation points is updated in a way that *preserves its geometric properties*, in the sense that the differences between points of this set are guaranteed to remain sufficiently linearly independent, therefore avoiding the difficulties associated with earlier proposals. A variant of this quadratic interpolation scheme was then discussed in Conn and Toint [10], where encouraging numerical results were presented. Powell [26] subsequently revisited this approach and showed similar computational results. The first convergence theorems for methods of this type were finally presented by Conn, Scheinberg and Toint [9], together with a description of alternative techniques to enforce the desired geometric properties of the set of interpolation points.

It is the purpose of the next section to investigate these crucial "geometry preserving" techniques, in the frameworks suggested by Powell and by Conn, Scheinberg and Toint.

3 A class of derivative free algorithms

The class of algorithms discussed in this paper belongs to the class of *trust-region* methods. Such algorithms are iterative and build, around the current iterate, a model of the true objective function which is cheaper to evaluate and easier to minimize than the objective function itself. This model is assumed to represent the objective function well in a so-called trust region, typically a ball centered at the current iterate, x_k, of the form

$$\mathcal{B}_k = \{x \in \mathbf{R}^n \mid \|x - x_k\| \leq \Delta_k\}$$

The radius of this ball, Δ_k, is called the trust region radius and indicates how far the model is thought to well represent the objective function. A new trial point is then computed, which minimizes or sufficiently reduces the model *within the trust region* and the true objective function is evaluated at this point. If the achieved objective function reduction is sufficient compared with the reduction predicted by the model, the trial point is accepted as the new iterate and the trust region is centered at the new point and possibly enlarged. On the other hand, if the achieved reduction is poor compared with the predicted one, the current iterate is typically unchanged[4] and the trust region is reduced. This process is then repeated until convergence (hopefully) occurs.

[3] The associated Fortran code, named COBYLA, is distributed to interested parties by M.J.D. Powell, DAMTP, University of Cambridge, Silver Street, Cambridge CB3 9EW, UK. An interface with the CUTE environment of Bongartz *et al.* [1] is also provided for COBYLA.

[4] This, of course, does not prevent the algorithm recording the best point found so far, and returning to this point at the end of the calculation, if no better point was found.

The first main ingredient of a trust region algorithm is thus the choice of an adequate objective function model. We will here follow a well established tradition in choosing, at iteration k, a quadratic model of the form

$$m_k(x_k + s) = f(x_k) + \langle g_k, s\rangle + \tfrac{1}{2}\langle s, H_k s\rangle, \tag{3.1}$$

where $\langle x, y\rangle$ denotes the usual Euclidean inner product between x and y, where g_k is a vector of $\mathbf{R}^n$ and where H_k is a square symmetric matrix of dimension n. However, we will depart from many trust-region algorithms in that g_k and H_k will not be determined by the (possibly approximate) first and second derivatives of $f(\cdot)$, but rather by imposing that the model (3.1) interpolates function values at past points, that is we will impose that

$$m_k(y) = f(y) \tag{3.2}$$

for each vector y in a set Y such that $f(y)$ is known for all $y \in Y$. Note that the cardinality of Y must be equal to

$$p = \tfrac{1}{2}(n+1)(n+2)$$

to ensure that the quadratic model is entirely determined by the equations (3.2). However, if $n > 1$, this last condition is not sufficient to guarantee the existence of an interpolant. For instance, six points on a line do not determine a two dimensional quadratic. Similarly, six interpolation points on a circle in the plane do not either, because any quadratic which is a multiple of the equation of the circle can be added to the interpolant without affecting (3.2). One therefore sees that some *geometric conditions* on Y must be added to the conditions (3.2) to ensure existence and uniqueness of the quadratic interpolant (see [11] or [28], for more detail). The purpose of the next two sections will be to relate these conditions to the more general question of exploring along suitable directions during the optimization. In the case of our second example, we must have that the interpolation points do not lie on any quadratic surface in $\mathbf{R}^n$ or that the model chosen includes terms of degree higher than two. More formally, we need the condition referred to as poisedness, which relates directly to the interpolation points and the approximation space. If we choose a basis $\{\phi_i(\cdot)\}_{i=1}^p$ of the linear space of n-dimensional quadratics, we follow [28] and say that $Y = \{y_1, \ldots, y_p\}$ is *poised* when the "interpolation determinant"

$$\delta(Y) = \det \begin{pmatrix} \phi_1(y_1) & \cdots & \phi_1(y_p) \\ \vdots & & \vdots \\ \phi_p(y_1) & \cdots & \phi_p(y_p) \end{pmatrix}$$

is non-zero. We note that the magnitude of the determinant $\delta(Y)$ is a measure of the linear independence of its columns. Of course, the quality of the model (3.1) as an approximation of the objective function around x_k will be dependent on the geometry of the considered interpolation points and, from a practical point of view, it is important that we are able to measure this quality. We will examine possible choices in the next two sections. In Section 4 we consider an implementation that uses a scaled version of $\delta(Y)$. In Section 5 we study a more convenient measure. For now, we only need to assume that we know what we mean when we say that the geometry of Y is *adequate* (in a given neighbourhood containing all the points of Y) or when we say that we *improve* this geometry with respect to our measure.

A second ingredient in trust-region algorithms is that one accepts a new point x_k^+ produced by the algorithm as soon as $f(x_k^+)$ is sufficiently smaller than $f(x_k)$, the current objective value. This is standard for trust-region methods, but the situation is more complex here because this means that we have to *include* x_k^+ in the set of interpolation points Y. In general[5], this means we need to remove another point $y \in Y$. Ideally, this point should be chosen to make the geometry of Y as adequate as possible. There are various ways to attempt to achieve this goal. We will again discuss some possibilities in the next section. It is important to note that, since x_k^+ is given irrespective of the geometry of the points already in Y, there is no guarantee that the quality of the geometry of Y will remain acceptable, or even that Y will remain poised, which opens the possibility that the quality of the geometry of Y might deteriorate as new iterates are accepted by the algorithm. This can happen, for instance, if successive iterates lie at the bottom of a steep valley whose shape may be described by a quadratic curve[6]. In such situations, $|\delta(Y)|$ may become very small, in which case we may question the relevance of our model.

As in all trust-region algorithms, we are also faced with the possibility that no further progress can be made from the current iterate x_k. In algorithms using exact derivative information, Taylor's theorem ensures that this problem can only occur away from stationary points because the trust-region radius Δ_k is too large, and thus guarantees that it disappears if Δ_k is made sufficiently small. In our framework, we must also consider the possibility that the model's quality is not adequate. This inadequacy could indeed result from the phenomenon described in the previous paragraph. If the algorithm is not able to progress, we thus have to check first if the geometry of Y is adequate, and, if it is not the case, we have to improve it before having to reduce Δ_k. The desired improvement is achieved by introducing a new interpolation point y_+ in the set Y such that $\|y_+ - x_k\| \le \Delta_k$ and using our measure of improvement to evaluate the merits of replacing some past point $y_- \in Y \setminus \{x_k\}$ by y_+, possibly comparing several choices for y_-. This is the third main ingredient of our class of algorithms.

Following this introduction to the main concepts, we may now outline this class as follows. We assume that the constants

$$0 < \eta_0 \le \eta_1 < 1, \text{ and } 0 < \gamma_0 \le \gamma_1 < 1 \le \gamma_2,$$

are given.

Outline of a derivative free trust-region algorithm

Step 0: Initializations.

Let x_s and $f(x_s)$ be given. Choose an initial interpolation set Y containing x_s. Then determine $x_0 \in Y$ such that $f(x_0) = \min_{y_i \in Y} f(y_i)$. Choose an initial trust region radius $\Delta_0 > 0$. Set $k = 0$.

[5] Not necessarily so in the early stages of the algorithm, see below.

[6] The bottom of the valley in Rosenbrock's function is described by the equation $x_2 = x_1^2$.

Step 1: Build the model.

Using the interpolation set Y, build a model $m_k(x_k+s)$, possibly restricting Y to a poised subset containing x_k, such that conditions (3.2) hold for the resulting Y.

Step 2: Minimize the model within the trust region.

Compute the point x_k^+ such that

$$m_k(x_k^+) = \min_{x \in \mathcal{B}_k} m_k(x). \tag{3.3}$$

Compute $f(x_k^+)$ and the ratio

$$\rho_k \stackrel{\text{def}}{=} \frac{f(x_k) - f(x_k^+)}{m_k(x_k) - m_k(x_k^+)}.$$

Step 3: Update the interpolation set.

- If $\rho_k \geq \eta_1$, *include* x_k^+ in Y, dropping one of the existing interpolation points in necessary.
- If $\rho_k < \eta_1$ and Y is inadequate in $\mathcal{B}_k$, *improve* the geometry of Y in $\mathcal{B}_k$.

Step 4: Update the trust-region radius.

- If $\rho_k \geq \eta_1$, then set

$$\Delta_{k+1} \in [\Delta_k, \gamma_2 \Delta_k].$$

- If $\rho_k < \eta_1$ and Y was adequate in $\mathcal{B}_k$ when x_k^+ was computed, then set

$$\Delta_{k+1} \in [\gamma_0 \Delta_k, \gamma_1 \Delta_k]$$

- Otherwise, set $\Delta_{k+1} = \Delta_k$.

Step 5: Update the current iterate.

Determine $\hat{x}_k$ such that

$$f(\hat{x}_k) = \min_{\substack{y_i \in Y \\ y_i \neq x_k}} f(y_i).$$

Then, if

$$\hat{\rho}_k \stackrel{\text{def}}{=} \frac{f(x_k) - f(\hat{x}_k)}{m_k(x_k) - m_k(x_k^+)} \geq \eta_0,$$

set $x_{k+1} = \hat{x}_k$. Otherwise, set $x_{k+1} = x_k$. Increment k by one and go to Step 1.

End of algorithm

Our outline is admittedly broad and simplistic. However, our current description is sufficient to provide the framework of the discussion of the next section. Practical algorithms involve a number of additional features that enhance efficiency. In particular, we have not mentioned a stopping test, a possible choice for which is to stop the calculation if the trust-region radius falls below a certain threshold and the geometry is adequate.

We also note that any further improvement in the model, compared with whatever the algorithm explicitly includes, is also possible. For instance, one might wish to include x_k^+ in Y, even if $\rho_k < \eta_1$, (perhaps provided it does not deteriorate the quality of the model). Indeed, we have computed $f(x_k^+)$ and any such evaluation should be exploited if at all possible. One could also decide to perform a geometry improvement step if ρ_k is very small, since this indicates a bad fit of the model to the objective function. Any possible further decrease in function values obtained within these additional steps is then taken into account by Step 5.

We finally mention that models other than full quadratics of the form (3.1) are also acceptable. This may be only necessary for the case where Y has to be restricted in Step 2 because it is not poised, in which case a model may be built that includes a basis of functions that does not necessarily span the full space of all quadratic polynomials. Models that are less than fully quadratic or even less than fully linear are also beneficial when function evaluations are expensive since one is then able to use these evaluations as soon as they are available. This is typically the case during the early iterations of the algorithm. On the other hand, the use of models of degree exceeding two has been suggested in [10].

We now return to the important question of providing a computable measure for the quality of the geometry of Y, which is necessary for substantiating the procedures of Step 3 for inclusion of x_k^+ and geometry improvement. We will consider two approaches here.

4 An approach using the Lagrange fundamental polynomials

The first approach is that of Powell in [25] and [26]. The idea is to measure the geometric quality of the model by the value of the scaled interpolation determinant $\delta(Y)$ relative to its theoretical maximum. More precisely, we follow Powell [25] and say that the geometry of Y is *adequate*[7] (with respect to a current iterate x_k and a trust-region radius Δ) when all the points in Y are no further away from x_k than 2Δ and when the scaled value of $|\delta(Y)|$ cannot be doubled by replacing one of the points of Y by an alternative value within distance Δ from x_k.

We first consider improving the geometry of Y by introducing a new interpolation point y_+ in Y such that $\|y_+ - x_k\| \le \Delta$ (together with its associated function value $f(y_+)$). This usually[8] means that we have to drop an existing interpolation point y_- from Y. We consider two cases. First, if $\|y_- - x_k\| \le \Delta$, a suitable measure is $|\delta(Y)|$, and we therefore wish to compute the factor by which $|\delta(Y)|$ is multiplied when y_- is replaced by y_+. Remarkably, this factor is independent of the basis $\{\phi_i\}$ and is equal to $|L(y_+, y_-)|$, where $L(\cdot, y_-)$ is the Lagrange interpolation function whose value is one at y_- and is zero at all other points of Y[9]. This very nice result was pointed out by Powell in [25]. Hence, if $\|y_- - x_k\| \le \Delta$, it makes sense to replace y_- by

$$y_+ = \arg \max_{\|y - x_k\| \le \Delta} |L(y, y_-)|. \tag{4.1}$$

[7] Powell uses the term "good".

[8] This may not be the case during the early stages of the algorithm.

[9] Note that $L(\cdot, \cdot)$ thus depends on all points in Y, and not just on its two explicit arguments.

On the other hand, if $\|y_- - x_k\| > \Delta$, it is important to take this inequality into account when choosing a suitable replacement y_+. One possible method is to compare y_+ not with y_- directly, but rather with the best point on the segment joining x_k to y_- limited to the ball of radius Δ around x_k. This "scaled down" version of y_- is the vector that maximizes $|L(x_k + td_-, y_-)|$ for $t \in [0, \Delta]$, where $d_- = (y_- - x_k)/\|y_- - x_k\|$. Hence, y_+ may be chosen in this case as

$$y_+ = \arg \max_{\|y - x_k\| \le \Delta} S(y, y_-), \tag{4.2}$$

where

$$S(y, y_-) = \frac{|L(y, y_-)|}{\min[1, \max_{t\in[0,\Delta]} |L(x_k + td_-, y_-)|]}. \tag{4.3}$$

The minimum in the denominator of (4.3) guarantees that the scaled down version of y_-, namely $\arg\max_{t\in[0,\Delta]} |L(x_k + td_-, y_-)|$, is treated exactly as any other point within distance Δ from x_k (that is according to (4.1)). We may therefore define our improvement procedure as the replacement of y_- by y_+, where we have chosen y_- to maximize $S(y_+, y_-)$ for all choices of $y_- \in Y \setminus \{x_k\}$, and where y_+ is defined by (4.2), given y_-.

Note also that the Lagrange fundamental polynomials (in this case quadratic) are well-defined, together with $S(\cdot, \cdot)$, if and only if Y is poised. Furthermore, problem (4.1) or (4.2) are thus of the same form as the trust-region subproblem (3.3).

The geometry of Y is then said to be adequate (with respect to x_k and Δ) when

$$\|y - x_k\| \le 2\Delta \text{ for all } y \in Y \tag{4.4}$$

and no point y_+ can be found such that

$$\|y_+ - x_k\| \le \Delta \text{ and } S(y_+, y_-) > 2 \text{ for at least one } y_- \in Y \setminus \{x_k\}. \tag{4.5}$$

If this is not the case, then a new point y_+ within distance Δ of x_k can be found if (4.4) is violated, or the point y_+ given by (4.2) can be used to replace the corresponding y_- if (4.5) is violated. In this latter case the gain of the replacement of y_- by y_+ is then given by $S(y_+, y_-)$.

Note that verifying the adequacy of the geometry of Y not only involves checking (4.4), a relatively simple task, but also the solution of $2(p-1)$ constrained quadratic maximization problems. Indeed, we may write

$$\max_{\|y-x_k\|\le\Delta} S(y, y_-) = \frac{\max\left[\max_{\|y-x_k\|\le\Delta} L(y, y_-), -\min_{\|y-x_k\|\le\Delta} L(y, y_-)\right]}{\min[1, \max_{t\in[0,\Delta]} |L(x_k + td_-, y_-)|]}$$

and the two optimization problems of the numerator must be solved for each $y_- \in Y \setminus \{x_k\}$. This is not unreasonable since we have assumed that the cost of evaluating the objective function dominates all other costs, but nevertheless constitutes a significant computational task when the number of problem variables grows.

To complete the description of this first approach, we only need to describe how we choose a point y_- to drop from Y when necessary when we include x_k^+. Using the same idea as above, we see that a reasonable choice is to choose

$$y_- = \arg\max_{y\in Y} S(x_k^+, y_-). \tag{4.6}$$

One could also attempt to restrict the maximization, in this last definition, to the subset of Y consisting of points which are at a distance exceeding $2\Delta_k$, if such points exist. Or one might want to compromise between these two techniques by accepting a y_- from the restricted set only if $S(x_k^+, y_-) > 1$ and resorting to (4.6) if no such point can be found.

It is interesting to note that this approach can be viewed as an attempt to reduce the Lebesgue constant associated with the interpolation operator, which is given by

$$\max_{\|y-x_k\|\leq\Delta} \sum_{y_-\in Y} |L(y, y_-)|,$$

(see [23], Sections 3.2 and 4.4). The choice (4.1) then aims at reducing one term of the sum in this definition.

Algorithms based on this first approach have been described in Powell [25], Conn and Toint [10] and Powell [26]. Preliminary numerical results have been presented in the last two contributions, which indicate that remarkably efficient algorithms can be designed along these lines.

5 An approach using Newton fundamental polynomials

We now consider a second approach to measuring the quality of the geometry of Y, and the implimentation of the multivariate interpolation techniques upon which it is based. This approach was introduced in Conn, Scheinberg and Toint [9] with both a theoretical and a practical motivation. From the theoretical point of view, this approach allows global convergence to be proved for the associated version of our algorithmic outline. On the more practical side, this second approach sometimes significantly reduces the calculations performed by the algorithm above and beyond the objective function evaluations. This is very desirable when the number of variables is not very small or when the cost of evaluating the objective function is not exceptionally high. At variance with the techniques of the previous section, we will emphasize here the use of Newton fundamental polynomials. It is important to realize that, for a given interpolation space, even if the interpolation techniques are different, the interpolating model is nevertheless the same whenever Y is poised. The motivation for introducing a new interpolation technique is therefore not to modify the model for a given Y, but rather to derive from the new technique better procedures for improving the geometry of Y and for including x_k^+ in the interpolation set, and hopefully a more efficient method of determining and updating the model. In order to continue the discussion, we need to consider some basic multivariate interpolation techniques, that is we consider the general problem of finding interpolating polynomials of degree d. In contrast with fundamental Lagrangian polynomials, in which the basis functions in this case would all be of degree d, Newton fundamental polynomials are of increasing degree arranged by blocks. In this framework, the points y in the interpolation set Y are also organized into $(d+1)$ *blocks* $Y^{[\ell]}$, $(\ell = 0, \ldots, d)$, the ℓ-th block containing $|Y^{[\ell]}| = \binom{\ell+n-1}{\ell}$ points. To each point $y_i^{[\ell]} \in Y^{[\ell]}$ corresponds a single *Newton fundamental polynomial* of degree ℓ satisfying the conditions

$$N_i^{[\ell]}(y_j^{[m]}) = \delta_{ij}\delta_{\ell m} \text{ for all } y_j^{[m]} \in Y^{[m]} \text{ with } m \leq \ell.$$

The interpolating polynomial $m(x)$ is then given as

$$m(x) = \sum_{y_i^{[\ell]} \in Y} \lambda_i^{[\ell]}(Y, f) N_i^{[\ell]}(x),$$

where the coefficients $\lambda_i^{[\ell]}(Y, f)$ are generalized finite differences applied on f. We refer the readers to [28] for more details on these and multivariate interpolation in general.

We now return to the concept of poisedness and consider in more detail the procedure of constructing the basis of fundamental Newton polynomials as described in [28]. Namely we consider the procedure below for any given Y.

Procedure CNP for constructing fundamental Newton polynomials

Initialize the $N_i^{[\ell]}$ $(i = 1, \ldots, |Y^{[\ell]}|, \ell = 0, \ldots, d)$ to the chosen polynomial basis (say, the monomials).
Set $Y_{temp} = \emptyset$.
For $\ell = 0, \ldots, d$,
  for $i = 1, \ldots, |Y^{[\ell]}|$
    choose some $y_i^{[\ell]} \in Y \setminus Y_{temp}$ such that $|N_i^{[\ell]}(y_i^{[\ell]})| \neq 0$,
    if no such $y_i^{[\ell]}$ exists in $Y \setminus Y_{temp}$, reset $Y = Y_{temp}$ and stop
      (the basis of Newton polynomials is incomplete),
    $Y_{temp} \leftarrow Y_{temp} \cup \{y_i^{[\ell]}\}$
    normalize the current polynomial by

$$N_i^{[\ell]}(x) \leftarrow N_i^{[\ell]}(x)/|N_i^{[\ell]}(y_i^{[\ell]})|, \tag{5.1}$$

    update all Newton polynomials in block ℓ and above by

$$N_j^{[\ell]}(x) \leftarrow N_j^{[\ell]}(x) - N_j^{[\ell]}(y_i^{[\ell]}) N_i^{[\ell]}(x) \quad (j \neq i, j = 1, \ldots, |Y^{[\ell]}|),$$

$$N_j^{[k]}(x) \leftarrow N_j^{[k]}(x) - N_j^{[k]}(y_i^{[\ell]}) N_i^{[\ell]}(x) \quad (j = 1, \ldots, |Y^{[k]}|, k = \ell + 1, \ldots, d).$$

  end
End (the basis of Newton polynomials is complete).

For instance, if we consider quadratic interpolation on a regular grid in the plane, we require six interpolation points using four blocks

$$Y^{[0]} = \{(0,0)\}, \quad Y^{[1]} = \{(1,0),(0,1)\}, \text{ and } Y^{[2]} = \{(2,0),(1,1),(0,2)\},$$

corresponding to the basis functions $1, x_1, x_2, x_1^2, x_1 x_2$ and x_2^2 respectively. Applying procedure CNP then yields

$$N_1^{[0]} = 1, \quad N_1^{[1]} = x_1, \quad N_2^{[1]} = x_2,$$
$$N_1^{[2]} = \tfrac{1}{2}(x_1^2 - x_1), \quad N_2^{[2]} = x_1 x_2 \text{ and } N_3^{[2]} = \tfrac{1}{2}(x_2^2 - x_2).$$

Clearly, poisedness relates to non-zero pivots in (5.1). Notice that after applying procedure CNP, Y is always poised since we only include the points that create non-zero pivots. This is true even if the procedure stops with an incomplete basis of Newton polynomials, which then results in an interpolating polynomial which is not of full degree d (meaning that it does not include contributions of all the monomials of degree d or less, see Step 2 of the algorithm). In practice we need sufficiently large pivots, which is equivalent to "well-poisedness". Thus checking if $|N_i^{[\ell]}(y_i^{[\ell]})| \neq 0$ is replaced by $|N_i^{[\ell]}(y_i^{[\ell]})| \geq \theta$, for some $\theta > 0$. We call θ the pivot threshold. In [9], the authors have shown that if throughout the algorithm the interpolation problem can be made sufficiently well-poised we are able to assure the existence of a bound on the distance between the interpolating polynomial and interpolated function at a point x ($x \notin Y$). Otherwise we provide a mechanism that guarantees we can find a suitable interpolant for which the bound holds. This bound depends upon an important property proved by Sauer and Xu and uses the concept of a *path* between the zero-th block and x, which uses a sequence of points of Y of the form

$$\pi(x) = (y_0^{[0]}, y_1^{[1]}, \ldots, y_d^{[d]}, y_{d+1}^{[d+1]} = x)$$

where

$$y_i^{[i]} \in Y^{[i]} \quad (i = 0, \ldots, d).$$

A path therefore contains, besides x itself, exactly one interpolation point in each block. Let us denote by $\Pi(x) = \{\pi(x)\}$, the set of all possible paths from $Y^{[0]}$ to x. Using this notion, Sauer and Xu [28] derive in their Theorem 3.11 a bound on $|f(x) - m(x)|$, where $m(x)$ is the polynomial interpolating the function $f(x)$ at the points in Y. This bound was further simplified by Sauer [27], giving that

$$|f(x) - m(x)| \leq \frac{n^{d+1}\|f^{(d)}\|_\infty}{(d+1)!} \sum_{\pi(x)\in\Pi(x)} \left[\prod_{i=0}^{d} \|y_{i+1}^{[i+1]} - y_i^{[i]}\|_\infty \, |N_i^{[i]}(y_{i+1}^{[i+1]})| \right],$$

for all x, where $f^{(d)}$ is the d-th derivative of f. Interestingly, the quantities $N_i^{[i]}(y_{i+1}^{[i+1]})$ are all computed in the course of the evaluation of the generalized finite differences $\lambda_i^{[\ell]}(Y, f)$. We see that the error between $m(x)$ and $f(x)$ is smaller if we can make the values $N_i^{[i]}(y_{i+1}^{[i+1]})\|y_{i+1}^{[i+1]} - y_i^{[i]}\|_\infty$ small. If all the interpolation points and the point x are chosen in a given hypersphere of radius δ, it is then possible to provide an upper bound on the maximal error. More precisely, the following theorem can be proved (see [9], Theorem 2).

Theorem 1 *Assume that an arbitrary $x_k \in \mathbf{R}^n$ and a $\Delta_k > 0$ are given, together with the interpolation degree d. Then it is possible to construct an interpolation set Y yielding a complete basis of Newton polynomials such that all $y \in Y$ satisfy*

$$y \in \mathcal{B}_k$$

and also that

$$|N_j^{[\ell]}(x)| \leq \kappa_0.$$

for all $\ell = 0, \ldots, d$, all $j = 1, \ldots, |y^{[\ell]}|$ and all $x \in \mathcal{B}_k$, and where the positive constant κ_0 is independent of x_k and Δ_k.

We now return to the case of quadratic models of the form (3.1) and define what we mean by an adequate geometry of the interpolation set. In the spirit of Theorem 1, we assume that we are at iteration k of the algorithm, where x_k is known (but arbitrary). We then say that Y is *adequate in* $\mathcal{B}_k$ whenever the cardinality of Y is at least $n+1$ (which means the model is at least fully linear) and

$$y \in \mathcal{B}_k \ \text{ for all } \ y \in Y,$$

$$|N_i^{[\ell]}(y_j^{[\ell+1]})| \leq \kappa_1 \quad (i = 1, \ldots, |Y^{[\ell]}|, j = 1, \ldots, |Y^{[\ell+1]}|, \ell = 0, \ldots, d-1), \tag{5.2}$$

and

$$|N_i^{[d]}(x)| \leq \kappa_1 \quad (i = 1, \ldots, |Y^{[d]}|, x \in \mathcal{B}_k), \tag{5.3}$$

where κ_1 is any positive constant such that $\kappa_1 > \kappa_0$. (This choice of κ_1 is merely intended to make (5.2) and (5.3) possible in view of Theorem 1.)

The inclusion of x_k^+ in Y is then simply defined as follows: we may add x_k^+ to Y if $|Y| < p$, and we need to remove a point y_- of Y, if $|Y|$ is already maximal ($|Y| = p$). Ideally, this point should be chosen to make the geometry of Y as good as possible. For instance, one might choose to remove $y_- = y_i^{[\ell]}$ such that $|N_i^{[\ell]}(x)|$ is maximal, therefore trying to make the pivots as large as possible, but other techniques may be preferable.

The last procedure that we have to describe is the model's geometry improvement in $\mathcal{B}_k$, which promotes making Y adequate in $\mathcal{B}_k$. Again, many different techniques are possible. For instance, a reasonable strategy consists of first eliminating a point $y_i^{[\ell]} \in Y$ which is not in $\mathcal{B}_k$ (if such a point exists), and replacing it in Y by

$$y_+ = \arg\max_{x \in \mathcal{B}_k} |N_i^{[\ell]}(x)|. \tag{5.4}$$

If no such exchange is possible, one may then consider replacing interpolation points in $Y \setminus \{x_k\}$ by new ones, again using (5.4). The theory of [9] then ensures that (5.2) and (5.3) hold after a finite number of such replacements. A computationally expensive version of the improvement procedure would compute y_+ for every possible choice of $y_- = y_i^{[\ell]}$, and then select that for which $|N_i^{[\ell]}(y_+)|$ is maximal, but substantially cheaper versions can be designed by choosing y_- from the information which has already been calculated when the interpolation model is computed. For instance, one may consider the vectors $y_i^{[\ell]}$ corresponding to polynomials for which $|N_j^{[\ell-1]}(y_i^{[\ell]})|$ is large for some j, or one may choose to replace fundamental polynomials corresponding to small pivots in Algorithm CNP. A closer look at the mechanism of this latter algorithm furthermore indicates that significant computational savings can be achieved if the polynomial to be replaced is selected in $Y^{[d]}$, or at least in the blocks of higher index, whenever possible.

The interested reader will find the global convergence theory associated with algorithms using this approach in [9].

6 Discussion and perspectives

We have shown so far that one can design derivative free trust-region algorithms for unconstrained minimization in a variety of ways, but that it is important for all these algorithms to take the geometry of the set of interpolation points Y into account. This is the main feature that distinguish modern methods from the proposal of Winfield in [34] and [35]. The crucial point is that this geometry should remain as adequate as possible, in the sense that the interpolation problem remains as far as possible from being ill-defined. We have examined two potential approaches that provide different means to ensure this requirement in practical computational methods. The first is based on the use of the Lagrange fundamental polynomials but is potentially expensive computationally. The second uses the Newton fundamental polynomials and also provides a framework for which global convergence can be proved for the resulting algorithm. Possible simplifications that reduce the computational complexity of the minimization method are suggested in this latter.

Significant work remains to be done for assessing the various possible algorithms discussed in this paper. In particular, it is desirable to explore to a much greater extent the computational tradeoffs between "best possible geometry" and an acceptable amount of calculation as the relative cost of the objective function evaluation and internal linear algebra varies. This work is currently ongoing and will be reported upon in the near future.

Besides this assessment, other questions of interest still remain. We have barely touched here the question of choosing initial models at the early iterations of the algorithm, when the cardinality of Y does not yet allow one to define a full quadratic model. Possible choices are as diverse as methods based upon statistical design of experiments and analogies with the theory of "thin plate" spline functions. More questions related to the theoretical significance of pivot values in the context of Newton fundamental polynomials are also open and interesting.

An important direction for future work is the extension of the techniques discussed here to handle problems with a larger number of variables. Clearly it would be desirable to exploit any structure present in the problem as efficiently as possible. For instance, if the Hessian of the objective function has a known sparsity pattern, this fact can be exploited by suitably restricting the basis of monomials spanning the desired interpolation space: the monomial $x_i x_j$ may be removed from this basis if the (i,j)-th entry of the Hessian is known to be zero. Another possibility is to exploit partial separability (see Griewank and Toint [16] or Conn, Gould and Toint [6]) in the objective function when possible. The idea would then be to build independent interpolation models for the elements of the objective function, maybe coupled with a structured trust-region scheme (see [8]).

Another direction of research, both useful and challenging, is to consider how the methods described here can be adapted to cases where some noise is present in the evaluation of the objective function. In this case, one expects that interpolation will be replaced by approximation in the model's definition, and the conditions on geometry will have then to be combined with that of sufficient sampling.

In addition, we intend to extend these results to general constraints. Not surprisingly, an approach based upon penalty functions may be suitable - although any resulting non-smoothness must be accounted for. We note here that our algorithms currently do handle simple bounds (using the modification of the trust-region scheme described in [7]). Finally, the authors acknowledge that the potential of the methods outlined in this paper will only be fully realized when associated high quality software will become available to users. The development of such software is again the subject of ongoing work. We intend to report some numerical result in the presentation.

To conclude our presentation, we wish to stress that derivative free methods for optimization remains a thriving and valuable area for research. It is indeed very encouraging that it presents such a remarkable combination of interesting theoretical concepts, useful algorithmic designs and high demand from potential users.

References

[1] I. Bongartz, A. R. Conn, N.I.M. Gould, and Ph. L. Toint. CUTE: Constrained and Unconstrained Testing Environment. *ACM Transactions on Mathematical Software*, 21(1):123–160, 1995.

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

[3] R. P. Brent. *Algorithms for Minimization Without Derivatives.* Prentice-Hall, Engelwood Cliffs, USA, 1973.

[4] F. M. Callier and Ph. L. Toint. Recent results on the accelerating property of an algorithm for function minimization without calculating derivatives. In A. Prekopa, editor, *Survey of Mathematical Programming*, pages 369–376. Publishing House of the Hungarian Academy of Sciences, 1977.

[5] I. G. Campey and D. G. Nickols. Simplex minimization. Program specification, Imperial Chemical Industries Ltd, UK, 1961.

[6] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. An introduction to the structure of large scale nonlinear optimization problems and the LANCELOT project. In R. Glowinski and A. Lichnewsky, editors, *Computing Methods in Applied Sciences and Engineering*, pages 42–51, Philadelphia, USA, 1990. SIAM.

[7] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. LANCELOT*: a Fortran package for large-scale nonlinear optimization (Release A).* Number 17 in Springer Series in Computational Mathematics. Springer Verlag, Heidelberg, Berlin, New York, 1992.

[8] A. R. Conn, Nick Gould, A. Sartenaer, and Ph. L. Toint. Convergence properties of minimization algorithms for convex constraints using a structured trust region. *SIAM Journal on Optimization*, 6(4):1059–1086, 1996.

[9] A. R. Conn, K. Scheinberg, and Ph. L. Toint. On the convergence of derivative-free methods for unconstrained optimization. In A. Iserles and M. Buhmann, editors, *Approximation Theory and Optimization: Tributes to M. J. D. Powell*, pages 83–108, Cambridge, UK, 1997. Cambridge University Press.

[10] A. R. Conn and Ph. L. Toint. An algorithm using quadratic interpolation for unconstrained derivative free optimization. In G. Di Pillo and F. Gianessi, editors, *Nonlinear Optimization and Applications*, pages 27–47, New York, 1996. Plenum Publishing.

[11] C. De Boor and A. Ron. Computational aspects of polynomial interpolation in several variables. *Mathematics of Computation*, 58(198):705–727, 1992.

[12] J. E. Dennis and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. Prentice-Hall, Englewood Cliffs, USA, 1983.

[13] J. E. Dennis and V. Torczon. Direct search methods on parallel machines. *SIAM Journal on Optimization*, 1(4):448–474, 1991.

[14] A. Griewank. Computational differentiation and optimization. In J. R. Birge and K. G. Murty, editors, *Mathematical Programming: State of the Art 1994*, pages 102–131, Ann Arbor, USA, 1994. The University of Michigan.

[15] A. Griewank and G. Corliss. *Automatic Differentiation of Algorithms*. SIAM, Philadelphia, USA, 1991.

[16] A. Griewank and Ph. L. Toint. On the unconstrained optimization of partially separable functions. In M. J. D. Powell, editor, *Nonlinear Optimization 1981*, pages 301–312, London and New York, 1982. Academic Press.

[17] R. Hooke and T. A. Jeeves. Direct search solution of numerical and statistical problems. *Journal of the ACM*, 8:212–229, 1961.

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

[19] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer Journal*, 17:155–162, 1964.

[20] M. J. D. Powell. A new algorithm for unconstrained optimization. In J. B. Rosen, O. L. Mangasarian, and K. Ritter, editors, *Nonlinear Programming*, New York, 1970. Academic Press.

[21] M. J. D. Powell. Unconstrained minimization algorithms without computation of derivatives. *Bollettino della Unione Matematica Italiana*, 9:60–69, 1974.

[22] M. J. D. Powell. A view of unconstrained minimization algorithms that do not require derivatives. *ACM Transactions on Mathematical Software*, 1(2):97–107, 1975.

[23] M. J. D. Powell. *Approximation Theory and Methods*. Cambridge University Press, Cambridge, UK, 1981.

[24] M. J. D. Powell. 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, Oaxaca, Mexico*, volume 275, pages 51–67, Dordrecht, NL, 1994. Kluwer Academic Publishers.

[25] M. J. D. Powell. A direct search optimization method that models the objective by quadratic interpolation. Presentation at the 5th Stockholm Optimization Days, 1994.

[26] M. J. D. Powell. Trust region methods that employ quadratic interpolation to the objective function. Presentation at the 5th SIAM Conference on Optimization, 1996.

[27] Th. Sauer. Notes on polynomial interpolation. Private communication, February 1996.

[28] Th. Sauer and Yuan Xu. On multivariate Lagrange interpolation. *Mathematics of Computation*, 64:1147–1170, 1995.

[29] W. Spendley, G. R. Hext, and F. R. Himsworth. Sequential application of simplex designs in optimization and evolutionary operation. *Technometrics*, 4, 1962.

[30] Ph. L. Toint. *Unconstrained optimization: the analysis of conjugate directions method without derivatives and a new sparse quasi-Newton update.* PhD thesis, Department of Mathematics, FUNDP, Namur, Belgium, 1978.

[31] Ph. L. Toint and F. M. Callier. On the accelerating property of an algorithm for function minimization without calculating derivatives. *Journal of Optimization Theory and Applications*, 23(4):531–547, 1977. See also same journal, vol. 26(3), pp. 465–467, 1978.

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

[33] V. Torczon. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 7(1):1–25, 1997.

[34] D. Winfield. *Function and functional optimization by interpolation in data tables.* PhD thesis, Harvard University, Cambridge, USA, 1969.

[35] D. Winfield. Function minimization by interpolation in a data table. *Journal of the Institute of Mathematics and its Applications*, 12:339–347, 1973.

[36] M. H. Wright. Direct search methods: once scorned, now respectable. In D. F. Griffiths and G. A. Watson, editors, *Proceedings of the 1995 Dundee Biennal Conference in Numerical Analysis*, Harlow, UK, 1996. Addison Wesley Longman.