Incorporating Minimum Frobenius Norm Models in Direct Search*

A. L. Custódio[†] H. Rocha[‡] L. N. Vicente[§]

July 31, 2009

Abstract

The goal of this paper is to show that the use of minimum Frobenius norm quadratic models can improve the performance of direct-search methods. The approach taken here is to maintain the structure of directional direct-search methods, organized around a search and a poll step, and to use the set of previously evaluated points generated during a direct-search run to build the models. The minimization of the models within a trust region provides an enhanced search step. Our numerical results show that such a procedure can lead to a significant improvement of direct search for smooth, piecewise smooth, and noisy problems.

1 Introduction

Direct-search methods are a very popular class of methods for derivative-free optimization whose distinctive feature is to guide the new algorithmic actions solely based on simple comparison rules of objective function values, without any attempt to approximate derivatives or build models. With some exceptions, like the Nelder-Mead methods, most of the direct-search methods are

*Support has been provided by FCT under grants POCI/MAT/59442/2004 and PTDC/MAT/64838/2006.

[†]CMA-UNL, Department of Mathematics, FCT-UNL, Quinta da Torre, 2829-516 Caparica, Portugal (alcustodio@fct.unl.pt).

[‡]INESC Coimbra, R. Antero de Quental, 199, 3000-033 Coimbra, Portugal (hrocha@mat.uc.pt).

[§]CMUC, Department of Mathematics, University of Coimbra, 3001-454 Coimbra, Portugal (lnv@mat.uc.pt).

relatively inefficient since no attempt is made to explore curvature. Direct-search methods of directional type (coordinate search, generalized pattern search, generating set search, mesh adaptive direct search) exhibit interesting convergence properties, in particular for nonsmooth problems, and are relatively easy to code and to parallelize.

Model-based methods, in particular interpolation-based trust-region methods, have been shown to be more efficient and robust than direct search on a reasonably representative test suite of unconstrained optimization problems [10]. Although this has been known among researchers who have tested both classes of methods, the recent results in [10] for piecewise smooth and noisy problems are nevertheless relatively surprising. One of the key ingredients of these model-based implementations is the use of quadratic models computed in an underdetermined form (using fewer points than the number of basis components) but enforcing the Frobenius norm of the Hessian model (or its variation) to be as small as possible.

It is therefore natural to ask for a combination of both techniques. In this paper we describe and test what might be considered as a natural way to accomplish such a task. We form minimum Frobenius norm (MFN) models based on sample points for which the objective function has already been evaluated during the course of direct search. It is expected then that the minimization of these MFN models speeds up the direct-search run. There are, however, a number of variants in which this simple idea can be implemented. We will report the various possibilities and describe the ones which we found to be the most successful. The final numerical results show a significant improvement in efficiency for all types of problems, although the performance of the modified direct-search method is still below the one achieved by model-based methods for smooth problems, specially when considering small function evaluation budgets. Focusing on unconstrained optimization, the best version we identified considers opportunistic polling and builds the MFN models by minimizing the Hessian norm rather than its variation. A significant speed up for larger function evaluation budgets is obtained by forming and minimizing least-squares regression models. The resulting hybrid algorithm is competitive for all the different classes of problems.

The paper is organized in the following way. In Section 2 we provide a short summary of MFN models. In Section 3, we show how to incorporate these models into a directional direct-search framework. Section 4 reports our numerical experiments. Some final remarks are given in Section 5. We point out that all norms in this paper are Euclidean.

2 Minimum Frobenius norm models

Given a sample set $Y = \{y^0, y^1, \ldots, y^p\}$, a polynomial basis

$$\phi = \{\phi_0(x), \phi_1(x), \ldots, \phi_q(x)\},$$

and a polynomial model $m(y) = \alpha^\top \phi(y)$, the conditions for polynomial interpolation can be written as a system of linear equations:

$$M(\phi, Y)\alpha \; = \; f(Y), \tag{1}$$

where

$$M(\phi, Y) = \begin{bmatrix} \phi_0(y^0) & \phi_1(y^0) & \cdots & \phi_q(y^0) \\ \phi_0(y^1) & \phi_1(y^1) & \cdots & \phi_q(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_0(y^p) & \phi_1(y^p) & \cdots & \phi_q(y^p) \end{bmatrix} \text{ and } f(Y) = \begin{bmatrix} f(y^0) \\ f(y^1) \\ \vdots \\ f(y^p) \end{bmatrix}.$$

In this paper we use the natural basis of monomials which appears in Taylor models (in two dimensions we have $\phi = \{1, x_1, x_2, x_1^2/2, x_2^2/2, x_1 x_2\}$). The system (1) is determined when $p = q$, overdetermined when $p > q$, or underdetermined when $p < q$. In this latter case, we can select a solution by computing the one with the minimum ℓ_2-norm, or the one with the minimum Frobenius norm if only the size of the quadratic coefficients is considered.

When $p \geq n$, the error bounds for polynomial interpolation typically obey (see [4, Sections 3.4 and 4.4])

$$\|\nabla f(y) - \nabla m(y)\| \; \leq \; [C_p C_f \Lambda] \; \Delta \qquad \forall y \in B(x; \Delta),$$

where Y is contained in the ball $B(x; \Delta)$ centered at x and of radius Δ, C_p is a positive constant depending on p, $C_f > 0$ measures the smoothness of f (e.g., the Lipschitz constant of ∇f), and $\Lambda > 0$ is a Λ–poisedness constant related to the geometry of Y.

The original definition of Λ–poisedness [4, Sections 3.2-3.3 and 4.2-4.3] says that the maximum absolute value in $B(x; \Delta)$ of all the Lagrange polynomials is bounded by Λ. Now let the inverse or left inverse of a matrix A with full column rank be denoted by $A^\dagger$. An equivalent definition of Λ–poisedness is

$$\|M(\phi, Y_{scaled})^\dagger\| \leq \Lambda,$$

with Y_{scaled} obtained from Y such that $Y_{scaled} \subset B(0; 1)$ and one of the points in Y_{scaled} has norm one.

The underdetermined case of interest to us is quadratic polynomial interpolation, corresponding to $q = (n+1)(n+2)/2-1$ and $p < q$. It is convenient to write these quadratic models also in the form

$$m(y) \;=\; c + g^\top y + \frac{1}{2} y^\top H y.$$

It will be necessary to explore the partition of the matrix $M(\phi, Y)$ into linear and quadratic terms

$$M(\phi, Y) \;=\; \left[\; M(\phi_L, Y) \quad M(\phi_Q, Y) \;\right]$$

(in two dimensions this corresponds to $\phi_L = \{1, x_1, x_2\}$ and $\phi_Q = \{x_1^2/2, x_2^2/2, x_1 x_2\}$). Using this notation we also have $m(y) = \alpha_L^\top \phi_L(y) + \alpha_Q^\top \phi_Q(y)$. One can state that Y is Λ_L–poised for linear interpolation or regression when

$$\|M(\phi_L, Y_{scaled})^\dagger\| \leq \Lambda_L.$$

The following result provides a general error bound for underdetermined quadratic polynomial interpolation [4, Theorem 5.4].

Theorem 2.1 *Let f be a continuously differentiable function in an open set containing the ball $B(x;\Delta)$ with Lipschitz continuous gradient in $B(x;\Delta)$ (and Lipschitz constant $C_f > 0$). If Y is Λ_L–poised for linear interpolation or regression then*

$$\|\nabla f(y) - \nabla m(y)\| \;\leq\; C_p \Lambda_L \left[C_f + \|H\|\right] \Delta \qquad \forall y \in B(x;\Delta).$$

where H is the Hessian of the model and C_p is a positive constant dependent on p.

The constant multiplying Δ in this error bound is strongly dependent on the norm of the model Hessian H. Thus, it is not surprising that the minimum Frobenius norm (MFN) models are built by minimizing the entries of the Hessian (in the Frobenius norm) subject to the interpolation conditions:

$$\begin{array}{ll} \min & \frac{1}{4}\|H\|_F^2 \\ \text{s.t.} & c + g^\top (y^i) + \frac{1}{2}(y^i)^\top H (y^i) \;=\; f(y^i), \quad i = 0, \ldots, p, \end{array} \tag{2}$$

or, equivalently,

$$\begin{array}{ll} \min & \frac{1}{2}\|\alpha_Q\|^2 \\ \text{s.t.} & M(\phi, Y)\alpha \;=\; f(Y). \end{array}$$

The solution of this quadratic problem requires the solution of a linear system of equations involving the matrix

$$F(\phi, Y) \;=\; \left[\begin{array}{cc} M(\phi_Q, Y) M(\phi_Q, Y)^\top & M(\phi_L, Y) \\ M(\phi_L, Y)^\top & 0 \end{array} \right].$$

The definition of Λ_F–poisedness in the minimum Frobenius norm sense, which we investigated in our computational tests, is given by the condition

$$\|F(\phi, Y_{scaled})^{-1}\| \;\leq\; \Lambda_F.$$

These MFN models are used in the `DFO` code of Scheinberg [1], which implements an interpolation-based trust-region method. It is possible to show that for these models the Hessian is bounded [4, Theorem 5.7]:

Theorem 2.2 *Let f be a continuously differentiable function in an open set containing the ball $B(x;\Delta)$ with Lipschitz continuous gradient in $B(x;\Delta)$ (and Lipschitz constant $C_f > 0$). If Y is Λ_F–poised in the minimum Frobenius norm sense then*

$$\|H\| \;\leq\; C_{p,q} C_f \Lambda_F,$$

where H is the Hessian of the model and $C_{p,q}$ is a positive constant depending on p and q.

These two theorems together yield the following error bound for MFN models:

$$\|\nabla f(y) - \nabla m(y)\| \;\leq\; C_p \Lambda_L C_f \left[1 + C_{p,q}\Lambda_F\right] \Delta \quad \forall y \in B(x;\Delta).$$

The conclusion is that MFN models are *fully linear*, as defined in [4, Definition 6.1], reproducing well the linear order of accuracy of first-order Taylor models (i.e., of models using first-order derivatives).

An alternative suggested by Powell [12] is to minimize the difference between the current and previous Hessians (in the Frobenius norm):

$$\begin{array}{ll} \min & \frac{1}{4}\|H - H^{old}\|_F^2 \\ \text{s.t.} & c + g^\top(y^i) + \frac{1}{2}(y^i)^\top H(y^i) \;=\; f(y^i), \quad i = 0, \ldots, p. \end{array} \tag{3}$$

The resulting models are called least updating MFN models and are used in Powell's `NEWUOA` interpolation-based trust-region solver [13]. Powell provided for these models the following theoretical insight [12].

Theorem 2.3 *If f is a quadratic function with Hessian denoted by $\nabla^2 f$, then:*

$$\|H - \nabla^2 f\| \;\leq\; \|H^{old} - \nabla^2 f\|.$$

MFN models are being used by other authors in trust-region interpolation-based methods (see, e.g., [14]) but their potential in optimization is still to be fully explored.

3 Using MFN models in direct search

Direct-search methods of directional type have been extensively analyzed in the literature [4, 9]. We are interested in studying the impact of using MFN models to enhance this class of methods. As a basis for our study we selected coordinate search which has been shown to behave well for unconstrained optimization [5, 6] among other generalized pattern search methods. The poll step in coordinate search operates with the positive basis $D_k = D_\oplus = [\, I \;\; -I \,]$ as the set of poll vectors and evaluates the objective function at the points in the poll set

$$P_k \;=\; \{x_k + \alpha_k d, \;\; d \in D_k\}.$$

Polling can be opportunistic (stopping once a decrease in the value of the objective function is found) or complete (identifying the lowest of the poll points). A search step can be applied before the poll step by considering a finite number of points in the current mesh:

$$M_k \;=\; \{x_k + \alpha_k D_k z, \;\; z \in \mathbb{Z}^{|D_k|}\}.$$

If a point in M_k is identified where the objective function is lower than $f(x_k)$, then such a point becomes the new iterate, and both the search step and the iteration are considered successful. Otherwise, a poll step is then applied. The poll step may be unsuccessful (and so is the iteration) when no point in P_k provides a function value lower than $f(x_k)$. The step size or mesh size parameter $\alpha_k > 0$ is decreased at unsuccessful iterations and increased or kept constant at successful poll or search steps.

The code `SID-PSM` is a MATLAB [2] implementation of a generalized pattern search method, developed by the authors, that uses simplex derivatives (i.e., derivatives of polynomial interpolation models) in the search and poll steps. The code handles constraints (if their derivatives are provided) but this is not treated in this paper. `SID-PSM` includes different strategies to order the poll vectors (the default being the one according to the angle between the vectors and the negative simplex gradient). However, the search step in `SID-PSM` has been very crude and consisted of the minimization of a quadratic model with a diagonal model Hessian.

Our idea is to form and minimize MFN models in the search step and thus improve the performance of `SID-PSM`. The motivation is twofold. On the one hand, we know that MFN models provide very good numerical results within the `DFO` and `NEWUOA` interpolation-based trust-region codes. On the other hand, we know that direct-search methods of directional type work well for noisy/nonsmooth problems. Our goal is then to derive a hybrid method capable of being competitive with interpolation-based trust-region methods for smooth problems and perhaps better than these methods for noisy/nonsmooth problems.

We will report below the version of our algorithm for incorporating MFN models into directional direct search which provided the best results for the whole test set among a number of versions tried. Two variants of this best version are considered depending on how quadratic models are built when the number of sample points exceeds the number of basis components.

Search step (general strategy)

In the search step, one computes a quadratic model when there are more than $n+1$ points for which the objective function has been previously evaluated. Note that we consider all the points previously evaluated (`store-all` mode in the code `SID-PSM`) and not just those which lead to a decrease in the objective function value (`store-successful`). We know from [5, 6] that this is a good strategy as it tries to capture as much function information available as possible.

A search step is always attempted after a first quadratic model has been built, by minimizing the model in $B(x_k; \Delta_k)$. If no new model is formed at the current iteration, then one uses the last previously built model. In this way we try to explore as much previous model information as possible.

The maximum number of points stored for building a quadratic model is $(n+1)(n+2)$.

Search step (trust-region subproblem)

This quadratic model is then minimized in a ball (or trust region)

$$B(x_k; \Delta_k) = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$$

centered at x_k with radius

$$\Delta_k \;=\; \sigma_k \, \alpha_{k-1} \, \max_{d \in D_{k-1}} \|d\|,$$

where D_{k-1} is the set of poll vectors considered in the last iteration and σ_k takes the value 1 if the previous iteration was unsuccessful, or 2 otherwise. The value of the trust-region radius for the minimization of the quadratic models is never allowed to be smaller than 10^{-5}.

In our numerical experiments we used the function `trust.m` from MATLAB to solve the trust-region subproblems consisting of the minimization of the quadratic models in the trust regions. The `DGQT` routine from the `MINPACK2` [11] package was also tested for this purpose but we did not observe significant differences in the overall performance.

Search step (overdetermined number of points)

When the number of points stored is in $[n+2, (n+1)(n+2)/2]$, one computes a MFN model using (2). The least updating MFN models (3) performed worse in our context. (Note that when the number of points is exactly $(n+1)(n+2)/2$, we are in the presence of complete quadratic interpolation but nevertheless we still use the MFN computation scheme.)

When the number of points is in $(\,(n+1)(n+2)/2, (n+1)(n+2)\,]$, two variants have been considered as we mentioned before. In the first variant, a number of points is discarded, and a (complete) quadratic interpolation model is built with exactly $(n+1)(n+2)/2$ points. In the second variant, a regression quadratic model is computed using all the stored points by solving in α the regularized least-squares problem

$$\min \ \frac{\rho}{2}\|\alpha_Q\|^2 + \frac{1}{2}\|M(\phi, Y)\alpha - f(Y)\|^2,$$

where $\rho \geq 0$ is a regularization parameter in the Hessian part of the model. The best results for the test sets considered in this paper seemed to indicate that regularization is unnecessary and thus we report results only for $\rho = 0$.

Whenever there are more points than the number selected for building a model, and since the purpose of using models is primarily local, 80% of the desired points are selected as the ones nearest to the current iterate. The last 20% are chosen as the ones further away from the current iterate, in an attempt of preserving geometry and diversifying the information used in the model computation (as in practical interpolation-based trust-region optimization approaches). We tried a few possibilities for the percentage P of far away points and identified $P = 20$ as a good choice.

Search step (geometry control)

The geometry control of the sample set used for the model computation is extremely loose. In fact, we do not use the condition number of the matrix $F(\phi, Y_{scaled})$ (or of the matrix $M(\phi, Y_{scaled})$ in the regression variant) to decide which points to consider in the sample set.

Since the search step is optional, this is an attempt to explore all model information independently of the quality of the underlying sample sets. This resembles, in some way, the observations made in [7], in the context of the use of complete quadratic interpolation models in trust-region methods, about the use of badly poised sample sets.

Instead of controlling the condition number of $F(\phi, Y)$ (or of $M(\phi, Y)$ in the regression variant), we determine the singular value decomposition of this matrix and, before solving the corresponding system which computes the model, replace all singular values smaller than machine precision `eps` by this threshold.

Poll step (directions) and step-size parameter

The underlying direct-search method uses the coordinate-search directions (the coordinate vectors and their negatives) and the directions e and $-e$, where e stands for a vector of ones of dimension n. Including e and $-e$ allows to change all the components at once.

The step-size parameter was maintained for successful iterations (which can occur either in a search or a poll step) and halved when no decrease has been achieved at a given iteration (after unsuccessful search and poll steps).

Poll step (type and ordering)

Polling has been considered opportunistic (complete polling performs clearly worse).

The initial ordering of the poll vectors is the one given by $D = [\, e \;\; -e \;\; I \;\; -I \,]$. At each iteration, there is an attempt to order the poll vectors in D_k according to a negative simplex gradient (see [5, 6]). With this purpose the code tries to identify a subset of points, for which the objective function has been previously evaluated and inside the trust region considered, which satisfies a Λ–poisedness condition. (In this case there is no minimum value for the size of the trust region.) When this procedure fails, polling is performed cyclically in D_k, i.e., the first polling vector at the beginning of a new poll step is chosen as the one positioned in D_k right after the last one used in the previous poll step. Cyclic polling promotes rotation of the poll vectors, avoiding to insist always on the vectors stored first.

4 Numerical experiments and analysis

To compare our new version of `SID-PSM` to other algorithms we chose to work with the recently proposed data profiles [10] for derivative-free optimization. Data profiles indicate how well a solver performs, given some computational budget, to reach a specific reduction in function value, measured by

$$f(x_0) - f(x) \;\geq\; (1 - \tau)[f(x_0) - f_L],$$

where x_0 is the initial iterate and f_L is the best objective value found by all solvers tested for a specific problem. The computational budget is measured in terms of the number of function evaluations.

The test suite is also the one proposed in [10], where the test problems have been divided into four classes: smooth (53 nonlinear least squares problems obtained from CUTEr functions, with $n \in [2, 12]$); nonstochastic noisy (obtained by adding oscillatory noise to the smooth ones); piecewise smooth (as in the smooth case but using ℓ_1-norms); stochastic noisy (obtained by adding random noise to the smooth ones).

We compared the results obtained for `SID-PSM` using MFN models against three solvers: `NEWUOA` [13], a trust-region code based on interpolation by MFN models; `NMSMAX` [3] an implementation of the Nelder-Mead simplex method; and `APPSPACK` [8], an implementation of an asynchronous generating set search method mainly derived for parallel computing, but considered here in the serial mode and with a random ordering of directions. Figures 1, 2, 3, and 4 report the data profiles obtained for the four test sets, considering the two different levels of accuracy $\tau = 10^{-3}$ and $\tau = 10^{-7}$ (Figure 1: smooth problems; Figure 2: nonstochastic noisy problems; Figure 3: piecewise smooth problems; Figure 4: stochastic noisy problems).

The maximum computational budget consisted of 1500 function evaluations, as we are primarily interested in the behavior of the algorithms for budgets applicable to problems of expensive objective function evaluation. Data profiles plot the percentage of problems solved by the solvers for different values of the computational budget. The computational budget is expressed in number of points required to form a simplex set $(n+1)$, allowing the combination of problems of different dimensions in the same profile. Note that a different function of n could be chosen, but $n+1$ is natural in derivative-free optimization (since it is the minimum number of points required to form a positive basis, a simplex gradient, or a model with first-order accuracy). Note also that the number of problems solved by each solver within the prescribed level of accuracy and maximum computational budget considered can be approximately obtained from the plots multiplying the final percentage (profile value for a budget of 115) by the number of problems (53) in each test set.

For smooth problems, the best performance is obtained by `NEWUOA`. As expected, our implementation, `SID-PSM`, takes advantage over `NEWUOA` only for noisy/nonsmooth problems and for larger budgets of function evaluations. In these cases, `SID-PSM` is the best of the four solvers tested. This effect is amplified for the regression version of `SID-PSM` which becomes competitive with `NEWUOA` even for the smooth problems. Note also that the gap between the `SID-PSM` and `NEWUOA` data profiles generally reduces from the low level to the high accuracy level.

Both variants of `SID-PSM` performed better than the other two direct-search solvers (`APPSPACK` and `NMSMAX`) for a high level of accuracy ($\tau = 10^{-7}$). The regression variant of `SID-PSM` performed better than the other two direct-

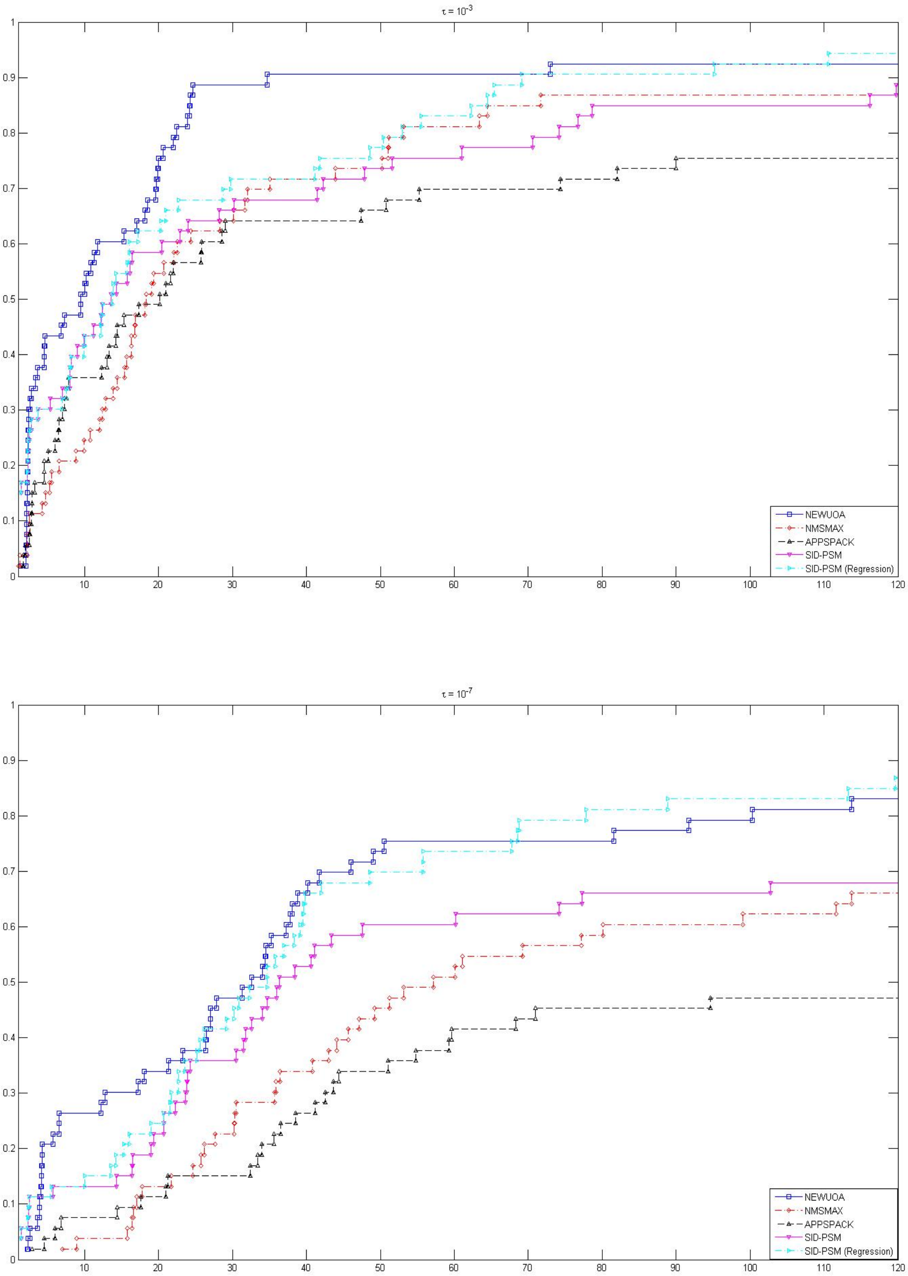

Figure 1: Data profiles computed for the set of smooth problems, considering the two levels of accuracy 10^{-3} and 10^{-7}. Each unit of the budget in the horizontal axis corresponds to $n+1$ function evaluations.

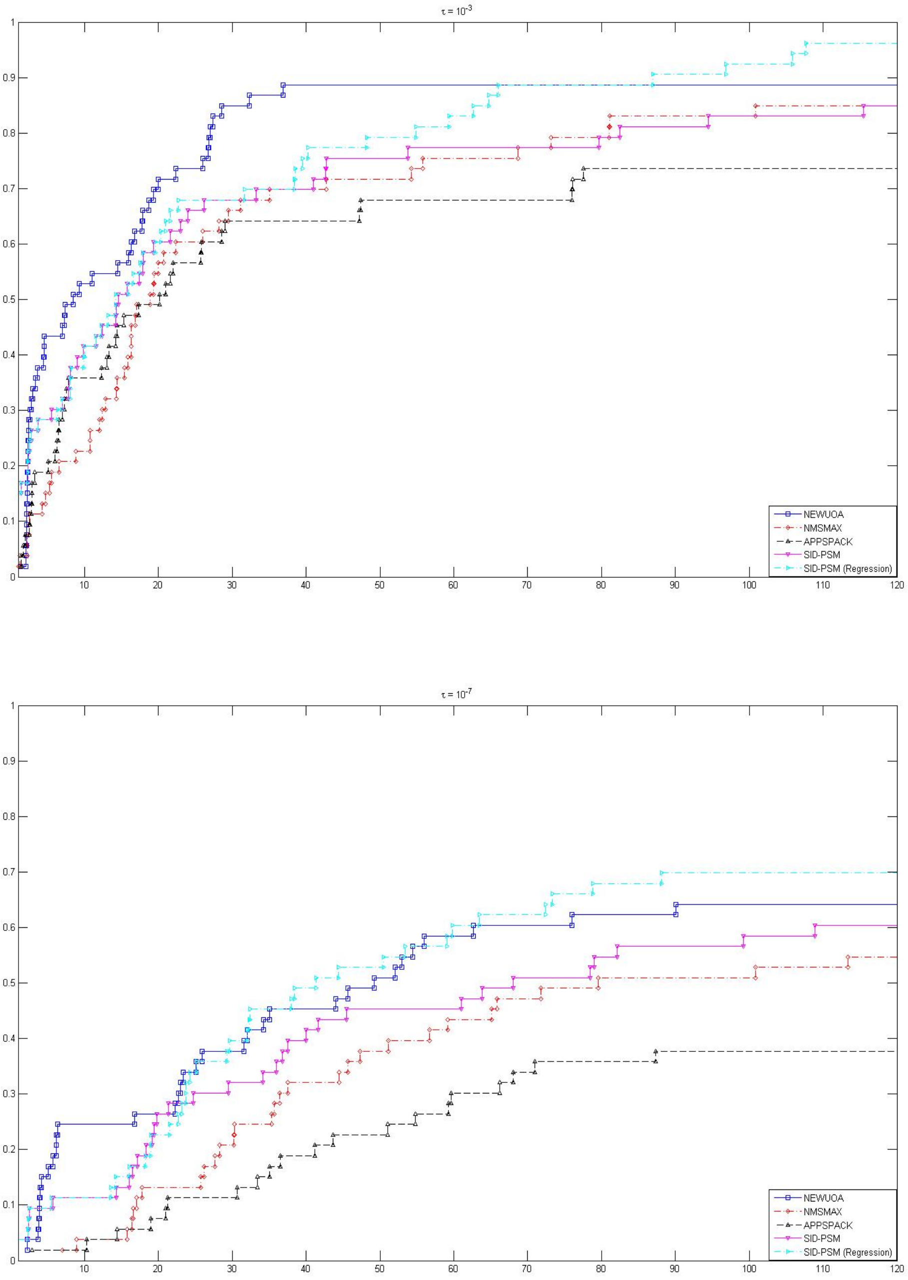

Figure 2: Data profiles computed for the set of nonstochastic noisy problems, considering the two levels of accuracy 10^{-3} and 10^{-7}. Each unit of the budget in the horizontal axis corresponds to $n+1$ function evaluations.

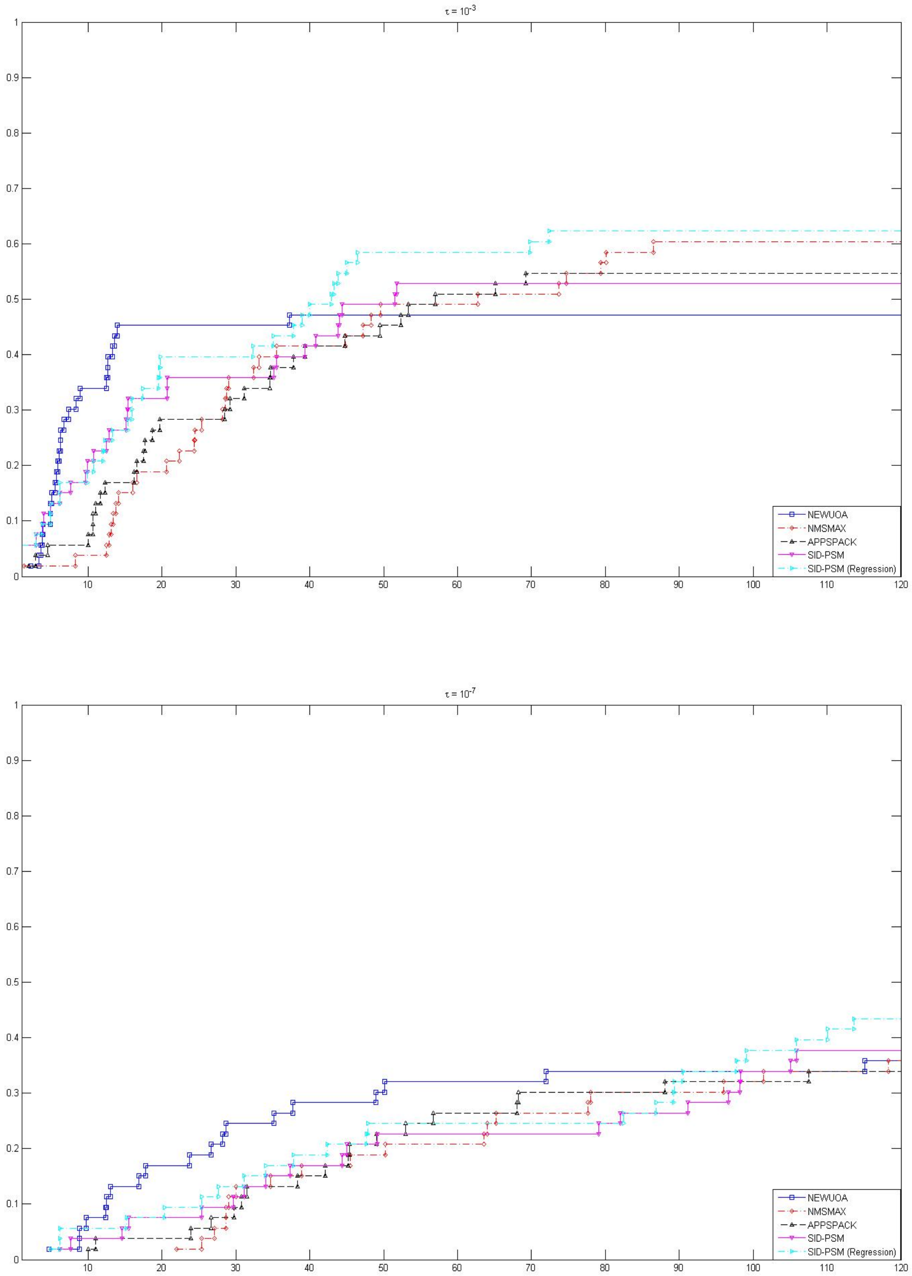

Figure 3: Data profiles computed for the set of piecewise smooth problems, considering the two levels of accuracy 10^{-3} and 10^{-7}. Each unit of the budget in the horizontal axis corresponds to $n + 1$ function evaluations.

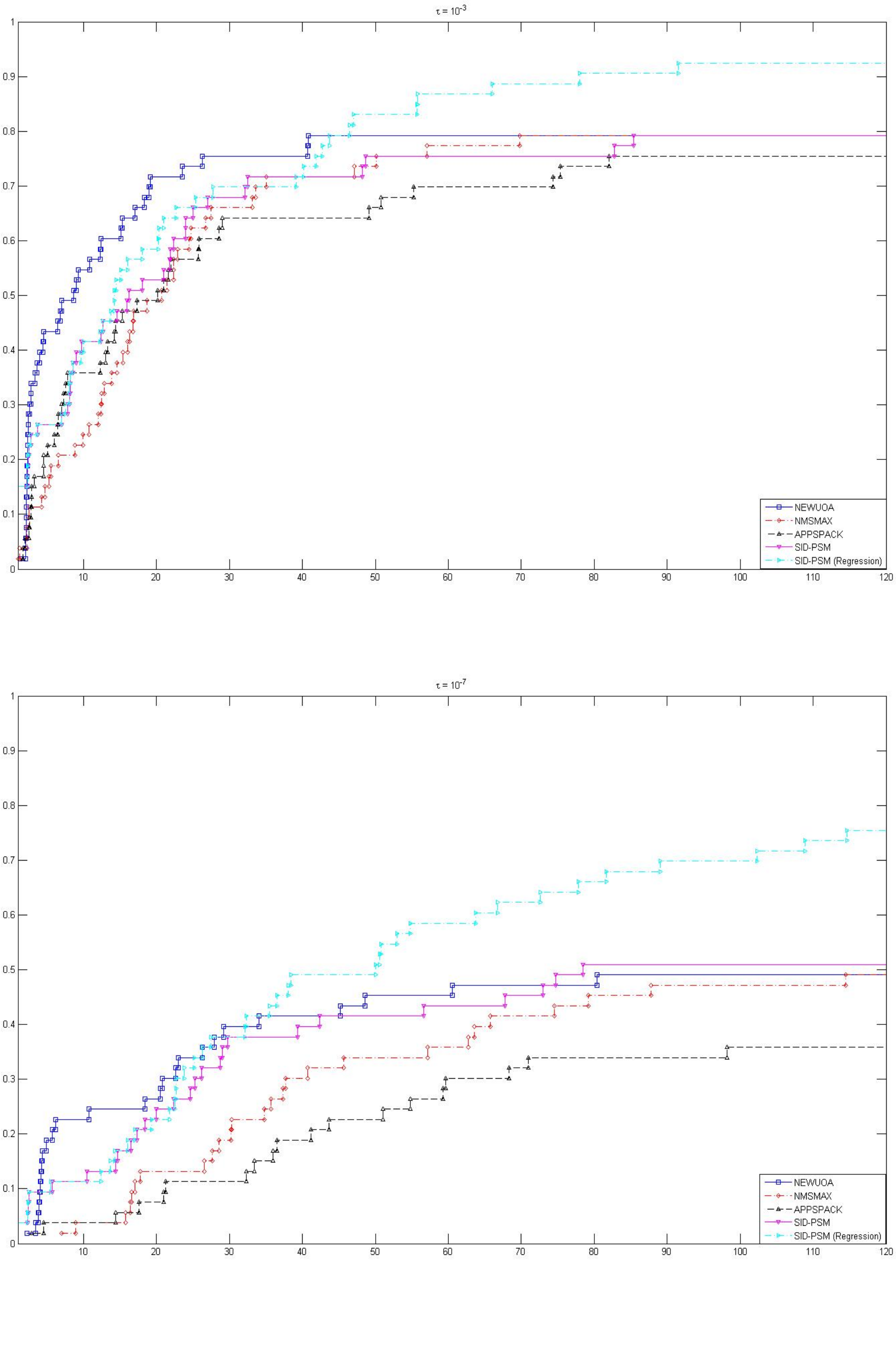

Figure 4: Data profiles computed for the set of stochastic noisy problems, considering the two levels of accuracy 10^{-3} and 10^{-7}. Each unit of the budget in the horizontal axis corresponds to $n+1$ function evaluations.

search solvers for both high and low accuracy levels.

Our conclusion from these numerical results is that the incorporation of MFN models in direct-search methods of directional type is advantageous, resulting in a superior method when compared to Nelder-Mead methods or basic directional direct-search methods. However, our implementation took also into consideration other relevant issues like the selection of poll vectors and its ordering for polling (which had a mild but non-negligible impact in the improvement of the numerical results) and the incorporation of regression in the model computation when an overdetermined number of sample points is available (which clearly improved the results for larger budgets, in particular for the noisy problems). Based on the data profiles presented and on the test problems selected, we claim that `SID-PSM` (using MFN models) is a competitive direct-search approach for unconstrained optimization, which outperforms `NEWUOA` ability to solve noisy/nonsmooth problems for larger budgets of functions evaluations.

The `SID-PSM` website is located at `http://www.mat.uc.pt/sid-psm`.

5 Concluding remarks

In this paper we reported what is to our knowledge the first attempt to enhance the performance of direct-search methods of the directional type by taking search steps based on the minimization of MFN models. Our approach consisted of using the points generated by the poll steps of these methods to build quadratic interpolation models. When the number of points was less than the number of basis components, the models were defined by the minimization of the model Hessians in the Frobenius norm. We also considered the incorporation of least-squares regression models when the number of points was above the number of basis components. The numerical results presented for the test suite chosen have shown that such procedures can improve significantly the performance of this type of direct-search methods.

The direct-search code `SID-PSM` (see above) is available to the community and it already incorporates the ideas of this paper for unconstrained optimization. `SID-PSM` can handle constraints if their derivatives are provided. A search step based on MFN models of the objective function is also provided in the constrained case although this is still subject of future research.

Acknowledgments

We would like to thank Jorge Moré (Argonne National Laboratory) and Stefan Wild (Cornell University) for providing the results of `NEWUOA` and

APPSPACK and a Matlab code for the computation of the data profiles. We are also grateful to two anonymous referees whose comments and suggestions led to improvements both in the presentation of the paper and in the performance of our code.

References

[1] *DFO*. http://www.coin-or.org/projects.html.

[2] *MATLAB, The MathWorks Inc.* http://www.mathworks.com.

[3] *The Matrix Computation Toolbox*. http://www.maths.manchester.ac.uk/~higham/mctoolbox.

[4] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to Derivative-Free Optimization*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2009.

[5] A. L. Custódio, J. E. Dennis Jr., and L. N. Vicente, *Using simplex gradients of nonsmooth functions in direct search methods*, IMA J. Numer. Anal., 28 (2008), pp. 770–784.

[6] A. L. Custódio and L. N. Vicente, *Using sampling and simplex derivatives in pattern search methods*, SIAM J. Optim., 18 (2007), pp. 537–555.

[7] G. Fasano, J. L. Morales, and J. Nocedal, *On the geometry phase in model-based algorithms for derivative-free optimization*, Optim. Methods Softw., 24 (2009), pp. 145–154.

[8] G. A. Gray and T. G. Kolda, *Algorithm 856: APPSPACK 4.0: Asynchronous parallel pattern search for derivative-free optimization*, ACM Trans. Math. Software, 32 (2006), pp. 485–507.

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

[10] J. J. Moré and S. M. Wild, *Benchmarking derivative-free optimization algorithms*, SIAM J. Optim., 20 (2009), pp. 172–191, http://www.mcs.anl.gov/~more/dfo.

[11] J. J. Moré, D. C. Sorensen, K. E. Hillstrom, and B. S. Garbow, *The MINPACK Project*, in Sources and Development of Mathematical Software, W. J. Cowell, ed., Prentice-Hall, NJ, 1984, pp. 88–111. `http://www.netlib.org/minpack`.

[12] M. J. D. Powell, *Least Frobenius norm updating of quadratic models that satisfy interpolation conditions*, Math. Program., 100 (2004), pp. 183–215.

[13] ——, *Developments of NEWUOA for minimization without derivatives*, IMA J. Numer. Anal., 28 (2008), pp. 649–664.

[14] S. M. Wild, *MNH: A derivative-free optimization algorithm using minimal norm Hessians*, in Tenth Copper Mountain Conference on Iterative Methods, April 2008.