On a New Method for Derivative Free Optimization

Lennart Frimannslund
Department of Informatics
University of Bergen
Bergen, Norway
Email: lennart@ii.uib.no

Trond Steihaug
Department of Informatics
University of Bergen
Bergen, Norway
Email: trond.steihaug@ii.uib.no

***Abstract*—A new derivative-free optimization method for unconstrained optimization of partially separable functions is presented. Using average curvature information computed from sampled function values the method generates an average Hessian-like matrix and uses its eigenvectors as new search directions. Numerical experiments demonstrate that this new derivative free optimization method has the very desirable property of avoiding saddle points. This is illustrated on two test functions and compared to other well known derivative free methods. Further, we compare the efficiency of the new method with two classical derivative methods using a class of testproblems.**

***Keywords*-Generating Set Search, Derivative-Free Optimization, Saddle points, Sparsity.**

I. Introduction

Continuous optimization is an important area of study, with applications in statistical parameter estimation, economics, medicine, industry — simply put, anywhere a mathematical model can be used to represent some real-world process or system which is to be optimized. Mathematically, we can express such a problem as

$$\min_{x \in D \subseteq \mathbb{R}^n} f(x), \tag{1}$$

where f is the objective function, based on the model which is defined on the domain D. These models can range from simple analytic expressions to complex simulations. Well known optimization methods such as Newton's method use derivatives to iteratively find a solution. These derivatives must somehow be provided, either through explicit formulas/computer code, or, for instance, automatic differentiation.

Suppose, however, that the objective function is produced by some sort of non-differentiable simulation, or that it involves expressions which can only be computed numerically, such as the solution to differential equations, integrals, and so on. In this case derivatives might not exist, or they may be unavailable if the numerically computed function is subject to some kind of adaptive discretization and truncation and therefore is non-differentiable, unlike the underlying mathematical function. In these cases derivative-based methods are not directly applicable, which leads to the need of methods that do not explicitly require derivatives. For an introduction to derivative free methods the reader is referred to [3].

Generating set search (GSS) methods are a subclass of derivative-free methods for unconstrained optimization. These methods can be extended to handle constraints, but we will focus on the unconstrained case where the domain D in the problem (1) is equal to $\mathbb{R}^n$. A comprehensive introduction to these methods can be found in [14]. In their most basic form these methods only use function values and do not collect any information such as average slope or average curvature information. Computing this information, however, can significantly speed up convergence, and this is done in the methods presented in [4], [6].

In addition, information about the structure of the function known a priori can also be useful. Suppose that the objective function f can be written as a sum of element functions,

$$f = \sum_{i=1}^{m} f_i,$$

where each element function has the property that it is unaffected when we move along one or more of the coordinate directions. For example, we might have

$$f(x_1, x_2, x_3) = f_1(x_1, x_2) + f_2(x_2, x_3). \tag{2}$$

Then, the function is said to be partially separable [10] and we say that f_i has a large null space. If f is partially separable and twice continuously differentiable, then its Hessian matrix,

$$\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix},$$

will be sparse. For the function (2) the Hessian element $\frac{\partial^2 f}{\partial x_1 \partial x_3}$ will be zero. If the function (2) is not twice continuously differentiable, then the matrix of the corresponding finite differences, that is, the matrix with

$$\Big[f(x_1+h,x_2,x_3+k)-f(x_1+h,x_2,x_3)$$

$$-f(x_1,x_2,x_3+k)+f(x_1,x_2,x_3)\Big]\Big/(hk)=0, \quad (3)$$

in position $(i,j) = (1,3)$ (and with similar expressions for all other (i,j)-pairs) will be sparse for any x, and any nonzero h and k, none of which have to be the same for each (i,j)-pair. The sparsity structure is the same as for the differentiable case, so that the expression (3) is identically zero. This result can be extended to any partially separable function, as proved in [7].

In [23], a GSS method which exploits such structure is presented, which is applicable to the case where these element functions are individually available.

In this paper, we present a GSS method which takes advantage of the partially separable functions, without requiring the element functions (which may or may not be differentiable) to be available. It is an extension of the paper [6]. We use the concept of average curvature introduced in [6].

This paper is organized as follows. In section II, we outline a basic framework for GSS, as well as the previous work of the authors on which the present paper is based. In Sections III and IV, we present the handling of partially separable functions and the convergence to second order stationary points. Section V contains a discussion of the methods used in the comparison and Section VI specifies the test functions. The main part of this paper is the testing presented in Section VII. Here, we define region of convergence and in the two sections VII and VIII we present the numerical properties of the methods on the two test functions. In Section IX, we show the efficiency of method derived in this paper compared to two classical methods for derivative free optimization. Concluding remarks are given in Section X.

II. Generating set search using curvature information

We restrict ourselves to a subset of GSS methods, namely sufficient decrease methods with $2n$ search directions, the positive and negative of n mutually orthogonal directions, of unit length. These directions will in general *not* be the coordinate directions. A simplified framework for the methods we consider is given in Figure 1. The univariate function ρ must be nondecreasing and satisfy $\lim_{x\downarrow 0}\frac{\rho(x)}{x} = 0$. For simplicity, increasing the step length can be thought of as multiplying it by 2, and decreasing it as dividing by 2, although these rules may be more advanced. For the formal requirements on these rules, see [14]. Given mild requirements on the function f the step length δ will ultimately go to zero, and the common convergence criterion for all GSS methods is that δ is smaller than some tolerance.

Given set of search directions $\mathcal{Q}$, step length δ and an initial guess $x \leftarrow x_0$.
While δ is larger than some tolerance
  Repeat until x has been updated or all $q \in \mathcal{Q}$ have been used:
    Get next search direction $q \in \mathcal{Q}$.
    If $f(x+\delta q) < f(x) - \rho(\delta)$
      Update x: $x \leftarrow x + \delta q$.
      Optionally increase δ.
    End if
  End repeat
  If no search direction provided a better function value, decrease δ.
  Optionally update $\mathcal{Q}$.
End while

Figure 1. Simplified framework for a sufficient decrease GSS method.

As can be seen from the pseudo code in Figure 1, the set of search directions can be periodically updated. In [6], the authors present a method that computes average curvature information from previously sampled points, assembles this information in a Hessian-like matrix and uses the eigenvectors of this matrix as the search directions, which amounts to a rotation of the old search directions. Once this rotation has been performed, the process restarts, and new curvature information is computed, periodically resulting in new search directions. It is shown that the efficiency of the method can be greatly improved compared to just using the coordinate directions as the search directions throughout.

A similar scheme, which aligns the basis to the average direction the search progresses, appeared as early as 1960 in [24] and implemented in 1973 [16]. To illustrate the idea of curvature information we use a quadratic model function by assuming we are minimizing, say,

$$g(y) = \phi + b^T(y-x) + \frac{1}{2}(y-x)^T C(y-x),$$

where C is a symmetric matrix. The search directions are positive and negative of the column vectors of the orthogonal matrix Q, that is,

$$Q = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix}.$$

Since g is a quadratic function, we have

$$q_i^T C q_j = \frac{g(x+\delta_i q_i + \delta_j q_j) - g(x+\delta_i q_i) - g(x+\delta_j q_j) + g(x)}{\delta_i \delta_j}.$$

For a general function f the computation of curvature information can be done in the following way, which is a slight modification of the methodology presented in [6]. Consider Figure 2, and assume that the current point is the

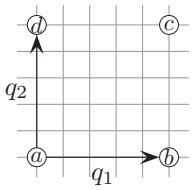

Figure 2. Location of sampled points used for curvature computation.

Outcome	Notes
SS	The search along q_1 moves the current best point to b, and the search along q_2 moves the current best point to c. The function value at d must be computed separately.
SF	The search along q_1 moves the current best point to b, and the search along q_2 computes the function value at c, but does not move the current best point. The function value at d must be computed separately.
FS	The search along q_1 computes the function value at point b, but does not move the current best point. The search along q_2 computes the function value at point d. The function value at point c must be computed separately.
FF	Neither the search along q_1 nor q_2 update the current best point, but the function values at points b and d are obtained. The function value at point c must be computed separately.

Table I
THE FOUR POSSIBLE OUTCOMES WHEN SEARCHING ALONG TWO CONSECUTIVE DIRECTIONS. S MEANS SUCCESS, F MEANS FAILURE.

point marked a, and that the next two search directions in the repeat-loop in the pseudo code are the directions shown, q_1 and q_2. When searching along two directions in a row, there are four possible outcomes. Success-success (both the search along q_1 and q_2 produce function values which satisfy the sufficient decrease condition), success-failure (the search along q_1 produces a sufficiently lower function value, but the search along q_2 does not), failure-success, and finally failure-failure. In all of these four cases, by computing the function value at a fourth point, the function values at four points in a rectangle can be obtained. The details are given in Table I. The function values at four such points a, b, c and d can be inserted into the formula

$$\frac{f(c) - f(b) - f(d) + f(a)}{\|b - a\| \, \|d - a\|}. \tag{4}$$

If the objective function is twice continuously differentiable, then the next lemma will show that (4) is equal to $q_1^T \nabla^2 f(\hat{x}) q_2$, where $\hat{x}$ is some point within the rectangle $abcd$. If the function is not twice continuously differentiable, (4) captures the average curvature in the rectangle.

The rectangle lies in the plane spanned by the search directions q_1 and q_2 since these were used consecutively. By successively reordering how the "get next search direction" statement considers the directions in $\mathcal{Q}$, one can obtain curvature information with respect to all the $n(n-1)/2$ possible different combinations of search directions, in a finite and uniformly bounded number of steps, which depends on n since there are $O(n^2)$ elements of curvature information which must be assembled. (For this reason, the method is not suitable for n larger than about 30, but exploiting structure can allow for much larger n, as will be explained in Section III.)

The following lemma is a slightly modified version of [5, Lemma 3.5] and can be found in calculus textbooks usually as a part of showing that the Hessian matrix is symmetric if the function is sufficiently smooth.

Lemma 1: Suppose the objective function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is twice continuously differentiable, assume we have given two orthogonal search directions q_i and q_j, and have computed

$$f(x), f(x + hq_i), f(x + kq_j), \text{ and } f(x + hq_i + kq_j)$$

for some x and some scalars h and k. Let element ij, $i > j$ of the symmetric matrix C_Q be

$$(C_Q)_{ij} = \frac{f(x + hq_i + kq_j) - f(x + hq_i) - f(x + kq_j) + f(x)}{hk}.$$

Then,

$$(C_Q)_{ij} = q_i^T \nabla^2 f(\hat{x}) q_j,$$

where $\hat{x} = x + \tau h q_i + \sigma k q_j$ for some $\tau, \sigma \in [0, 1]$.
The matrix C_Q contains $q_i^T \nabla^2 f(\hat{x}) q_j$ in positions (i, j) and (j, i), which is curvature information with respect to the coordinate system defined by the n directions $q_1, \ldots, q_n$ in $\mathcal{Q}$. Note that the point $\hat{x}$ is different for each (i, j)-pair. Also note that both $q_i \in \mathcal{Q}$ and $-q_i \in \mathcal{Q}$. The diagonal elements of C_Q must be computed separately, for instance when the step length is reduced, since the preceding repeat-loop, combined with the current f-value then gives the function values at three equally spaced points on a straight line for all n search directions.

Once the matrix C_Q is complete, it is subjected to the rotation

$$C = Q C_Q Q^T, \tag{5}$$

where Q is the matrix with the n unique search directions as its columns, ordered so that they correspond to the ordering of the elements in C_Q. C now contains curvature information with respect to the standard coordinate system. The search directions in $\mathcal{Q}$ are then replaced with the positive and negative of the eigenvectors of C.

To build up C_Q in a systematic fashion we need to specify one way to choose the order. For instance, for $n = 4$ and one wants to compute $(C_Q)_{21}$, $(C_Q)_{31}$, $(C_Q)_{24}$, and $(C_Q)_{34}$, then one can let the order of the directions be:

$$q_1, \quad q_2, \quad -q_1, \quad q_3, \quad -q_2, \quad q_4, \quad -q_3 \quad -q_4 \ .$$

Here, the search along q_1 and q_2 enables us to compute $(C_Q)_{21}$. The directions $-q_1$ and q_3 provide us with $(C_Q)_{31}$,

and so on. A discussion and analysis of ordering are found in Macklem [17].

We now investigate the relationship between C and $\nabla^2 f(x)$. The search directions are the orthogonal directions $q_1, \ldots, q_n$ and assume that the elements $(C_Q)_{ij}$ of the symmetric $m \times m$ matrix C_Q have been computed at the points

$$\left\{x^{ij},\ x^{ij} + h^{ij} q_i,\ x^{ij} + k^{ij} q_j,\ x^{ij} + h^{ij} q_i + k^{ij} q_j\right\}, \tag{6}$$

for all $(i, j),\ i \geq j$ and $(C_Q)_{ji}$ set to be equal to $(C_Q)_{ij}$. Let N be the union of all such points and let

$$\delta = \max_{z,y \in N} \|z - y\|, \tag{7}$$

and

$$\mathcal{N} = \left\{ x \in \mathbb{R}^n \mid \max_{y \in N} \|x - y\| \leq \delta \right\}. \tag{8}$$

Lemma 2: Assume that f is twice continuously differentiable and $\nabla^2 f$ is Lipschitz-continuous in $\mathcal{N}$

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq L\|x - y\|, \text{ for all } x, y \in \mathcal{N}.$$

Let the $n \times n$ symmetric matrix C_Q be computed with the points (6). Then any $x \in \mathcal{N}$ satisfies

$$\|Q C_Q Q^T - \nabla^2 f(x)\| \leq nL\delta. \tag{9}$$

The proof can be found in [6] and we will in the next section prove a more general result. In case of a quadratic function $L = 0$, the exact Hessian matrix is recovered. The second derivative is only required to be locally Lipschitz with respect to $\mathcal{N}$.

III. Extension to separable functions

Suppose the function f is partially separable. As mentioned in the introduction, the Hessian will be sparse if f is twice continuously differentiable, and if the Hessian is not defined, the matrix of average curvature information will be sparse [7]. Let r be the number of nonzero elements in the lower diagonal of these curvature matrices. Then, even though the matrix C can be restricted to have this sparsity pattern, the matrix C_Q cannot be assumed to be sparse, since we cannot expect the finite differences (4) to be zero for arbitrary search directions $\mathcal{Q}$. However, sparsity can still be exploited.

Given two matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{r \times s}$, the Kronecker product $A \otimes B$ is a $mr \times ns$ block matrix given as

$$A \otimes B = \begin{bmatrix} A_{11}B & \cdots & A_{1n}B \\ \vdots & & \vdots \\ A_{m1}B & \cdots & A_{mn}B \end{bmatrix}. \tag{10}$$

The Kronecker product is useful in the present context because of the relation

$$AXB = C \Leftrightarrow (B^T \otimes A)\mathbf{vec}(X) = \mathbf{vec}(C). \tag{11}$$

Here $\mathbf{vec}(X)$ and $\mathbf{vec}(C)$ are vectors containing the entries of the matrices X and C stacked row-wise [13].

Using (10) and (11) the rotation (5) can be written implicitly as

$$(Q^T \otimes Q^T)\mathbf{vec}(C) = \mathbf{vec}(C_Q). \tag{12}$$

Since we impose a sparsity structure on C as well as symmetry, all the entries in the upper triangle, as well as all the zero entries of $\mathbf{vec}(C)$ can be removed from (12), resulting in the overdetermined equation system

$$(Q^T \otimes Q^T) P_c \overline{\mathbf{vec}}(C) = \mathbf{vec}(C_Q), \tag{13}$$

where the vector $\overline{\mathbf{vec}}(C)$ contains the r elements of C to be determined, and the $n^2 \times r$ 0-1 matrix P_c adds together the columns corresponding to upper and lower diagonal elements C_{ij} and C_{ji} for all off-diagonal elements, and deletes the columns corresponding to zero entries in C. For example, if C is to be tridiagonal and is of size 3×3, that is,

$$C = \begin{bmatrix} \times & \times & \\ \times & \times & \times \\ & \times & \times \end{bmatrix},$$

then it has one zero element and five nonzero elements in the lower triangle, so that P_c has size 9×5 and reads:

$$P_c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{14}$$

Since the equation system (13) is overdetermined, we can select r rows from the coefficient matrix and the right-hand side, resulting in the $r \times r$ equation system

$$P_{\text{row}}(Q^T \otimes Q^T) P_c \overline{\mathbf{vec}}(C) = P_{\text{row}} \mathbf{vec}(C_Q), \tag{15}$$

where P_{row} is an $r \times n^2$ 0-1 matrix which selects r rows. P_{row} will be the first r rows of a permuted $n^2 \times n^2$ identity matrix. The resulting equation system (15) will be significantly smaller than its counterpart (12) when a sparsity structure is imposed on C, and the corresponding effort required to compute the right-hand side is similarly smaller. If there are only $O(n)$ elements to be determined, then the number of steps needed to compute the entire right-hand side $P_{\text{row}}\mathbf{vec}(C_Q)$ does not depend on n, which does away with the practical limit on dimension discussed in the previous section.

Exactly which rows P_{row} should select in order to create a well-conditioned coefficient matrix is nontrivial, and is sometimes called the subset selection problem in the literature (see e.g., [9]). One suitable solution procedure is to

determine these rows by computing a strong rank-revealing QR factorization of the transpose of $P_{\text{row}}(Q^T \otimes Q^T)$ and selecting the rows chosen by the theory and algorithms of Gu and Eisenstat, presented in [11]. An implementation of this selection procedure can be found in [21].

IV. Convergence theory

The method presented so far, being a sufficient decrease method with $2n$ search directions which are the positive and negative of n mutually orthogonal directions, adheres to the algorithmic framework and convergence theory of Lucidi and Sciandrone [15]. We can therefore state the following theorem, without proof.

Theorem 3: Suppose f is continuously differentiable, bounded below and the level set $\mathcal{L}(x) = \{y \mid f(y) \leq f(x)\}$ is compact. Then, the method converges to a first-order stationary point.

We now prove that if f is twice continuously differentiable, then the computed curvature matrix C converges to the true Hessian in the limit.

Define

$$A = P_{\text{row}}(Q^T \otimes Q^T)P_c.$$

Let f be twice continuously differentiable and Hessian Lipschitz-continuous in the sense that

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq L\|x - y\|. \tag{16}$$

Define r pairs of vectors $p^{(k)}, q^{(k)}$ $k = 1, \ldots, r$, all of unit length, such that the kth row of A is equal to

$$\left(p^{(k)T} \otimes q^{(k)T}\right) P_c. \tag{17}$$

This means some of these vectors will be equal, but the pairs will be unique. In addition let r points x^k, $k = 1, \ldots, r$, be such that element k of $P_{\text{row}}\mathbf{vec}(C_Q)$,

$$(P_{\text{row}}\mathbf{vec}(C_Q))_k = p^{(k)T}\nabla^2 f(x^k)q^{(k)}.$$

Let η be such that

$$\max_{i,j} \|x^i - x^j\| = \eta.$$

Let $\mathcal{N}$ be the neighborhood of points such that

$$\mathcal{N} = \left\{x \mid \|x - x^k\| \leq \eta,\ k = 1, \ldots, r\right\}.$$

For convenience, let us restate (15), as

$$A\overline{\mathbf{vec}}(C) = P_{\text{row}}\mathbf{vec}(C_Q). \tag{18}$$

Lemma 4: Assume A is invertible. Let C be the symmetric $n \times n$ matrix constructed from the solution of (18). Then, there exists an $x \in \mathcal{N}$ such that

$$\|\nabla^2 f(x) - C\| \leq \|A^{-1}\| nL\eta.$$

Proof. Let us rewrite the contents of $P_{\text{row}}\mathbf{vec}(C_Q)$:

$$\begin{aligned} & (P_{\text{row}}\mathbf{vec}(C_Q))_k \\ = \ & p^{(k)T}\nabla^2 f(x^k)q^{(k)} \\ = \ & p^{(k)T}\left(\nabla^2 f(x) + \nabla^2 f(x^k) - \nabla^2 f(x)\right) q^{(k)} \\ = \ & \left[p^{(k)T}\nabla^2 f(x)q^{(k)}\right] + \\ & \left[p^{(k)T}(\nabla^2 f(x^k) - \nabla^2 f(x))q^{(k)}\right]. \end{aligned} \tag{19}$$

Then, and in addition defining $h = \overline{\mathbf{vec}}(\nabla^2 f(x))$, equation (18) can be written as

$$A\overline{\mathbf{vec}}(C) = Ah + \epsilon. \tag{20}$$

Here $(Ah)_k$ is the expression in the first parenthesis of (19), and ϵ_k is the expression in the last parenthesis of (19). If we consider the norm of a single element in ϵ, this is

$$\begin{aligned} |\epsilon_k| &\leq \|p^{(k)}\| \, \|\nabla^2 f(x^k) - \nabla^2 f(x)\| \, \|q^{(k)}\| \\ &\leq L\eta, \end{aligned} \tag{21}$$

using (16) and the fact that p and q have unit length. When solving (18), we get

$$\overline{\mathbf{vec}}(C) = h + A^{-1}\epsilon.$$

If we consider a single element of $\overline{\mathbf{vec}}(C)$ and h we can write

$$|(\overline{\mathbf{vec}}(C))_k - h_k| \leq \|A^{-1}\||\epsilon_k|,$$

which can also be written

$$|C_{ij} - (\nabla^2 f(x))_{ij}| \leq \|A^{-1}\||\epsilon_k| \tag{22}$$

Using the property of the 2-norm that

$$\|A\|_2 \leq n \max_{i,j} |a_{ij}|,$$

as well as (21) we can extend (22) to

$$\|C - \nabla^2 f(x)\| \leq \|A^{-1}\| nL\eta,$$

which completes the proof. □

We must now prove that there always exists a matrix A with rank r, and that the term $\|A^{-1}\|$ is uniformly bounded. Since A is made up of the rows of the matrix $(Q^T \otimes Q^T)P_c$, there will be a choice of rows which imply full rank if the matrix $(Q^T \otimes Q^T)P_c$ has rank r.

Lemma 5: For any orthogonal matrix Q and any sparsity structure to be imposed on C, the matrix $(Q^T \otimes Q^T)P_c$ has full rank r, and its smallest singular value σ_r satisfies $\sigma_r \geq 1$.

Proof. Since Q is orthogonal, so is Q^T, and also $(Q^T \otimes Q^T)$. For any sparsity structure, right-multiplying $(Q^T \otimes Q^T)$ with P_c either adds together two columns, or deletes columns. Consequently, the columns of the resulting matrix $(Q^T \otimes Q^T)P_c$ are orthogonal (which implies full rank), and have either length one or length $\sqrt{2}$. It then

follows that the singular values are equal to the length of the column vectors, either 1 or $\sqrt{2}$. □

If we consider (14) and the corresponding $(Q^T \otimes Q^T)P_c$, the norm of first column of $(Q^T \otimes Q^T)P_c$ is 1 and the norm of the second column is $\sqrt{2}$.

Lemma 6: P_{row} can be chosen such that for a given n, the smallest singular value of A is uniformly bounded below, and consequently that $\|A^{-1}\|$ is uniformly bounded.

Proof. This result follows from the theory and methods of Gu and Eisenstat [11], which guarantee that the rows of A (or equivalently the columns of A^T, as is done in [11]) can be selected from the rows of $(Q^T \otimes Q^T)P_c$ in such a way that the smallest singular value of A is larger than or equal to the smallest singular value of $(Q^T \otimes Q^T)P_c$, divided by a low order polynomial in n and r. Since n and r are given and the smallest singular value of $(Q^T \otimes Q^T)P_c$ is always larger than or equal to 1, the result follows. □

Finally, we show that η goes to zero as the GSS method converges to a stationary point.

Lemma 7: Assume that the step length expansion factor is uniformly bounded by, say, M. Then, as the step length δ go to zero, so does η.

Proof. That the step length δ goes to zero is an integral part of the convergence theory of GSS methods and is proved in e.g. [14]. η is the diameter of neighborhood of points $\mathcal{N}$. Since all the points in $\mathcal{N}$ lie within the rectangles of points used in the formula (4), it follows that η must be smaller than maximum possible distance between the first and the last points used for computing C (the corner points of the rectangle $abcd$ in Figure 2). Suppose, that when the computation of C is started the step length is $\delta_{\max}$, and that the maximum possible number of step length increases before C is computed is t. Then we have

$$\eta \leq \sum_{k=0}^{t} \delta_{\max} M^{k-1}.$$

The only variable in this expression is $\delta_{\max}$, and we know it goes to zero as the method converges. Consequently, so must η. □

This allows us to state the following theorem:

Theorem 8: Assume that f is twice continuously differentiable, bounded below and that the level sets $\mathcal{L}(x)$ are compact. Then, as the method converges, C converges to the true Hessian.

The proof follows from the preceding Lemmas. This result, together with the preliminary numerical results in this paper allows us to conjecture that the method actually converges to second-order stationary points.

V. Testing derivative free optimization methods

The purpose of the following sections is to report on numerical experiments on two unconstrained optimization problems where methods risk terminating at a saddle point, avoiding a nearby strict local minimizer. For visualization purposes we have chosen problems with two unknowns.

The first problem is a modification of a problem suggested by Wolfe [26]. In its unmodified form this problem has been used to show that gradient based methods tend to converge to a saddle point. The modification will make the function bounded below and introduce a local minimizer but not change the region where gradient based methods converge to the saddle-point. The second example is a modification of a function presented in [1], which has a very narrow cone of negative curvature. Again the modification will make the function bounded below and introduce local minimizers.

Generating set search methods described in the previous sections are shown to converge to stationary points and the set of search directions in the limit will be the eigenvalues of the Hessian matrix at the solution. If the Hessian matrix at the stationary point has a negative eigenvalue, one of the search directions will be a descent direction. A generating set search should therefore not experience convergence to the saddle points of the two test functions. A generating set search method is compared with two methods which do not have the same property to generate descent at a saddle point.

In the second part of the experiments we compare the efficiency of GSS-CI with two classical derivative free methods on a class of test problems.

A. The methods

The three methods primarily used in testing, are GSS-CI, NEWUOA and NMSMAX. We will briefly discuss two additional methods, MDSMAX and fminsearch.

1) GSS-CI: This is the method presented in the previous sections and [8], [2], and is based on the method of [6]. Since the method gathers average slope information the method can consequently perform Newton-like steps at regular intervals.

The initial search directions are chosen to be the positive and negative coordinate vectors. Each pair of search directions (e.g. $\pm q_i$, where q_i is a search direction) has a step length δ_i associated with it. In our experiments these are initially set to the same value, $0.2\|x_0\|_1$, but they will be increased or decreased individually depending on the success or failure of the search along the corresponding pairs of search directions. A search is deemed successful it it produces sufficient decrease, that is, if

$$f(x + \delta_i q_i) < f(x) - \rho(\delta_i),$$

where $\rho : \mathbb{R} \mapsto \mathbb{R}$ is nondecreasing function satisfying a few technical requirements, outlined in [14]. In our implementation we use

$$\rho(\delta) = 10^{-4}\delta^2.$$

The termination criterion is that the product of all the step lengths should be less than or equal to a tolerance. In our

experiments this is

$$\prod_{i=1}^{n} \delta_i \leq \left(10^{-4}\|x_0\|_1\right)^n.$$

B. NEWUOA

NEWUOA [22] is an interpolation method, where the number of interpolation points can be determined by the user. The remaining degrees of freedom are taken up by minimizing the Frobenius norm of the difference between one Hessian approximation and the next.

An initial vector $x_0 \in \mathbb{R}^n$, the number m of interpolation points, and the initial and final values of a trust region radius, namely ρ_{beg} and ρ_{end} must be provided by the user. The number of interpolation points m is a fixed integer from the interval $n+2 \leq m \leq \frac{1}{2}(n+1)(n+2)$. It is recommended to use $m = 2n+1$ for efficiency. The initial interpolation points $x_i, i = 0, 1, 2, \ldots, m$, have the property that $\|x_i - x_0\|_2 = \rho_{\text{beg}}, i = 1, 2, \ldots, m$, unless $m > 2n + 1$, in which case this distance is $\sqrt{2}\rho_{\text{beg}}$. The termination criterion is related to the radius ρ_{end}.

C. NMSMAX

NMSMAX [12] is an implementation by Nicholas J. Higham of the classical Nelder-Mead simplex method [20]. The user can choose whether the initial simplex is right-angled or regular (with sides of equal length). The initial simplex size is not input by the user, but taken to be the order of $\max(\|x_0\|_\infty, 1)$. The method terminates when either the maximum number of function evaluations is reached, or when the relative size of the simplex, is below a certain threshold. That is,

$$\frac{1}{\max(1, \|v_0\|_1)} \max_{1 \leq i \leq n} \|v_i - v_0\|_1 \leq \text{tol}.$$

Here v_0 and $v_i,\ i = 1, \ldots, n$ are the vertices of the simplex. In our experiments we use $\text{tol} = 10^{-6}\|x_0\|_1$.

D. The methods MDSMAX and fminsearch

We also included a brief test of the methods MDSMAX, which is an implementation by Nicholas J. Higham of the multidirectional search method due to Virginia Torczon [25], and fminsearch [18], which is the Matlab implementation of the Nelder-Mead method.

VI. The functions

The two functions are in two variables, are twice continuously differentiable and bounded below.

A. Function I – A Narrow Positive Cone

The function (23) is a modification of a test function in [1]:

$$f(x, y) = (9x - y)(11x - y) + \frac{x^4}{2}. \qquad (23)$$

It has a saddle point at the origin, and two local minimizers at $(x, y) = \pm(1, 10)$. Level curves for this function can be seen in Figure 8.

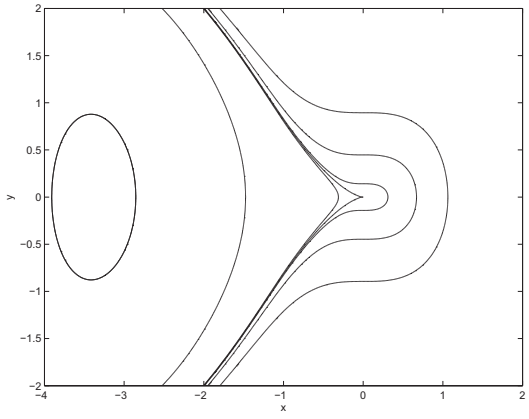

Figure 3. Level curves for the function (24).

B. Function II – Modified Wolfe Function

The second function (24) is a modified version of a test function due to Wolfe [26]:

$$f(x, y) = \frac{x^3}{3} + \frac{y^2}{2} - \frac{2}{3}(\min[x, -1] + 1)^3, \qquad (24)$$

It has a saddle point at the origin and a minimum at $(x, y) = (-2 - \sqrt{2}, 0)$. Level curves for this function are shown in Figure 3. The original function (not bounded below) is given by $f(x, y) = \frac{x^3}{3} + \frac{y^2}{2}$.

VII. Numerical experience

Region of Convergence: The region of convergence of a stationary point is the set of starting points for which a given method terminates close to the stationary point. In addition to the input parameters for the methods we need to specify the tolerance (or distance between) the terminating point and the stationary point. A globally convergent method on a sufficiently smooth function is characterized by for all starting points, the method will for any $\varepsilon > 0$ generate an iterate x_k so that $\|\nabla f(x_k)\| \leq \varepsilon$. However, the stopping criteria of the implementation may be based on changes in the function values or on the difference between two iterates. Even the case $\|\nabla f(x_k)\| \leq \varepsilon$ will in general not guarantee that the distance between the stationary point and x_k is small. We can thus expect that even if the methods terminate successfully, the distance to a stationary point will not be smaller than the tolerance for some starting points. For simplicity we say that a method terminates at a stationary point when it terminates at a point that satisfies the tolerance.

A. Function I

For this function we generate starting points in the fourth quadrant $\{(x, y) | x \leq 0, y \geq 0\}$. The minimizers of the function are in the first and third quadrants, so we expect the methods to terminate successfully at the minimizers

if started in these quadrants. This is confirmed in the preliminary numerical testing. Furthermore, because of the symmetry of the function we can choose either the second or fourth quadrants, at least for GSS-CI and NEWUOA.

GSS-CI: We discretize the area $[-8, 0] \times [0, 10]$, into a 201×201 grid of points, and start the method with initial step length $0.2\|x_0\|_1$ for all directions, and the termination criterion is that the product of all the step lengths should be less than or equal to $(10^{-4}\|x_0\|_1)^n$. (If $x_0 = 0$ then nonzero values are used.) The results are given in Figure 4. In the figure, a blue color means that the method terminated

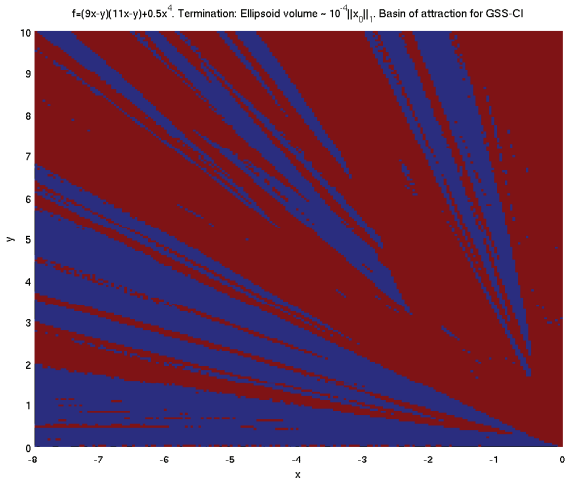

Figure 4. GSS-CI on the function $f = (9x_1 - x_2)(11x_1 - x_2) + \frac{1}{2}x_1^4$. The initial step length is $0.2\|x_0\|_1$ for all directions, and the termination criterion that the volume of the ellipsoid defined by the scaled search directions should be proportional to $10^{-4}\|x_0\|_1$, that is, $\prod_i \delta_i \leq [10^{-4}\|x_0\|_1]^n$. (The discretization is 201×201.)

close to $(x, y) = (1, 10)$ for the corresponding starting point, red color means termination close to $(x, y) = (-1, -10)$. As one can see, the method does not terminate close to the saddle point for any of these starting points. When starting at the origin, setting a nonzero step length results in convergence to a minimizer.

NEWUOA: We generate starting points on a 1001×1001 discretization of the region $[-8, 0] \times [0, 10]$, and run NEWUOA with parameters $\rho_{\text{beg}} = 0.2\|x_0\|_1$, and $\rho_{\text{end}} = 10^{-5}\|x_0\|_1$. (Once again, if $x_0 = 0$ then nonzero values are used.) The results are visualized in Figure 5. As before, blue color means that the method terminated close to $(x, y) = (1, 10)$ for the corresponding starting point, red color means termination close to $(x, y) = (-1, -10)$. In addition, green means termination close to the origin, and orange means none of the above. As one can see, the method does terminate close to the saddle point for some starting points, and these points make up a small region on the border between the basins of attraction of $(x, y) = (1, 10)$ and $(x, y) = (-1, -10)$.

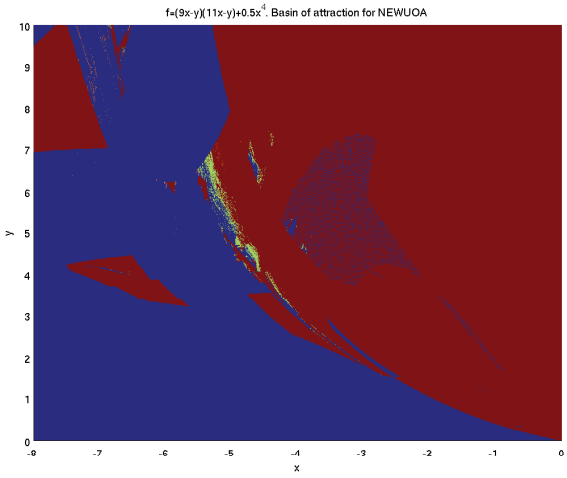

Figure 5. Plot of basins of attraction for NEWUOA on $f = (9x_1 - x_2)(11x_1 - x_2) + \frac{1}{2}x_1^4$, with $\rho_{\text{beg}} = 0.2\|x_0\|_1$ and $\rho_{\text{end}} = 10^{-5}\|x_0\|_1$. (The discretization is 1001×1001.)

To check if the basins of attraction are sensitive to the termination criterion we repeat the experiment, but this time with $\rho_{\text{end}} = 10^{-6}\|x_0\|_1$. The results are given in Figure 6. As we can see in this figure, the starting points for which the method terminates at the saddle point are still wedged between the red and blue regions, but the green region is now much smaller.

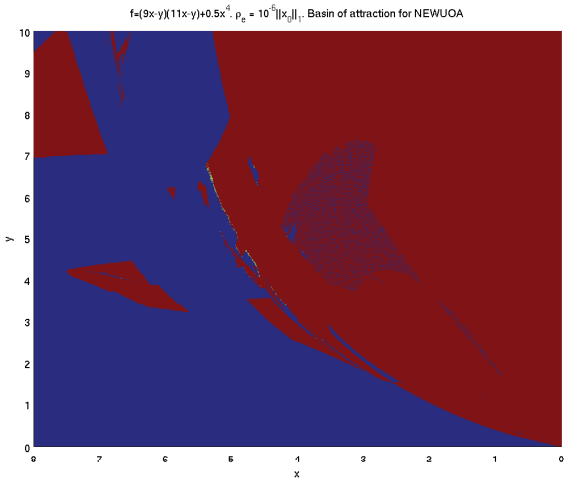

Figure 6. Decreasing ρ_{end} in NEWUOA to $10^{-6}\|x_0\|_1$ will basically not change the region of convergence for the local minimizers, but the region of convergence to the saddle-point gets smaller, squeezed between the regions of convergence to the local minimizers. (The discretization is 1001×1001.)

Similarly, we test what happens with a looser convergence criterion, namely $\rho_{\text{end}} = 10^{-4}\|x_0\|_1$. The results are in

Figure 7. For this convergence criterion the green region is

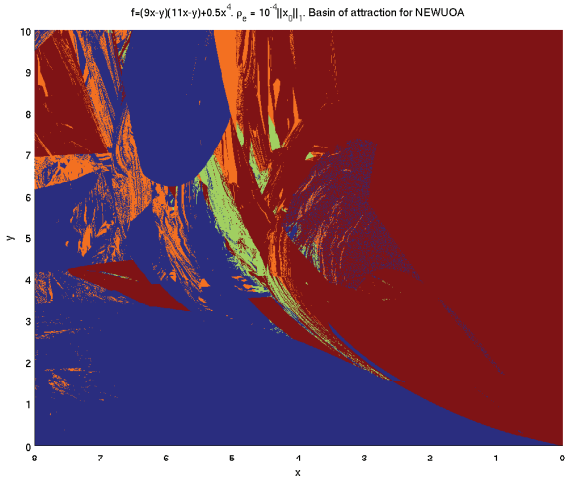

Figure 7. Increasing ρ_{end} in NEWUOA to $10^{-4}\|x_0\|_1$ will force many starting points not to be accepted as close to a stationary point. The regions are basically the same, but the region of convergence for the saddle-point is larger. (The discretization is 1001×1001.)

much larger, and there are also large orange regions, which correspond to termination no closer to any of the stationary points than 0.2.

NMSMAX: For NMSMAX, we discretize the region $[-10, 10] \times [-10, 10]$ into a 201×201 grid. We choose a right-angled initial simplex. The size of the initial simplex is not determined by the user, but we set the termination criterion to be a simplex size of $10^{-6}\|x_0\|_1$. The results, as well as level curves of the function are given in Figure 8. As one can see, for about half the fourth quadrant the method terminates at the saddle point, even though the convergence criterion is quite strict. In addition, termination at the saddle point occurs for points close to the negative y-axis, and along

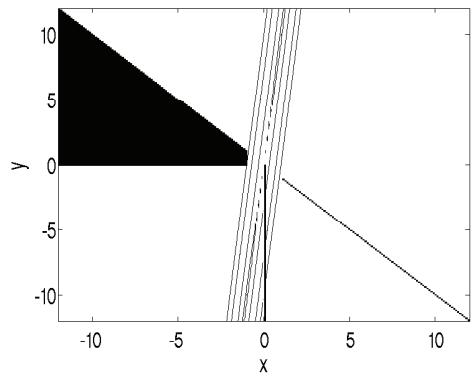

Figure 8. Level curves of the function (23), and starting points for which NMSMAX terminates at the saddle point at the origin, marked in black.

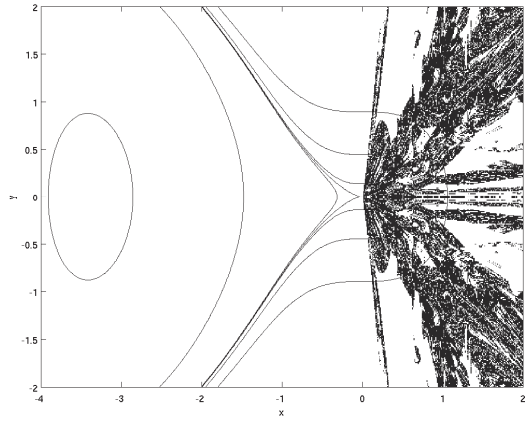

Figure 9. Level curves for the function (24) as well as starting points for which NEWUOA terminates at the saddle point at the origin, marked in black.

the line $y = -x$.

It is also interesting to note that in this case, the behavior in quadrants two and four are *not* the same.

B. Function II

For this function we discretize the region $[-4, 2] \times [-2, 2]$ into a 601×401 grid.

GSS-CI: Using the same parameter settings as for function I, GSS-CI once again terminates at the (single) minimizer, so an attraction basin plot would simply be the region filled with one color. (When $x_0 = 0$, nonzero step lengths are used, and the method converges to the minimizer.)

NEWUOA: For this function we also use the same parameter values as before, namely $\rho_{\text{beg}} = 0.2\|x_0\|_1$ and $\rho_{\text{end}} = 10^{-6}\|x_0\|_1$. The results are in Figure 9. As one can see, there is a relatively large collection of points in the first and second quadrants, for which the method terminates at the saddle point at the origin.

To see if the cause of this behavior was the number of interpolation points ($2n + 1$ in this case), we also tried a full quadratic model, by using six interpolation points. The results for this case are in Figure 10. As one can see, the black region now has a different shape, but is located approximately in the same position, and is of similar size.

NMSMAX: For this function, NMSMAX terminates at the saddle point for a few starting points on the y-axis only.

VIII. Testing the methods MDSMAX and FMINSEARCH

MDSMAX: The results are reported in Figures 11 and 12 and Figures 13 and 14. For the function (24) termination close to the saddle points rarely occurs, and when it does the corresponding starting points lie along straight lines, one at the upper right corner of Figure 14, and one on the y-axis close to the bottom of the figure using right angled simplex.

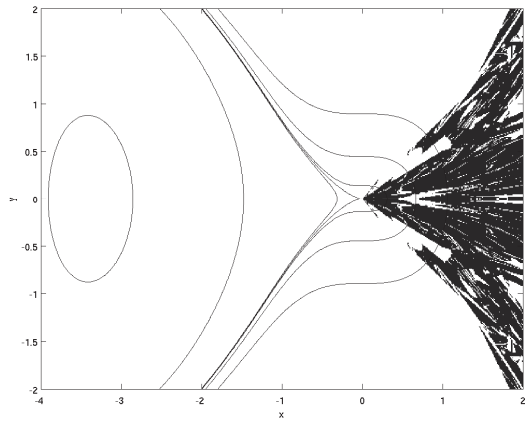

Figure 10. Level curves for the function (24) as well as starting points for which NEWUOA terminates at the saddle point at the origin, marked in black. Number of interpolation points $m = 6$.

However, MDSMAX has serious problems with stagnation on the function (23), as can be seen in Figures 11 and 12.

fminsearch: This method has few problems on these functions, except when the starting points lie on one of the axes. For the function (23), 208 of the 40401 starting points result in termination close to the saddle point, 201 of these 208 points being on the x-axis. For the function (24), 890 of the 241001 starting points result in termination close to the saddle point, all of these on or immediately next to the y-axis. These results were obtained using the standard convergence tolerances. Tightening the convergence criteria gives an even more favorable result.

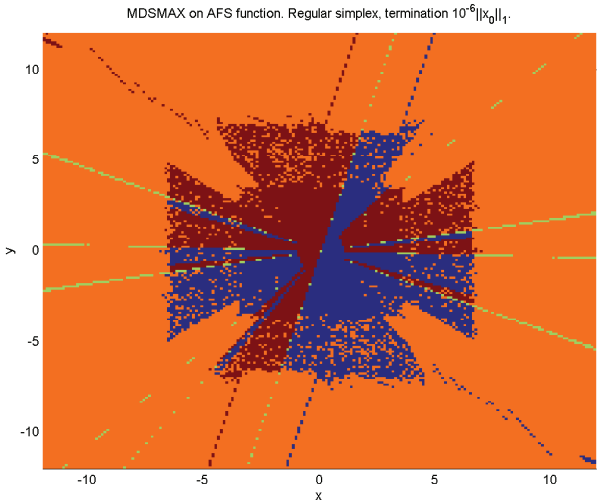

Figure 11. MDSMAX on the function $f = (9x - y)(11x - y) + \frac{x^4}{2}$. Red means termination close to $(x, y) = (1, 10)$, blue means termination close to $(x, y) = (-1, -10)$, green termination close to the saddle point at the origin, and orange means stagnation. Regular simplex.

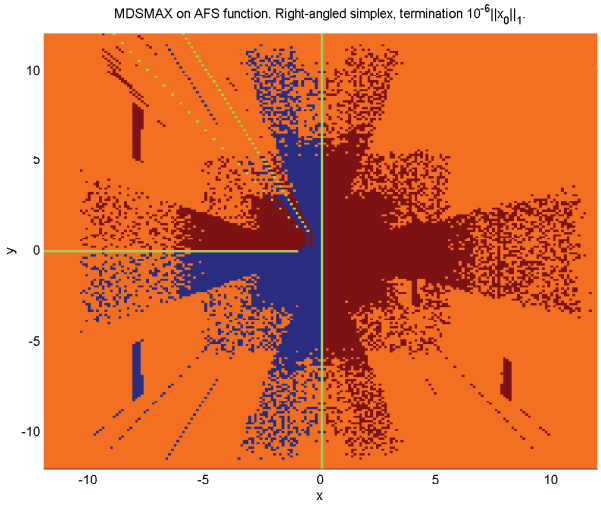

Figure 12. MDSMAX on the function $f = (9x - y)(11x - y) + \frac{x^4}{2}$. Red means termination close to $(x, y) = (1, 10)$, blue means termination close to $(x, y) = (-1, -10)$, green termination close to the saddle point at the origin, and orange means stagnation. Right-angled simplex.

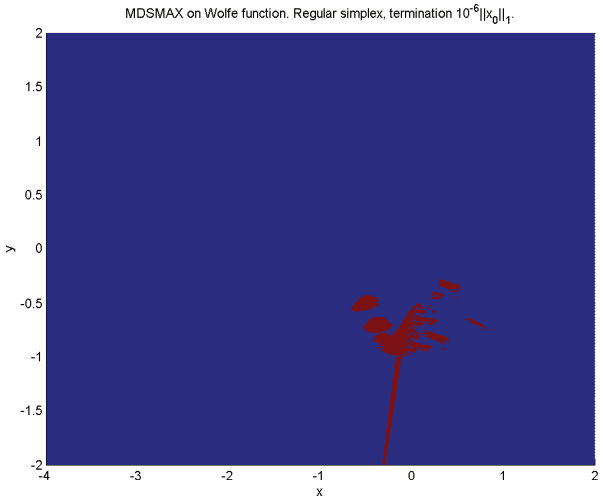

Figure 13. MDSMAX on the function $f = \frac{x^3}{3} + \frac{y^2}{2} - \frac{2}{3}(\min[x, -1] + 1)^3$. Blue means termination close to the minimum at $(x, y) = (-2 - \sqrt{2}, 0)$, red means termination close to the saddle point at the origin. Regular simplex.

IX. Efficiency

Moré and Wild [19], benchmarked different derivative-free optimization solvers on 53 smooth problems. In this test we use the same set of problems and two of the same solvers (NEWUOA and NMSMAX) as Moré and Wild [19]. We run the three methods on each problem, and declare a success if a method uses less than 5000 function evaluations, and the gradient corresponding to the solution satisfies $\|\nabla f(x)\| \leq 10^{-2}$, where this gradient is computed with finite differences. The corresponding data profile is

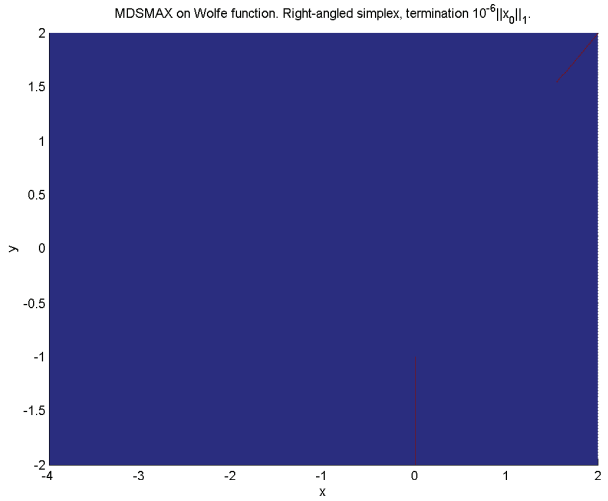

Figure 14. MDSMAX on the function $f = \frac{x^3}{3} + \frac{y^2}{2} - \frac{2}{3}(\min[x, -1] + 1)^3$. Blue means termination close to the minimum at $(x, y) = (-2 - \sqrt{2}, 0)$, red means termination close to the saddle point at the origin. Right-angled simplex.

shown in Figure 15. The horizontal axis is number of equivalent gradient evaluations, i.e. one equivalent gradient is n function evaluations. As one can see from the figures NEWUOA solves the most problems if one has a tight computational budget, and GSS-CI solves the most problems if one has as moderate computational budget. NMSMAX performs the weakest among the methods on these functions.

X. Discussion

The simplex method [20] is one of the most used derivative free optimization methods. In this note we have used the implementations NMSMAX and fminsearch of the simplex method. The other three methods discussed in the paper represents different approaches of derivative free optimization. We have shown that GSS-CI solves the most problems if one has as moderate computational budget compared to NEWUOA and NMSMAX. The methods NEWUOA, NMSMAX, and fminsearch may terminate close to the saddle point while GSS-CI will not converge to the saddle point for these two examples. This supports the observed convergence properties of GSS-CI. The regions of convergence are dependent on the input parameters and the results presented are typical behavior of the methods.

Acknowledgements

The authors would like to thank Mike Powell and Nicholas J. Higham for helpful comments on the experiments and on an earlier version of the section on basin of attraction. The authors would also like to thank Marielba Rojas for the help on rank revealing QR.

The first author gratefully acknowledges partial funding from The Norwegian Research Council, Gassco and Statoil.

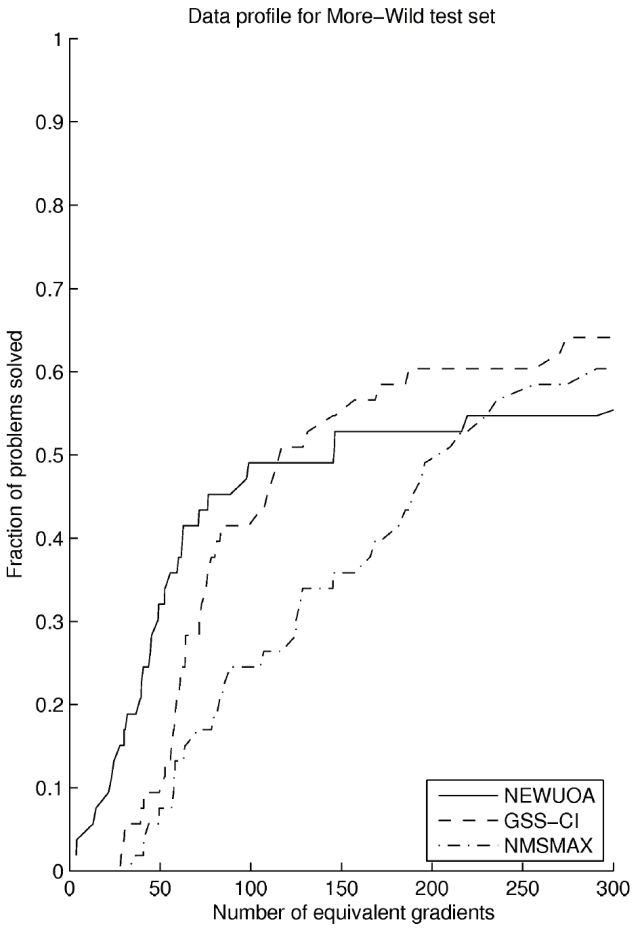

Figure 15. Data profile for the smooth functions of the test set of Moré and Wild [19].

References

[1] M. A. Abramson. Second-order behavior of pattern search. *SIAM Journal on Optimization*, 16(2):315–330, 2005.

[2] M. A. Abramson, L. Frimannslund, and T. Steihaug. A subclass of generating set search with convergence to second-order stationary points. To be submitted, 2011.

[3] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to Derivative-Free Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009.

[4] I. D. Coope and C. J. Price. A direct search conjugate directions algorithm for unconstrained minimization. *The ANZIAM Journal*, 42(E):C478–C498, 2000.

[5] C. H. Edwards. *Advanced Calculus of Several Variables*. Academic Press, 1973. ISBN 0–12–232550–8.

[6] L. Frimannslund and T. Steihaug. A generating set search method using curvature information. *Computational Optimization and Applications*, 38(1):105–121, 2007.

[7] L. Frimannslund and T. Steihaug. Sparsity of the average curvature information matrix. *PAMM, Proc. Appl. Math. Mech.*, 7:1062101–1062102, 2007.

[8] L. Frimannslund and T. Steihaug. A new generating set search algorithm for partially separable functions. In *Proceedings ADVCOMP 2010: The Fourth International Conference on Advanced Engineering Computing and Applications in Sciences*, pages 65–70. The International Academy, Research and Industry Association (IARIA), 2010. ISBN:978-1-61208-000-0.

[9] G. H. Golub and C. F. van Loan. *Matrix computations*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

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

[11] M. Gu and S. C. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM Journal on Scientific Computing*, 17(4):848–869, 1996.

[12] N. J. Higham. The Matrix Computation Toolbox. `http://www.ma.man.ac.uk/~higham/mctoolbox`.

[13] R. A. Horn and C. R. Johnson. *Topics in matrix analysis*. Cambridge University Press, Cambridge, United Kingdom, 1991.

[14] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45:385–482, 2003.

[15] S. Lucidi and M. Sciandrone. On the global convergence of derivative-free methods for unconstrained optimization. *SIAM Journal on Optimization*, 13(1):97–116, 2002.

[16] M. Machura and A. Mulawa. Algorithm 450 Rosenbrock function minimization. *Communications of the ACM*, 16(8):482–483, 1973.

[17] M. Macklem. *Low Dimensional Curvature Methods in Derivative-free Optimization on Shared Computing Networks*. PhD thesis, Computer Science, Dalhousie University, Halifax, Nova Scotia, Canada, 2009.

[18] The MathWorks, Inc., Natick, Massachusetts, USA. *MATLAB Optimization Toolbox User's Guide*, 2010.

[19] J. J. Moré and S. M. Wild. Benchmarking derivative-free optimization algorithms. *SIAM Journal on Optimization*, 20(1):172–191, 2009.

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

[21] S. R. Pope. *Parameter Identification in Lumped Compartment Cardiorespiratory Models*. PhD thesis, North Carolina State University, Raleigh, North Carolina, USA, 2009.

[22] M. J. D. Powell. *Large-Scale nonlinear optimization*, volume 83 of *Nonconvex Optimization and its applications*, chapter The NEWUOA software for unconstrained optimization without derivatives, pages 255–297. Springer US, 2006.

[23] C. P. Price and Ph. L. Toint. Exploiting problem structure in pattern search methods for unconstrained optimization. *Optimization Methods and Software*, 21(3):479–491, 2006.

[24] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175–184, Oct. 1960.

[25] V. Torczon. *Multi-Directional Search: A Direct Search Algorithm for Parallel Machines*. PhD thesis, Department of Mathematical Sciences, Rice University, Houston, Texas, 1989. Available as Tech. Rep. 90-07, Department of Computational and Applied Mathematics, Rice University, Houston, Texas 77005-1892.

[26] P. Wolfe. Convergence conditions for ascent methods. II: Some corrections. *SIAM Review*, 13(2):185–188, 1971.