A grid algorithm for bound constrained optimization of noisy functions

Clemens Elster
Physikalisch-Technische Bundesanstalt
Abbestr. 2-12
D-10587 Berlin
Germany
email: celster@chbrb.berlin.ptb.de

Arnold Neumaier
Institut für Mathematik
Universität Wien
Strudlhofgasse 4
A-1090 Wien
Austria
email: neum@cma.univie.ac.at

ABSTRACT:

The optimization of noisy functions is a common problem occurring in various applications. In this paper, a new approach is presented for low-dimensional bound constrained problems, based on the use of quadratic models and a restriction of the evaluation points to successively refined grids. In the noiseless case, global convergence of the algorithm to a stationary point is proved; in the noisy case a bound for the limit accuracy is derived.

An extensive numerical comparison with two widely used methods – a quasi-Newton method and the simplex method of Nelder and Mead – performed on a standard collection of test problems, shows that the new algorithm is comparable with quasi-Newton in the noiseless case, and is much more robust than Nelder-Mead in the noisy case. If performance is measured solely by the number of function evaluations needed to achieve a prescribed reduction of the difference to the minimal function value (as for instance in the optimization of experiments), the new algorithm is also significantly faster than Nelder-Mead.

Revised version, February 1995

KEY WORDS:
bound constrained optimization, noisy functions, Nelder-Mead method, quasi-Newton method, grid algorithm, quadratic model.
1991 MSC Classification: 90C30

1 Introduction

We consider problems of the form

$$\min f(x) \quad \text{s.t.} \quad x \in \Omega = [a_1, b_1] \times \ldots \times [a_n, b_n]\,, \tag{1}$$

where f is assumed to be smooth. Information about $f(x)$ shall be gained only in the form of $f^\sigma(x)$, where $f^\sigma(x)$ is an approximation to the true function value $f(x)$, contaminated by a small amount of noise η:

$$|f^\sigma(x) - f(x)| \leq \Delta. \tag{2}$$

Such problems are common in real life applications as in the optimization of parameters in chemical experiments or finite element calculations, where a single measurement (function evaluation) takes several hours or even days. With such applications in mind, we ask for robust methods which make good progress with the fewest possible number of function evaluations, while the work to select new evaluation points can be neglected. We also demand that the function is evaluated only at feasible points since this is often essential in applications.

No knowledge of the statistical properties of the noise is assumed. In particular, the noise may be deterministic (due to modeling errors, truncation errors, or discretization errors) or stochastic (due to inaccurate measurements or rounding errors). The bounds in (2) are required only for the analysis of the limit accuracy, and a similar analysis could be given for multiplicative noise with $|f^\sigma(x) - f(x)| \leq \Delta |f(x)|$.

In the noiseless case, powerful tools for solving problem (1) exist that are closely related to the methods of unconstrained optimization, which ignore the bounds in (1). For instance, Newton type methods and quasi-Newton methods using a (approximated or possibly modified) Hessian have proved very powerful, and strong statements on their convergence properties are known. Although these methods assume information about gradient or Hessian it is believed that quasi-Newton methods using a finite-difference approximation of the gradient are most efficient for the task of optimizing smooth functions when only function values are available [4], [14], [6].

In the case of substantial noise, however, the obtained function values show a discontinuous behaviour, and quasi-Newton methods using a finite-difference estimate of the gradient break down. Gill et al. [6], suggest in Section 8.6.2.2 some remedy through larger difference intervals, but this requires knowledge of the precision of $f^\sigma(x)$. This can also be estimated ([6], Section 8.5.2.3), but only at the expense of further function evaluations, and the estimate must be repeated regularly when the precision depends on x.

For noisy function optimization, the usually recommended method (e.g., Nocedal [14], p.202) and certainly the most used one (cf. Powell [15]) is the simplex method of Nelder & Mead [13] which makes no smoothness assumptions but has poor convergence properties.

In this paper we propose an algorithm based on the use of quadratic models minimized over adaptively refined trust regions together with the restriction of the evaluation points on a sequence of nested grids; this exploits both the fact that the underlying function f is smooth and the need for robustness with respect to noise.

However, since bound constrained problems frequently have several local minima, we made some attempt to obtain a reasonable good local (if not global) minimum *without significantly increasing the number of function evaluations.* This contrasts with fully local techniques which try to estimate gradients, and therefore often move to the closest minimizer, and also with stochastic techniques (cf. Törn & Žilinskas [17], Byrd et al. [2], Rinnooy Kan & Timmer [16]), which attempt to obtain a global optimizer but typically need thousands of function values even for low-dimensional problems.

Section 2 gives the motivation and details of the algorithm. As currently implemented, the dominant linear algebra work occurs in the least squares estimation of the quadratic model which takes $O(n^6)$ operations. Finding a stationary point for the (possibly indefinite) quadratic program satisfying the second-order necessary conditions for a minimum might also be expensive (though it usually is $O(n^3)$ only). The amount of operations per iteration restricts applications to expensive objective functions with low dimensions; for $n \leq 12$, the linear algebra requires at most about $2 \cdot 10^6$ operations per iteration ($12^6 \approx 3 \cdot 10^6$), which is acceptable in many circumstances.

In Section 3, convergence properties of our algorithm are investigated. In the noiseless case, our algorithm converges to the set of stationary points, thus improving on convergence results for other optimization algorithms using (noiseless) function evaluations only (see Torczon [19, 20] and references quoted there) which only prove the existence of a subsequence converging to a stationary point. In the noisy case, we are able to prove that our algorithm generates at least one point with a gradient of an order of magnitude which is optimal for a given noise level. The only previous result of this type is due to Glad & Goldstein [7], but our assumptions for sufficient decrease are much weaker than theirs. Our proof techniques are based on arguments of the type used in the papers cited above, combined and extended to give the sharper results for our algorithm.

Section 4 summarizes extensive numerical tests which compare the new method with a widely used quasi-Newton method in the noiseless case and with modifications of the Nelder-Mead method on noisy functions.

A comparison with Nelder-Mead is appropriate in spite of a wide-spread negative assessment of this method (expressed, e.g., in [6], Section 4.2.1, but valid only in the noiseless case). Indeed, in the presence of noise, we checked that Nelder-Mead performed on the average best among the publicly available algorithms for *unconstrained* optimization (from IMSL, NAG, and netlib). Although no convergence statements can be made and the method may get stuck on very simple problems in high dimensions ($n \geq 8$ for Penalty I, $n \geq 16$ for $f(x) = x^t x$;

see Torczon [18]), this method has proved to be useful in many low-dimensional applications.

The deficiencies of Nelder-Mead (caused by the appearance of very flat simplices) can be avoided by suitable variants as discussed for example by Gill et al. [6] or Torczon [18]; the latter variant is rather slow but provably convergent, using the fact that the points generated lie on a sequence of nested grids, a point of contact with the present method. More sophisticated major departures from the simplex method are discussed by Powell [15] who assumes that the objective function can be evaluated at infeasible points. However, for the optimization of parameters in experiments or finite element calculations, reasonable function values can often be computed in the feasible region only.

Very recently, we also learned of an old paper of Dixon [3] which combines a simplex technique similar to [13] with the use of occasional quadratic models, estimated by some kind of finite differences (Dixon's modules 5 and 6). The method works in the presence of arbitrary constraints, and does not require function values at infeasible points. A comparison with the results quoted there (for the noiseless case, with different constraints) suggests that our algorithm is somewhat faster.

Acknowledgments. This work was done while the first author was at the Freiburger Materialforschungszentrum and the second author was at the Institut für Angewandte Mathematik, both at the Universität Freiburg, Germany. Detailed remarks of the referees and the associate editor resulted in significant improvements in the discussion of the algorithm and the comparison with other methods. In particular, this lead to the recognition of the value of Phase I as preprocessing phase for quasi-Newton methods in the noiseless case. We also want to thank Virginia Torczon for useful discussions on the history of the subject.

2 The algorithm

Our algorithm proceeds in three phases: Phase I is a starting procedure which tries to locate a good point on the boundary of the box, Phase II consists of a descent phase on a grid with suitable spacing, using a trust region approach to locate suitable trial grid points. Phase III checks whether the current grid spacing is still adequate, or whether the grid should be refined. After leaving Phase I, Phase II and Phase III alternate until convergence.

Restricting evaluations to grid points allows us to give a simple convergence proof for our algorithm. The spacing of the grid will be reduced adaptively during the optimization process, but it is never increased. Except for the initial vector x^0 of variables, provided by the user of the algorithm, all function evaluations are taken on points of the current grid only. Hence the optimization of f is performed on successively reduced scales with an attempt to maintain the scale as large as possible. Since the points are allowed to cluster on small scales only late in the search, this leads to more robust quadratic model fits in the earlier stages, a feature which reduces the influence of noise in the function evaluations.

We also observed that the imposition of the grid structure improved the performance of the algorithm at times where the quadratic approximation of f was not very good, partially since the function is explored first over large regions. This also enhances the possibility of finding a low lying local optimum far from the starting point.

On the other hand, the restriction of evaluation points to grid points makes the local convergence behavior worse. At least n function evaluations are needed to check reliably whether the grid needs to be reduced, so we can have only local linear convergence (with convergence factor 0.1, the factor we use for grid refinement).

The grids are obtained by splitting, for each component index $\nu = 1, \dots, n$, the constraint interval $[a_\nu, b_\nu]$ into M intervals of equal size $(b_\nu - a_\nu)/M$. This has the advantage that, initially, the feasible domain is searched on the largest possible scale, but it is appropriate only if one assumes that the optimal solution is expected to have components whose order of magnitude is about that of their bounds. (For problems whose bounds are of widely differing magnitudes but where the optimal variables are expected to have a similar magnitude, one should instead split first only the widest intervals, and split more intervals later as their width becomes larger than the step size of the grid. One could also try to estimate correctly scaled steps in each component by looking at the Hessian information obtained during the algorithm.) Acceptable points for evaluation are the *h-grid points*, i.e., the points $x \in \Omega$ such that $(x_\nu - a_\nu)/(b_\nu - a_\nu)$ is an integral multiple of the normalized grid size $h := 1/M$.

In view of the limits on the dimension ($n \leq 12$) and number of function evaluations ($N_{\max} = 200$) for which the algorithm is designed, we can afford to store all old points and associated function values. For larger problems the points could be stored with integral coordinates since all but the first lie on a grid, and one could use *k*-*d*-trees to speed up the check for known function values at trial points. Alternatively, one could limit the storage to a fixed number of old points closest to the best point found. A limit of 50 points did not significantly change the results reported in Section 4, but this is likely to be dimension dependent.

We also avoid repeated evaluation at the same grid point by checking each trial point against all previous grid points. In case of stochastic noise, a high function value could be a chance effect. But in such situations, the mean function values at all nearby points on sufficiently fine grids have lower values, too, so that this case is handled automatically by the grid refinement.

In the following we measure closeness of points by the scaled maximum norm appropriate for the box Ω, namely

$$\|x\|_\Omega = \max_{\nu=1}^{n} \frac{|\, x_\nu \,|}{b_\nu - a_\nu}. \tag{3}$$

Points at which f is evaluated are denoted by x^i. The best grid point found in each stage is called x^*, and the overall best point $x^\dagger$ (including the starting point which may be off-grid). The different phases will be described separately; the outline of each phase is followed by the details on the actual implementation.

Phase I (starting phase):

STEP 1. Evaluate f at the vertex x^1 closest to and the vertex x^2 farthest away from x^0, and denote by x^* the point with the lower function value obtained.

STEP 2. Set $\nu = 1$ and $x^\dagger = x^*$.

STEP 3. Define $y_\alpha = x^*_\alpha, \alpha \neq \nu$, $y_\nu = \begin{cases} a_\nu & \text{if } x^*_\nu = b_\nu, \\ b_\nu & \text{otherwise.} \end{cases}$

STEP 4. If $f^\sigma(y) < f^\sigma(x^*)$ set $x^* = y$ and $x^\dagger = x^*$.

STEP 5. If $\nu < n$, set $\nu = \nu + 1$ and goto Step 3.

STEP 6. If the starting guess x^0 was one of the vertices already visited, set $N = n+2$. Otherwise evaluate $f^\sigma(y)$ at $y = x^0$, update $x^\dagger = y$ when f^σ improved, and set $N = n + 3$. (Do not update x^*, which is required to lie on the grid!)

STEP 7. Set the trust region radius to $\rho = 1$, the normalized grid size to $h = 1/10$, the level to $g = 1$ and the 'no progress' counter to $s = 0$, and enter Phase II.

This procedure is motivated by the fact that many practical problems of the form (1) have solutions on the boundary of Ω. Thus, the starting procedure consists of coordinate relaxations from a vertex of the box towards an opposite boundary in order to find a good boundary point. The procedure ensures that the final best point is already optimal when f is monotone in each variable (and the noise is sufficiently small). It also ensures a small amount of global search which occasionally avoids getting trapped in high-lying local minima or valleys (but sometimes it doesn't).

As starting vertex we use the better one of the vertices closest and farthest away from the starting point, to take care of the cases when x^0 is particularly good or bad.

Phase II (descent phase):

STEP 1. Build a quadratic surrogate function q around $x^\dagger$.

STEP 2. Minimize q over the current trust region, and denote by x an h-grid point closest to the minimizer obtained.

STEP 3. If f has already been evaluated at x, enter Phase III.

STEP 4. Increase N and s by 1, evaluate f at x and adapt the trust region. If $f^\sigma(x) < f^\sigma(x^*)$, set $x^* = x$, $s = 0$, and if also $f^\sigma(x) < f^\sigma(x^\dagger)$, set $x^\dagger = x^*$.

STEP 5. If $s \geq 3$ enter Phase III; otherwise goto Step 1.

Note that the details of Steps 1 and 2 are irrelevant for the global convergence analysis in Section 3; any surrogate function or trust region would do. The quadratic model considered is sufficient to guarantee in most cases fast local behavior (though we cannot prove this, since the least squares problem may be very ill-conditioned; enforcing good condition by adding further points turned out to be a bad strategy in practice).

In Step 1, a quadratic approximation of the underlying function f is built from the function values already known, and used for prediction of progress. (This idea is not new; see, e.g., Karidis & Turns [9], though they describe the details of their implementation in vague terms only, and our best interpretation of it turned out not to give a very robust algorithm.)

More specifically, let S denote the set of points where f has already been evaluated. For positive integers $k < N$ (specified in (6) and (7) below), let d_k be the k-th smallest number among the numbers $\|x - x^\dagger\|_\Omega$ $(x \in S)$, and define S_k to be the set of all evaluation points within the distance d_k, i.e.

$$S_k = \left\{ x \in S \mid \ \|x - x^\dagger\|_\Omega \le d_k \right\}. \tag{4}$$

The list of distances to $x^\dagger$ is stored, since it is used over and over again. The list is updated whenever a new point is evaluated or $x^\dagger$ changes. This involves at most $O(N_{\max}^2 n)$ operations, which, for the desired application range ($N_{\max} \le 200, n \le 12$), is small compared to other work.

Note that S_k may contain more than k points. For integers $k \ge N$ we define $S_k := S$. The function values at arguments from S_k, with k given in (6) and (7), will be used to determine a quadratic *surrogate function*

$$q(x) = a + g^t(x - x^\dagger) + \frac{1}{2}(x - x^\dagger)^t G (x - x^\dagger) \tag{5}$$

with $\binom{n+2}{2}$ free parameters as follows: First a Hessian approximation G is determined by minimizing

$$\chi_1^2 = \sum_{x \in S_{k_1}} \left[q(x) - f^\sigma(x) \right]^2, \quad k_1 = \binom{n+2}{2} + 2, \tag{6}$$

over all choices of a, g and $G = G^T$. (When there is no noise, one could keep $a = f(x^\dagger)$ fixed; but since our algorithm is mainly intended for noisy functions, we do not consider this variant.) Using the Hessian approximation G obtained, a local fit is then performed by minimizing

$$\chi_2^2 = \sum_{x \in S_{k_2}} \left[q(x) - f^\sigma(x) \right]^2, \quad k_2 = 2n + 2, \tag{7}$$

over all choices of a, g and κ, where

$$q(x) = a + g^t(x - x^\dagger) + \kappa \frac{1}{2}(x - x^\dagger)^t G (x - x^\dagger). \tag{8}$$

This yields a hopefully more accurate gradient g and a modified Hessian κG; note that in general neither G nor κG will be positive definite.

The least squares problems may be written as

$$\min_\theta \|\mathbf{X}\theta - y\|_2^2 \tag{9}$$

with suitable matrices $\mathbf{X}$, where θ denotes the parameter vector (a, g, G) or (a, g, κ) to be estimated and y the vector of function values at the $x \in S_k$.

To take care of the case when in (9) the number of parameters exceeds the number of data points for the least squares problem, we scale the least squares coefficient matrix to $X' = XD^{-1}$, where D is the diagonal matrix whose ith diagonal entry is the maximum norm of the ith column of $\mathbf{X}$. We then compute a solution minimizing $\|D\theta\|$ using the IMSL routine DL2VRR [8] for finding the minimum norm solution of least squares problems; this routine also handles the numerical rank determination problem via a singular value decomposition.

Since (6) contains $O(n^2)$ variables, the solution of (9) takes $O(n^6)$ operations; by updating a QR factorization of $\mathbf{X}$, this can be reduced to $O(n^4)$ for those subsequent iterations where S_k changes little and no rank deficiencies occur. For $n \leq 12$, computing time for the linear algebra remains in the seconds on a SUN SPARC station, and does not matter for the applications to expensive function minimization. (The methods of Box & Wilson [1], Dixon [3], and Glad & Goldstein [7] avoid expensive linear algebra by evaluating the function on well-chosen experimental designs for which the least squares problem can be solved more cheaply explicitly; but this requires an excessive number $O(n^2)$ of additional function evaluations).

For the surrogate function q obtained in this way, we then find the minimizer

$$\zeta = \arg \min_{x \in C} q(x), \tag{10}$$

over a trust region C consisting of all points $x \in \Omega$ satisfying

$$\|x^\dagger - x\|_\Omega \leq \rho, \tag{11}$$

and

$$\left|\frac{x_\nu - x_\nu^\dagger}{b_\nu - a_\nu}\right| \leq h \text{ for all } \nu \text{ with } \frac{\min(x_\nu^\dagger - a_\nu, b_\nu - x_\nu^\dagger)}{b_\nu - a_\nu} \leq h, \tag{12}$$

where ρ denotes a trust region parameter that is adapted during the optimization process (see (13)).

Finding (10) amounts to the solution of a (possibly indefinite) bound constrained quadratic program; however, in order to keep the work for this step at $O(n^3)$, one only calculates a stationary point satisfying the second order optimality conditions. (Actually, since we don't use an interior point method, the work may still be higher, but in the test runs it was significantly less than that for the least squares problem.) Note that (11) does not restrict C when $\rho = 1$, and that in case one bound is (nearly) active, condition (12) allows the corresponding component to be moved only by a small step. This gives a small improvement of performence in cases where many bounds are active.

Since we restrict evaluation of f to grid points, we compute the (lexicographically first) h-grid point x closest to ζ. (One could also consider a search for the optimal grid point; but this requires much extra programming and should be useful only locally when the quadratic model is already good, where the grid is quickly refined anyway.)

If f has already been evaluated at x we assume that we have no descent on the current grid, and Phase III is entered to check this assumption. Otherwise f is evaluated at x. In this case the trust region parameter ρ is adjusted by the following simple scheme:

$$\rho = \max(h, \min(\rho', 1)), \tag{13}$$

where

$$\rho' = \begin{cases} 2\rho & \text{if } f^\sigma(x) < f^\sigma(x^\dagger) \text{ and } \|\zeta - x^\dagger\|_\Omega > 0.5\rho, \\ \frac{1}{2}\|\zeta - x^\dagger\|_\Omega & \text{in case} \quad f^\sigma(x) > f_3, \\ \rho & \text{otherwise.} \end{cases} \tag{14}$$

Here f_3 denotes the third best function value gained so far. At the start of the algorithm ρ is set equal to one and $h = 1/10$. The adaptation strategy was chosen on the basis of simplicity, since the average behaviour of our method depends very little on the way the trust region is adapted, and convergence is not at all affected. More sophisticated strategies assessing the quality of the prediction (see Moré [10]) which are useful for the case when exact gradients are available, have no significant effect on the number of function evaluations needed: The inaccurate gradient obtained from the model fit reduces the effectiveness of step size adaption, and furthermore, the comparison of predicted versus achieved reduction (used in [10]) is corrupted by noise.

Whenever $s \geq 3$, there have been 3 function evaluations without improvement. We take this as a sign that progress slows down and Phase III is entered, too.

Phase III (refinement check phase):

Step 1. Add just as many points as are needed to get a numerically nonsingular linear model from h-grid points in the box within one grid step size around the best h-grid point x^*. The points are chosen in (supposed) downhill directions along the coordinate axes, unless bounds force the opposite orientation.

Step 2. If during Step 1) one of the obtained points yielded a better value than $f^\sigma(x^*)$, goto Step 7.

Step 3. Minimize the linear model within one grid step size and denote by y the corresponding solution.

STEP 4. If y is a new evaluation point and $f^\sigma(y)$ is less than $f^\sigma(x^*)$, goto Step 7.

STEP 5. If during Phase III new evaluation points have been gained, construct a quadratic surrogate function according to Phase II and denote by x the grid point closest to the minimizer of this surrogate function within a trust region around $x^\dagger$.

STEP 6. If x is a new evaluation point and $f^\sigma(x)$ is less than $f^\sigma(x^*)$: Set x^* to the point with best grid function value; if $f^\sigma(x^*) < f^\sigma(x^\dagger)$, set $x^\dagger = x^*$; update N and goto Step 1 of Phase II.

STEP 7. If $g = g_{max}$, stop. Otherwise set $h = h/10$ and $g = g + 1$, update N and goto Step 1 of Phase II.

Phase III checks whether one can expect further progress with the current grid spacing or whether the grid needs to be refined. The details are chosen with the intention to ensure the convergence of the algorithm in the noiseless case (see Section 3). If the linear (and possibly quadratic) model predictors do not lead to a better grid point it is concluded that no further progress is likely on the current grid, and the grid spacing is reduced. We used the refinement factor 10 in order that a refinement of the grid may improve one further digit in the current approximation to the solution. Moreover, this choice is most suitable for the optimization of experiments since it produces evaluation points whose coordinates have no more digits than needed at the current level of accuracy.

For Step 1, the details are as follows. Denote by $S^{'}$ the set of all h-grid evaluation points, and enumerate the points in

$$M = \left\{ x \in S^{'} \mid \|x^* - x\|_\Omega \leq h \right\} \tag{15}$$

as $x^1, \ldots, x^m$. Then the linear function

$$l(x) := f + g^t(x - x^*) \tag{16}$$

satisfies $l(x^i) = (\mathbf{X}\theta)_i$, where $\theta^t = (f, g^t)$ and, for $i = 1, \ldots, m$,

$$\mathbf{X}_{ij} = \begin{cases} 1 & \text{if } j = 1, \\ (x^i - x^*)_{j-1} & \text{if } j = 2, \ldots, n+1. \end{cases}$$

We check the rank of $\mathbf{X}$ as in Phase II by scaling $\mathbf{X}$ and using DL2VRR. (Since $\mathbf{X}$ is integral up to a scaling factor, one could determine the determinant of $\mathbf{X}^T\mathbf{X}$ exactly and decide in this way on the rank. However, when using floating point arithmetic, a numerical rank is more appropriate.)

In case when, numerically, $\text{rk}(\mathbf{X}) < n + 1$, we check, for each component $\nu = 1, \ldots, n$ in turn, whether the numerical rank of $\mathbf{X}$ increases through the addition of the point

$$y = x^* \pm h(b_\nu - a_\nu)e^\nu, \tag{17}$$

where e^ν is the ν-th column of the unit matrix and the sign in (17) is determined as follows: In case when x^*_ν attains an upper/lower bound, an h-step in component ν away from this bound is taken. Otherwise the most recent quadratic model built within Phase II is used to estimate the gradient at x^* and the step in component ν is taken with the opposite of the sign of this ν-th gradient component.

Since x^* and the vectors (17) span the full affine space in a numerically stable fashion, the numerical rank of $\mathbf{X}$ becomes $n+1$ after adding to M each of the points (17) which lead to further rank increase. The free parameters in the linear model (16) can now be determined by least squares fitting to the function values with arguments in M.

To ensure that the minimizer

$$y = \arg\min \{l(x) \mid x \in \Omega, \|x - x^*\|_\Omega \le h\} \tag{18}$$

of the linear model lies on the grid, we set y_ν to x^*_ν in the ambiguous case where $g_\nu = 0$.

Stopping rules: The algorithm is terminated before the grid spacing h is reduced more than g_{max} times. For the numerical calculations in Section 4 we used $g_{max} = 12$; we also stopped whenever $N_{\max} = 200$ function evaluations had already been performed. This implies a minimal number of $12n$ function evaluations (since h cannot decrease before at least n function evaluations have been done on the current level), even when the optimum is found very early (which often happens if it lies in a corner of the initial box). However, in practice, function evaluation is often very slow or by hand input of measurement data; then g_{max} is taken much smaller, and in fact, stopping is usually done interactively.

3 Convergence analysis

In this section we consider the behavior of the grid algorithm described in Section 2, but without stopping rules, i.e., for $g_{max} = \infty$. We shall show convergence to a stationary point in the noiseless case. Unlike trust region methods using exact gradients, where convergence follows from a careful adaption of the trust region radius, our method derives its convergence properties mainly from Phase III and the restriction (in Phase II) of function evaluations to h-grid points (cf. Torczon [19]). Indeed, we shall show that (using Step 1 of Phase III) the coefficients of the linear model in Phase III give asymptotically accurate estimates of the function value and gradient (within the accuracy which can be expected in the presence of noise) and Steps 3 and 4 of Phase III therefore give sufficient descent to force convergence. Note that most details in Phase I and II do not affect convergence, but are important for fast progress in typical problems, while Phase III is responsible for the good robustness properties.

In the following we assume that f is twice continuously differentiable in Ω. We also assume that the inaccuracy of function evaluation is globally bounded by a threshold Δ, so that

$$|f^\sigma(x) - f(x)| \leq \Delta \quad \text{for all } x \in \Omega. \tag{19}$$

$g(x)$ denotes the gradient of f at x.

By rescaling the problem we may assume w.l.o.g. that

$$\Omega = [0,1] \times \ldots \times [0,1],$$

so that $\|x\|_\Omega = \|x\|_\infty$ and the dual norm

$$\|g\|_\Omega^* = \sum_\nu (b_\nu - a_\nu)|g_\nu|$$

becomes simply $\|g\|_1$. As a measure for the amount of nonlinearity in f we use

$$\gamma = \max_{x\in\Omega} \sum_{i,j} |f''_{ij}(x)|. \tag{20}$$

Note that (20) implies

$$|f(x) + g(x)^t s - f(x+s)| \leq \frac{\gamma}{2}h^2 \quad \text{if } x, x+s \in \Omega \text{ and } \|s\|_\infty \leq h. \tag{21}$$

Proposition 1: Let $x^1, \ldots, x^m$ be m distinct h-grid points in the neighbourhood of an h-grid point x^*, i.e. $\|x^i - x^*\|_\Omega = h$, for $i = 1, \ldots, m$, and suppose that the $(m+1) \times (n+1)$ design matrix

$$\mathbf{X} = (e, h\mathbf{E}) \tag{22}$$

with $e_0 = e_i = 1$, $\mathbf{E}_{0j} = 0$, and $h\mathbf{E}_{ij} = x^i_j - x^*_j$ ($i = 1, \ldots, m$, $j = 1, \ldots, n$), has rank $n+1$. Then the least squares fit

$$\min_{f,g} \sum_{i=0}^{m} \left[f^\sigma(x^i) - f - g^t\left(x^i - x^*\right)\right]^2 \tag{23}$$

yields estimates $\hat{f}, \hat{g}$ satisfying

$$|\ \hat{f} - f(x^*)\ | \leq C_0 \epsilon \tag{24}$$

and

$$\|\hat{g} - g(x^*)\|_\Omega^* \leq C_0 \epsilon / h \tag{25}$$

for $\epsilon = \Delta + \frac{\gamma}{2}h^2$ with a suitable constant C_0 depending only on the relative grid spacings.

Proof: By (19) and (21), the vector $b = (f^\sigma(x^*) \dots, f^\sigma(x^m))^t$ satisfies

$$b = f(x^*)e + h\mathbf{E}g(x^*) + \delta, \tag{26}$$

with

$$\|\delta\|_\infty \leq \Delta + \frac{\gamma}{2}h^2 = \epsilon. \tag{27}$$

The normal equations for the least squares fit of (23) read

$$(e, h\mathbf{E})^t \left[(e, h\mathbf{E})(\hat{f}, \hat{g}^t)^t - b\right] = 0, \tag{28}$$

and hence by (26)

$$(e, h\mathbf{E})^t \left[e\mathit{df} + h\mathbf{E}\mathit{dg} - \delta\right] = 0, \tag{29}$$

with $\mathit{df} = \hat{f} - f(x^*)$, $\mathit{dg} = \hat{g} - g(x^*)$. Thus

$$\mathit{df}\, e^t e + h e^t \mathbf{E}\mathit{dg} = e^t\delta \tag{30}$$

and

$$h\mathit{df}\,\mathbf{E}^t e + h^2\mathbf{E}^t\mathbf{E}\mathit{dg} = h\mathbf{E}^t\delta \tag{31}$$

hold. Since the rank assumption implies that $\mathbf{E}$ has rank n, $\mathbf{E}^t\mathbf{E}$ is nonsingular, and we conclude that

$$\mathit{dg} = \frac{1}{h}(\mathbf{E}^t\mathbf{E})^{-1}\left[\mathbf{E}^t\delta - \mathit{df}\,\mathbf{E}^t e\right]. \tag{32}$$

By inserting this into (30) we find after some simplification that

$$\alpha\mathit{df} = \left(e^t - e^t\mathbf{E}(\mathbf{E}^t\mathbf{E})^{-1}\mathbf{E}^t\right)\delta, \tag{33}$$

where

$$\alpha = e^t e - e^t\mathbf{E}\left(\mathbf{E}^t\mathbf{E}\right)^{-1}\mathbf{E}^t e. \tag{34}$$

Now since $\mathbf{X}$ has rank $n+1$, *df* and *dg* must be uniquely determined, and it follows that α never vanishes. Moreover, because only a finite number of possible different matrices $\mathbf{E}$ exist (independent of the grid size h) we conclude that

$$\mid \mathit{df} \mid \leq C\epsilon \tag{35}$$

for some suitable constant C independent of the x^i. (This type of argument is also used in the convergence proof of so-called *pattern search methods* in Torczon [20].) Insertion into (32) shows that

$$\|\mathit{dg}\|^*_\Omega = \|\mathit{dg}\|_1 \leq \frac{1}{h}\|(\mathbf{E}^t\mathbf{E})^{-1}\|_1\|\mathbf{E}^t\delta - \mathit{df}\,\mathbf{E}^t e\|_1 \leq C'\epsilon/h \tag{36}$$

holds for some suitable constant C' independent of the x^i. Then (24) and (25) hold with $C_0 = \max(C, C')$. □

We shall use the well-known first order optimality conditions for bound constrained minima in the following form:

At any local minimizer x of $f(x)$ $(x \in \Omega)$, the *reduced gradient* $g^{red}(x)$, defined by

$$g^{red}_\nu(x) = \begin{cases} \max\,(g_\nu(x), 0) & \text{if} \quad x_\nu = b_\nu, \\ \min\,(g_\nu(x), 0) & \text{if} \quad x_\nu = a_\nu, \\ g_\nu(x) & \text{otherwise} \end{cases} \tag{37}$$

$(\nu = 1, \dots, n)$, vanishes.

Proposition 2: Assume $\Delta = 0$, and let $\omega > 0$ be fixed. Denote by $\hat{f}(x) = \hat{f} + \hat{g}^t(x - x^*)$ the prediction of the linear model as obtained in Phase III around the best grid point x^*. Let $y = x^* + s$ be the corresponding minimizer within one h-grid spacing. Then

$$\|g^{red}(x^*)\|_\infty > \omega \;\Rightarrow\; \hat{f}(x^*) - \hat{f}(y) > \frac{\omega}{2}h \tag{38}$$

for sufficiently small h, independent of x^*.

Proof: By assumption, there is an index α such that $|g_\alpha(x^*)| = |g^{red}_\alpha(x^*)| > \omega > 0$. W.l.o.g. (replace, if necessary, x_α by $1 - x_\alpha$ in the problem formulation) we may assume $g_\alpha(x^*) = g^{red}_\alpha(x^*) > \omega > 0$. Because of the discrete grid, (37) then implies $0x^*_\alpha \geq h$. For all $h < \omega/C_0\gamma$ we obtain from Proposition 1 the inequality

$$\hat{g}_\alpha - g_\alpha(x^*) \;\; \leq C_0\frac{\gamma}{2}h < \frac{\omega}{2} \tag{39}$$

independent of x^* and hence

$$\hat{g}_\alpha > \frac{\omega}{2}. \tag{40}$$

Thus we conclude $s_\alpha = -h$ and

$$\hat{f}(x^*) - \hat{f}(y) = -\hat{g}^t(y - x^*) > \frac{\omega}{2}h - \sum_{\nu \neq \alpha} \hat{g}_\nu s_\nu \geq \frac{\omega}{2}h, \tag{41}$$

since y is the minimizer of $\hat{f}$. □

Proposition 3: In the course of the algorithm the grid will be arbitrarily tightly refined, i.e. as $N \to \infty$, the grid spacing h tends to zero.

Proof: For any given grid size h only a finite number of different h-grid points exist. Since all (but the first) evaluation points lie on the current grid, only a finite number of function evaluations may be taken until a refinement of the grid spacing h is forced in Phase III. (The power of this finiteness argument in convergence proofs was recognized by Torczon [18, 19].) □

Theorem 1: Assume $\Delta = 0$ and let z^k denote the best grid points obtained up to the k-th grid refinement. Then the reduced gradients $g^{red}(z^k)$ converge to zero, i.e. $\lim_{k\to\infty} g^{red}(z^k) = 0$.

Proof: By Proposition 3, z^k is defined for all $k > 0$. Denote by $\hat{f}(x)$ the linear model obtained in Phase III and by $z^k + s$ the minimizer of this model within one grid step size h_k. Note that the assumptions of Proposition 1 hold with $x^* = z^k$ for the linear model built in Phase III. Note further that

$$f(z^k + s) \geq f(z^k) \tag{42}$$

since Step 6 of Phase III reduced h. Now suppose the assertion of the theorem fails to hold. Then

$$\omega = \frac{1}{2} \limsup \|g^{red}(z^k)\|_\infty > 0. \tag{43}$$

Now the grid step size before the k-th grid refinement satisfies $h_k = 0.1^k h_0 \to 0$ for $k \to \infty$. Hence Proposition 2 implies that for infinitely many k,

$$\hat{f}(z^k) - \hat{f}(z^k + s) > \frac{\omega}{2} h_k. \tag{44}$$

Since $\Delta = 0, \epsilon = \frac{\gamma}{2} h_k^2$, so using (21), (24) and (25) we obtain

$$|\hat{f}(z^k + s) - f(z^k + s)| = |(\hat{g} - g(z^k))^t s| + |\hat{f}(z^k) - f(z^k)| + O(h_k^2) = O(h_k^2).$$

Using this, (42), (44) and again (24) we find

$$\begin{aligned} 0 &\geq f(z^k) - f(z^k + s) \\ &\geq \hat{f}(z^k) - \hat{f}(z^k + s) - |f(z^k) - \hat{f}(z^k)| - |f(z^k + s) - \hat{f}(z^k + s)| \\ &> \frac{\omega}{2} h_k + O(h_k^2) \end{aligned}$$

for infinitely many k, which contradicts $h_k \to 0$. Thus the assertion must hold. □

Theorem 1 ensures in the noiseless case that the sequence of best grid points converges to a stationary point as the number of function evaluations tends to infinity. For the general case, the following theorem, inspired by Glad & Goldstein [7], yields an upper bound for the gradients up to the k-th refinement step.

Theorem 2: Let z^k and h_k denote the best grid point and the grid size, respectively, obtained up to the k-th grid refinement. Then there is a constant C depending only on the relative grid spacings, such that

$$\|g^{red}(z^k)\|_\Omega^* \leq C(\gamma h_k + \frac{2\Delta}{h_k}). \tag{45}$$

Proof: For simplicity we assume that no bound is active; the proof applies with trivial modifications in the general case. We write $g = g_{red}(z^k) = g(z^k)$ and denote again by $y = z^k + s$ the minimizer of the linear model built in Phase III within one grid step size $h = h_k$. Then we have

$$\begin{aligned} f(z^k) - \epsilon &\leq f(z^k) - \Delta \leq f^\sigma(z^k) \leq f^\sigma(y) \leq f(y) + \Delta \\ &\leq f(z^k) + g(z^k)^t s + \frac{\gamma}{2}h^2 + \Delta \\ &= f(z^k) + g^t s + \epsilon, \end{aligned} \tag{46}$$

and we conclude

$$-\frac{g^t s}{h} \leq 2\epsilon/h. \tag{47}$$

With the index sets $I = \{i \mid \hat{g}_i g_i > 0\}$ and $K = \{i \mid \hat{g}_i g_i < 0\}$, we have

$$-\frac{g^t s}{h} = \sum_{i\in I} \mid g_i \mid - \sum_{i \in K} \mid g_i \mid, \tag{48}$$

and by Proposition 1 we conclude that

$$\sum_{i\in K} \mid g_i \mid \leq \sum_{i\in K} \mid \hat{g}_i - g_i \mid \leq \|\hat{g} - g\|_1 \leq \frac{C_0 \epsilon}{h}. \tag{49}$$

Hence

$$\|g\|_1 = \sum_{i\in I} \mid g_i \mid + \sum_{i\in K} \mid g_i \mid \leq -\frac{g^t s}{h} + \frac{2C_0\epsilon}{h}, \tag{50}$$

and since $\epsilon = \Delta + \frac{\gamma}{2}h^2$, (47) yields

$$\|g\|_1 \leq (1 + C_0)2\epsilon/h = (2C_0 + 2)(\gamma h + \frac{2\Delta}{h}). \tag{51}$$

□

Corollary. Under the assumptions of Theorem 2,

$$\|g^{red}(x)\| = O(\Delta/h_0 + \sqrt{\gamma\Delta}) \tag{52}$$

holds for at least one $x = x_k$.

Proof: The right hand side of (45) is a convex function of $h = h_k$ with minimum at $h_{min} = \sqrt{2\Delta/\gamma}$. If $h_0 < h_{min}$ then $\gamma h_0^2 < \gamma h_{min}^2 \leq 2\Delta$, and for $k = 1$, the bound in Theorem 2 is of order $O(\Delta/h_0)$. If $h_0 \geq h_{min}$, the algorithm uses at some stage a grid size $h = \kappa h_{min}$ with $0.1 \leq \kappa \leq 1$, and the bound is of order $O(\gamma h_{min}) + O(\Delta/h_{min}) = O(\sqrt{\gamma\Delta})$. □

As one can easily see, $\|g^{red}(x)\| = O(\sqrt{\gamma\Delta})$ is the best order of magnitude which can be achieved at a given noise level Δ and curvature bound γ. The algorithm achieves this magnitude unless the initial grid size h_0 (i.e., the box size) is so small that the second term in (45) (due to the noise) dominates the bound. This happens precisely when the curvature contribution to the funcion value (cf. (21)) is masked by the noise, $\frac{\gamma}{2}h_0^2 \ll \Delta$. In this case we can shrink the reduced gradient only to $\|g^{red}(x)\| = O(\Delta/h_0)$. However, since the function can hardly be distinguished from a linear function, the minimum in such a situation is attained at a bound for all components which contribute to the function value beyond noise level. Thus our algorithm behaves adequately even in this situation.

4 Numerical results

Numerical tests of the above proposed algorithm were performed using the 18 functions from a standard test set as described in Moré et al. [11]. The dimensions and bounds (see Table I) were chosen as given in Gay [5].

Table I: Bounds specifying Ω for the test problems (from Gay [5]; exponents indicate repetition)

problem	dim	lower bounds	upper bounds
7	3	$-100, -1, -1$	$.8, 1, 1$
18	6	$0^3, 1, 0, 0$	$2, 8, 1, 7, 5, 5$
9	3	$.398, 1, -.5$	$4.2, 2, .1$
3	2	$0, 1$	$1, 9$
12	3	$0, 5, 0$	$2, 9.5, 20$
25	10	0^{10}	$10, 20, 30, 40, 50, 60, 70, 80, 90, .5$
20	9	$-.00001, 0^4, -3, 0, -3, 0$	$.00001, .9, .1, 1, 1, 0, 4, 0, 2$
20	12	$-1, 0, -1^3, 0, -3, 0, -10, 0, -5, 0$	$0, .9, 0, .3, 0, 1, 0, 10, 0, 10, 0, 1$
23	10	$0, 1, 0^3, 1, 0^3, 1$	100^{10}
24	4	$-10, .3, 0, -1$	$50^3, .5$
24	10	$-10, .1, 0, .05, 0, -10, 0, .2, 0, 0$	$50^9, .5$
4	2	$0, .00003$	$1000000, 100$
16	4	$-10, 0, -100, -20$	$100, 15, 0, .2$
11	3	0^3	10^3
26	10	$0, 10, 20, 30, 40, 50, 60, 70, 80, 90$	$10, 20, 30, 40, 50, 60, 70, 80, 90, 100$
21	2	$-50, 0$	$.5, 100$
22	4	$.1, -20, -1, -1$	$100, 20, 1, 50$
5	2	$.6, .5$	$10, 100$
14	4	-100^4	$0, 10, 100, 100$
35	7	0^7	$.05, .23, .333, 1^4$
35	8	$0, 0, .1, 0^5$	$.04, .2, .3, 1^5$
35	9	$0, 0, .1, 0^6$	$1, .2, .23, .4, 1^5$
35	10	$0, .1, .2, 0, 0, .5^5$	$1, .2, .3, .4, .4, 1^5$

Note that in many cases they exclude the global minimum of these function (usually $f = 0$), causing the optimum to lie on the boundary or even at a vertex of the box. (This is to be expected in many practical situations, too.) As starting values, the standard starting points defined in [11] (and their multiples by 10 and 100) were projected on the given box Ω. Since on some problems the projections of the multiples were identical, a total set of only 46 test problems resulted.

The function values were obtained as

$$f^\sigma(x) = f(x)\,(1+\eta)\,, \quad \eta \sim N(0,\sigma^2), \tag{53}$$

where $N(0,\sigma^2)$ denotes a Gaussian distributed random number with zero mean and variance σ^2, i.e. relative stochastic errors were used for the test problems. This reflects the fact that for problems where function values do not differ much in magnitude, relative and absolute errors behave nearly the same, but for problems where, in the feasible region, function values are positive and range over several orders of magnitude (as in most of our test examples), relative errors are more appropriate.

Two of our functions allow their values to approach zero; the results for these functions show to which extent a method can recover from regions of high noise to find with high accuracy the well-defined region where f is tiny. In particular, though we do not address this possibility directly, this situation models the possibility that – as in many finite element applications – the user can adjust the accuracy of the function evaluation to the progress of the optimization.

Our new grid algorithm (GR) was compared to both a quasi-Newton approach using finite difference gradients (QN) and the simplex method of Nelder and Mead (NM). For QN, the implementation in the NAG library (Mark 14) [12] was used, which allows for bound constraints. For NM, we modified the Nelder-Mead method to take care of bound constraints. (Simply adjusting the unconstrained NAG implementation by defining $f(x) = \infty$ in case $x \notin \Omega$ does not give adequate results.) In case that the unconstrained method requests an evaluation at $x \notin \Omega$, this point is replaced by

$$x' = \tilde{x} + 0.1(x^* - \tilde{x}),$$

where x^* is the best point found so far and $\tilde{x}$ is the projection of x to the box,

$$\tilde{x}_\alpha = \max(a_\alpha, \min(x_\alpha, b_\alpha)) \;\text{ for } \alpha = 1, \ldots, n.$$

(In case of a minimizer in a corner, the factor 0.1 gives linear convergence at the same rate as GR.) If during an iteration the first reflection step has touched at least one bound no expansion step is performed during that iteration. We stop the iteration after 6 consecutive unsuccessful contractions, or after $3n + 20$ iterations without improving the best function value. The other details were implemented as stated in [6], Section 4.2.2, with the improved contraction step 3'; an expansion factor $\beta = 2$, a contraction factor $\gamma = \frac{1}{2}$ and a reflection factor $\alpha = 1$ were chosen.

The starting simplex was chosen by the following rule: x^0 was taken as the given starting point and the vertices $x^i (i = 1, \dots, n)$ were chosen to get a large rectangular simplex,

$$x^i_\alpha = \begin{cases} x^0_\alpha & \text{in case } \alpha \neq i, \\ a_\alpha & \text{in case } \alpha = i \text{ and } x^0_i - a_i > b_i - x^0_i, \\ b_\alpha & \text{otherwise .} \end{cases}$$

The implementation of GR uses the IMSL [8] routine DL2VRR (singular value decomposition) for the solution of the least squares problem, and the NAG [12] routine E04NAF for solving the quadratic programming problem of minimizing the quadratic surrogate function. The starting value was the best point found. (We also tried a limited form of multiple start to enhance the chance of finding a better global optimizer in the indefinite case, but the overall results differed only insignificantly.)

As generally known and confirmed in all examples by our experiments (with $\sigma = 0.01$), in the noisy case a quasi-Newton approach using the standard finite difference approximation for the gradient breaks down. Hence we only report the following tests:

1. In case $\sigma = 0$, i.e. in the noiseless case, the grid algorithm (GR) was compared to both QN and NM.

2. NM and GR were compared for noisy functions with standard deviation $\sigma \in \{0.01, 0.05, 0.10\}$.

As a measure of the speed of convergence we introduce the quotients

$$q_i = \frac{f_{target,i} - f_{target}}{f_0 - f_{target}}, \tag{54}$$

with f_0 the value of f at x^0 (and *not* f^σ), $f_{target,i} = \min_{j \le i} f_j$ (and *not* f^σ_j), and f_{target} a target value achieved at some point within the bounds. These target values f_{target} (see Table II) were in each case taken as the best function value found over a sequence of many noise-free local optimizations, using both QN and NM, and hence are likely to be global optima (within the region defined by the bounds). Also stated in table II is the number of variables that achieve their upper or lower bounds at the optimum. The numbers q_i (which are of course not available in real applications) describe the speed of approaching the minimum of the smooth uncorrupted function f.

Table II: Target function values of test problems. The last column gives the number of bounds active at the target point.

problem	dimension	f_{target}	active bounds
7	3	$0.99042212 \cdot 10^{-0}$	1
18	6	$0.53209865 \cdot 10^{-3}$	2
9	3	$0.11279300 \cdot 10^{-7}$	1
3	2	$0.15125900 \cdot 10^{-9}$	1
12	3	$0.30998153 \cdot 10^{-5}$	1
25	10	$0.33741268 \cdot 10^{-0}$	1
20	9	$0.37401397 \cdot 10^{-1}$	5
20	12	$0.71642800 \cdot 10^{-1}$	7
23	10	$0.75625699 \cdot 10^{+1}$	10
24	4	$0.94343600 \cdot 10^{-5}$	1
24	10	$0.29442600 \cdot 10^{-3}$	0
4	2	$0.78400000 \cdot 10^{+3}$	2
16	4	$0.88860479 \cdot 10^{+5}$	2
11	3	$0.58281431 \cdot 10^{-4}$	2
26	10	$0.00000000 \cdot 10^{-0}$	0
21	2	$0.25000000 \cdot 10^{-0}$	1
22	4	$0.18781963 \cdot 10^{-3}$	1
5	2	$0.00000000 \cdot 10^{-0}$	1
14	4	$0.15567008 \cdot 10^{+1}$	1
35	7	$0.98323258 \cdot 10^{-3}$	3
35	8	$0.36399851 \cdot 10^{-2}$	1
35	9	$0.10941440 \cdot 10^{-4}$	2
35	10	$0.65039548 \cdot 10^{-2}$	0

For a compact summary of our results, the following characteristics were picked out:

- $Nfail_i, i = 1, 2, 6$: number of failures to achieve a relative function reduction of $10^{-1}, 10^{-2}$ and 10^{-6} in $f_0 - f_{target}$, respectively.

  This criterion is addressed to measure the robustness. Failures were almost always due to reaching $N_{\max} = 200$ function evaluations, with the best point being near (but not near enough) to the point giving rise to the target value. Occasionally, failure was due to not reaching the target value because of being stuck near a local minimum. However, this only happened rarely, and did not favor a particular method (see Remark 1 below). We adopted this measure for simplicity and because of its relevance for practical applications.

- $Nf_i, i = 1, 2, 6$: mean number of function evaluations needed to achieve a reduction in $f - f_{target}$ by a factor of $10^{-1}, 10^{-2}$ and 10^{-6}, respectively. Note that we compare the uncorrupted function values; hence in some cases a high reduction is obtained in spite of much noise (especially when the target value is zero or attained in a vertex of the feasible domain). Cases where a prescribed reduction could not be achieved were counted with $N_{\max} = 200$. This criterion measures the mean speed of descent.

- Finally the number of cases were counted where one algorithm performed better or at least as well as the other one with respect to *all* characteristics $q_{50}, q_{100}, q_{150}, q_{200}$ and Nf_1, Nf_2, Nf_6.

Table III: Summary statistics on the test set

σ	$Nfail_1$	$Nfail_2$	$Nfail_6$	Nf_1	Nf_2	Nf_6	better
	QN/GR	QN/GR	QN/GR	QN/GR	QN/GR	QN/GR	QN/GR
0.00	8/2	10/5	19/15	56/25	76/45	124/94	5/22
	NM/GR	NM/GR	NM/GR	NM/GR	NM/GR	NM/GR	NM/GR
0.00	6/2	11/5	23/15	53/25	77/45	145/94	3/30
0.01	8/4	17/8	33/24	60/29	87/50	161/117	4/31
0.01	10/3	15/9	31/22	62/29	83/53	159/115	4/30
0.01	11/4	16/6	34/26	67/31	86/48	167/124	6/32
0.05	13/5	17/10	35/26	70/33	89/56	168/124	5/30
0.05	13/5	17/12	37/26	72/33	91/63	173/124	4/28
0.05	13/6	19/10	40/26	73/35	95/56	180/124	2/32
0.10	14/6	18/12	38/28	75/37	91/63	177/131	7/29
0.10	11/6	16/12	38/26	69/37	85/62	176/122	4/30
0.10	13/7	19/12	40/28	72/40	95/63	183/132	7/27

Table III shows the results comparing QN versus GR and NM versus GR. The noise level σ chosen appears in the first column; since the noise in (53) is stochastic, the test program was run several times for each value of $\sigma > 0$ with different choices of the stochastic noise.

One can see that GR performs significantly better with respect to the above characteristics and is essentially more robust both in the noiseless case and in the noisy case. Moreover, the overall results are fairly independent of the particular realization of the simulated noise. The results for GR are nearly independent of the amount of noise, a feature mainly due to the grid technique.

To some extent the good performance of GR is due to the starting phase (see Table V below). In a few cases, (problems 23 and 4), Phase I finds the optimal vertex very quickly (it was designed to do so for monotone functions, and some functions behave on a global scale roughly monotonic). In these cases, the rest of the time is spent in the other phases to contract the grid towards this vertex. This is necessary since there is no other way to get some assurance that we are at the minimum. (We could contract with a smaller factor, but this decreases the robustness in the general case).

To check the influence of phase I, we started QN with the final point of Phase I of our algorithm (QN1). The results are given in Table IV. We see that in the noiseless case, QN1 is an improvement over QN comparable to GR. Hence we conclude that *for problems without noise, quasi-Newton methods for bound constrained optimization are as fast and robust as the new method, provided they are started with Phase I.*

We also tried to start NM with the simplex formed by the best point in Step 1) of Phase I and the n points defined in Step 3) of Phase I, while replacing the vertex x^1 closest to x^0 by x^0 if the latter had a smaller function value. Except in the cases where phase I already produced the optimum, this turned out to be only a slight improvement over the straightforward start, and was still far from being competitive with GR.

Table IV: QN with Phase I

σ	$Nfail_1$	$Nfail_2$	$Nfail_6$	Nf_1	Nf_2	Nf_6	better
	QN1/GR	QN1/GR	QN1/GR	QN1/GR	QN1/GR	QN1/GR	QN1/GR
0.00	2/2	7/5	16/15	27/25	51/45	95/94	10/12

Remark 1. In some cases, any of the algorithms may get stuck in a local minimum. This shows in our tests as a termination with less than 200 function evaluations while a relative reduction of 10^{-6} was not achieved. However, this happened (with $\sigma = 0$) for QN only 3 times, for QN1 4 times, for NM in 3 cases, and for GR in 2 cases, respectively.

Since the summary statistics does not reveal the behavior on individual functions, we give in Table V more detailed results for a particular run through the test set (with standard starting points only and $\sigma = 0.01$).

The first three columns of table V specify the test function (number in [11], name and dimension). Column 4 contains the algorithm used. The fourth column, titled q_{PhaseI}, displays the quotients q_i at the end of Phase I, and the next column, titled q_{200}, displays the quotients q_{200} (or the quotient q_i reached at termination with less than 200 function values). The final three columns display the number of function evaluations N_i necessary for reducing q_i to less than 10^{-1}, 10^{-2} and 10^{-6} respectively. A dash indicates that the corresponding reduction was not achieved.

Table V:

No.	problem	n	method	q_{PhaseI}	q_{200}	N_1	N_2	N_6
7	Helical valley	3	NM	–	0.66E-05	7	10	–
7		3	GR	0.10E-01	0.12E-02	3	7	–
18	Biggs EXP6	6	NM	–	0.44E-01	118	–	–
18		6	GR	1	0.14E-04	30	61	–
9	Gaussian	3	NM	–	0.12E-02	50	61	–
9		3	GR	1	0.13E-04	33	50	–
3	Powell badly scaled	2	NM	–	0.74E-08	14	27	53
3		2	GR	0.88	0.42E-13	15	15	21
12	Box three-dim.	3	NM	–	0.24E-04	4	4	–
12		3	GR	0.56E-03	0.75E-07	2	2	37
25	Variably dim.	10	NM	–	0.50E-05	3	3	–
25		10	GR	0.20E-03	0.18E-03	5	5	–
20	Watson	9	NM	–	0.93E-03	25	33	–
20		9	GR	0.12	0.42E-03	14	69	–
20		12	NM	–	0.10E-02	40	66	–
20		12	GR	0.59E-01	0.49E-02	5	55	–
23	Penalty I	10	NM	–	0.64E-04	30	49	–
23		10	GR	0.68E-12	0.68E-12	3	3	3
24	Penalty II	4	NM	–	0.36E-06	22	27	124
24		4	GR	1	0.27E-06	25	31	71
24		10	NM	–	0.13E-05	63	69	–
24		10	GR	1	0.12E-03	56	69	–
4	Brown badly scaled	2	NM	–	0.78E-09	18	23	31
4		2	GR	0.78E-09	0.78E-09	4	4	4
16	Brown and Dennis	4	NM	–	0.40E-03	8	15	–
16		4	GR	0.35E-01	0.31E-04	5	8	–
11	Gulf research dev.	3	NM	–	0.99E-03	2	2	–
11		3	GR	0.99E-03	0.99E-03	2	2	–
26	Trigonometric	10	NM	–	0.13E-01	56	–	–
26		10	GR	0.19	0.13E-01	22	–	–
21	Rosenbrock	2	NM	–	0.16	–	–	–
21		2	GR	0.26	0.42E-05	10	27	–
22	Extended Powell singular	4	NM	–	0.38E-01	54	–	–
22		4	GR	1	0.18E-03	23	46	–
5	Beale	2	NM	–	0.21E-01	31	–	–
5		2	GR	0.64	0.28E-24	7	11	19
14	Wood	4	NM	–	0.19	–	–	–
14		4	GR	1	0.18E-02	10	10	–

Table V (ctd.):

No.	problem	n	method	q_{PhaseI}	q_{200}	N_1	N_2	N_6
35	Chebyquad	7	NM	–	0.26E-01	164	–	–
35		7	GR	1	0.12E-01	50	–	–
35	Chebyquad	8	NM	–	0.26	–	–	–
35		8	GR	1	0.11	–	–	–
35	Chebyquad	9	NM	–	0.63	–	–	–
35		9	GR	1	0.10E-01	63	–	–
35	Chebyquad	10	NM	–	0.24	–	–	–
35		10	GR	1	0.27	–	–	–

5 Conclusions

A new approach for the task of solving bound constrained problems using function values only was offered. The proposed algorithm is particularly suitable for situations where the obtained function values are corrupted by (deterministic or stochastic) noise but the underlying function is smooth.

A convergence result was proved for noiseless functions, and the limit accuracy was analyzed in the noisy case. Extensive numerical calculations were performed on a standard collection of test problems. The results were compared to those obtained by two well-known, widely used methods, that of Nelder and Mead (NM) and a quasi-Newton method (QN), the latter only without noise. The results show that in the noiseless case the new method is very reliable and fast in terms of total number of function evaluations, and comparable in speed and robustness with the improved version of QN started with the preprocessing Phase I. In the presence of noise the method proved to be significantly more efficient than our best variants of NM.

Since, at least in the present implementation, the time for intermediate calculations significantly exceeds that used by NM or QN, the conclusion is that GR should be applied in situations where function evaluation costs are the dominating factor, as for instance in the optimization of functions whose values are determined by real life experiments.

The amount of linear algebra work can be reduced to $O(n^4)$ by using rank one updates for the least squares problems. Quasi-Newton techniques might reduce this further to $O(n^2)$, but it is difficult to devise a strategy which gives locally accurate quadratic models without extra function evaluations. Research on this is in progress.

It is also worthwhile to note that the convergence analysis given only depends on very few (and the least technical) properties of the algorithm. This leaves some scope for future improvements in Phase II, which is essential for the obtainable speed.

References

[1] Box, G.E.P. and Wilson, K.B., On the experimental attainment of optimum conditions, *J. Royal Stat. Soc., Ser. B*, **13** (1951), 1-45.

[2] Byrd, R.H., Dert, C.L., Rinnooy Kan, A.H.G. and Schnabel, R.B., Concurrent stochastic methods for global optimization. *Mathematical Programming*, **46** (1990), 1-29.

[3] Dixon, L.C.W., ACSIM – an accelerated constrained simplex technique, *Computer Aided Design*, **5** (1973), 22-32.

[4] Fletcher, R., *Practical Methods of Optimization*, Wiley, New York, 1987.

[5] Gay, D.M., A trust-region approach to linearly constrained optimization, pp. 72-105 in: *Numerical Analysis* (Griffiths, D.F., ed.), Lecture Notes in Mathematics 1066, Springer, Berlin 1984.

[6] Gill, P.E., Murray, W. and Wright, M.H., *Practical Optimization*, Academic Press, London, 1981.

[7] Glad, T. and Goldstein, A., Optimization of functions whose values are subject to small errors, *BIT* **17** (1977), 160-169.

[8] IMSL, Inc., *Fortran Library 1.1*, 2500 Permian Tower, 2500 City West Boulevard, Houston, Texas USA, 1990.

[9] Karidis, J.P. and Turns, S.R., Efficient optimization of computationally expensive objective functions, *IBM Research Report* RC10698 (#47972), Yorktown Heights, NY (1984).

[10] Moré, J.J., Recent developments in algorithms and software for trust region methods, pp. 256-287 in *Mathematical Programming, The State of the Art* (A. Bachem et al., eds.), Springer, Berlin 1983.

[11] Moré, J.J., Garbow, B.S. and Hillstrom, K.E., Testing Unconstrained Optimization Software, *ACM Trans. Math. Software* **7** (1981), 17-41.

[12] Numerical Algorithm Group (NAG), *Fortran Library Mark* 14, Oxford, 1990.

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

[14] Nocedal, J., Theory of algorithms for unconstrained optimization, pp. 199-242 in: *Acta Numerica* 1992 (A. Iserles, ed.), Cambridge University Press, Cambridge 1992.

[15] Powell, M.J.D., A direct search optimization method that models the objective and constraint functions by linear interpolation, *Numerical Analysis Reports*, DAMTP 1992/NA5, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, England, April 1992.

[16] Rinnooy Kan, A.H.G. and Timmer, G.T., Stochastic methods for global optimization. *Amer. J. Math. Management Sci.*, **4** (1984), 7-10.

[17] Törn, A. and Žilinskas, A., *Global Optimization*, Lecture Notes in Computer Science 350, Springer, Berlin 1989.

[18] Torczon V., Multi-Directional Search: A Direct Search Algorithm for Parallel Machines, Ph.D. thesis, *Tech. Report TR90-7*, Dept. of Math. Sciences, Rice University, Houston TX, 1989.

[19] Torczon, V.J., On the convergence of the multidirectional search algorithm. *SIAM J. Optimization*, **1** (1991), 123-145.

[20] Torczon, V.J., On the convergence of pattern search algorithms, *Tech. Report TR93-10*, Dept. of Math. Sciences, Rice University, Houston TX, 1993.