Optim Lett (2014) 8:1599–1610
DOI 10.1007/s11590-013-0688-4

ORIGINAL PAPER

A variance-based method to rank input variables of the Mesh Adaptive Direct Search algorithm

Luc Adjengue · Charles Audet · Imen Ben Yahia

Received: 22 February 2013 / Accepted: 27 August 2013 / Published online: 8 September 2013

Abstract The Mesh Adaptive Direct Search algorithm (MADS) algorithm is designed for nonsmooth blackbox optimization problems in which the evaluation of the functions defining the problems are expensive to compute. The MADS algorithm is not designed for problems with a large number of variables. The present paper uses a statistical tool based on variance decomposition to rank the relative importance of the variables. This statistical method is then coupled with the MADS algorithm so that the optimization is performed either in the entire space of variables or in subspaces associated with statistically important variables. The resulting algorithm is called STATS-MADS and is tested on bound constrained test problems having up to 500 variables. The numerical results show a significant improvement in the objective function value after a fixed budget of function evaluations.

Keywords Direct search · Ranking of variables · Blackbox optimization

L. Adjengue
Département de Mathématiques et de Génie Industriel,
École Polytechnique de Montréal, Montreal, Canada
e-mail: Luc.Adjengue@polymtl.ca

C. Audet (✉) · I. Ben Yahia
GERAD and Département de Mathématiques et de Génie Industriel,
École Polytechnique de Montréal, Montreal, Canada
e-mail: Charles.Audet@gerad.ca

I. Ben Yahia
e-mail: Imen.ben-yahia@polymtl.ca

1 Introduction

In this paper we consider nonsmooth optimization problems of the form,

$$\min_{x \in X} f(x), \tag{1}$$

in which f is a single-valued objective function, and $X = \{x \in \mathbb{R}^n : l \leq x \leq u\}$ is the set of feasible solutions in which l and u are bound vectors, with finite or infinite components. The function defining the problem is referred to as a blackbox, and is typically obtained by a computer code. A blackbox receives input variables and returns its output with a hidden or unaccessible internal mechanism. The evaluation of a given input can be very costly and may fail to return a value for unknown reasons. This situation is formally handled by assigning the value $+\infty$ to f.

The Mesh Adaptive Direct Search (MADS) algorithm [6] is designed to handle the more general class of blackbox problems which allows blackbox constraints in addition to the bounds on the variables. MADS is supported by a hierarchical convergence analysis based on the Clarke nonsmooth calculus [14]. This method was shown to be efficient [5,6,9] on problems with a number of variables less than 20.

Derivative-free optimization includes methods that typically deal with problems where derivative estimation (for example, per finite difference [17] or automatic differentiation techniques [19]) is either very expensive, inaccurate, or even impossible, given the blackbox nature of the problem. This is the case for several industrial problems (e.g., see the recent survey [5] and the introductory textbook [16] for numerous real applications). Modelling problems may lead to problems with a large number of variables. Some optimization software packages, such as NOMAD [24], are unsuitable beyond a certain number of input variables.

In the present paper, we describe a statistical approach with similarities to the one used in [13]. That paper used analysis of variance (ANOVA) techniques to reduce the dimension of an helicopter rotor blade design problem from 31 to 11 variables, and then used a pattern search algorithm within the surrogate management framework. Other methods that use statistical analyses, such as expected improvement, include [21,23]. More recently, variable selection and sensitivity analyses using dynamic trees are studied in the context of code performance running [22].

In [8], parallelism is used to improve the efficiency of MADS. In both [13] and [8] optimization is performed in subspaces of the space of variables, while the former fixes less important variables using ANOVA, the latter randomly chooses variables, fixes them and explores subspaces in parallel using different processors.

In the present paper, we propose a different approach for problems with a large number of variables. We first launch MADS on the entire space of variables, and interrupt it after it has completed a small predetermined number of evaluations. Then, statistical tools are used to analyze the sensitivity of the objective function value with respect to each input variables. The less important ones are fixed, and an optimization process is launched in the subspace of important variables. Both these steps are then repeated until the overall budget of blackbox calls is reached. The resulting method is called STATS-MADS.

The paper is divided into three parts as follows: the next section provides a short description of the MADS derivative-free algorithm. Section 3 presents the high-level structure of our algorithm, followed by the description of our statistical tools. We then present the technical integration of the statistical tools within STATS-MADS. Section 4 tests our method on a collection of problems from the literature. Finally, Sect. 5 is devoted to concluding remarks.

2 A brief review of MADS algorithms

MADS is a direct search algorithm that encompasses the Generalized Pattern Search (GPS) algorithm [29]. It improves GPS by providing more thorough space exploration and rigourous convergence results on nonsmooth problems. Thus, it fills gaps observed in [4]. Furthermore, MADS may be applied to nonsmooth optimization problems with inequality constraints. The complete description together with its convergence analysis may be found in [6], but we present here the main ideas, sufficient for the development of STATS-MADS.

MADS is an iterative algorithm that generates some trial points on a spatial discretization called the mesh,

$$M_k \;=\; \left\{\hat{x} + \Delta_k^m D z : z \in \mathbb{N}^{|D|}, \hat{x} \in V_k\right\} \;\subseteq\; \mathbb{R}^n,$$

where $\hat{x}$ is the currently best known feasible solution by the start of iteration k, V_k is the cache (the set of points for which the blackbox has been evaluated by the start of iteration k), Δ_k^m is the mesh size parameter at iteration k and D is a set of directions (with cardinality $|D|$), fixed throughout the deployment of the algorithm. Each iteration is composed of two principal steps: the SEARCH step (which is optional and flexible as it allows evaluations of promising trial points using different search strategies) and the POLL step (which follows some rules on which the convergence analysis relies). The poll set is given by

$$P_k = \left\{\hat{x} + \Delta_k^m d : d \in D_k\right\} \subseteq M_k,$$

where D_k is the set of polling directions at iteration k, constructed by taking nonnegative integer combinations of the set of directions D. ORTHOMADS is the deterministic instance of MADS introduced in [3]. ORTHOMADS constructs the set of directions D_k in such a way to avoid large angles between poll directions (i.e. unexplored areas) and the union of all normalized polling directions grows dense in the unit sphere as the iteration k goes to infinity.

Constraints are handled using one of the following three approaches: the extreme barrier [6], which excludes all infeasible solutions, the progressive barrier [7], which tolerates trial points that violates relaxable constraints, or by the progressive to extreme barrier [9], which is a combination of the two previous approaches. ORTHOMADS with the extreme barrier for the bounds and X and the progressive barrier for the other quantifiable constraints is the default implementation in the NOMAD software [2,10,24].

3 The STATS-MADS algorithm

3.1 General description of STATS-MADS

The basic idea of STATS-MADS is to alternate optimization processes in the entire space of variables and in subspaces. Optimization in the whole space allows us to gather information on the sensitivity of the objective f with respect to each variable while still attempting to improve the best known solution. Optimization in a subspace can be seen as a local descent that exploits the fact that MADS is efficient when the dimension is small. The counter for these main iterations is denoted by ℓ. The present paper does not study the question of launching these optimization in a parallel environment.

We introduce the following notation to represent the subspaces. Let $I = \{1, 2, \ldots, n\}$ be the set of indices of the n input variables and let J_ℓ denote a subset of I corresponding to the variables tagged by a statistical analysis to be important. By stating that MADS is run on J_ℓ we mean that all the variables whose index j are not in J_ℓ are fixed to $\hat{x}_j$ (where $\hat{x}$ is the current best known solution) and MADS is only allowed to modify the variables whose indices belong to J_ℓ. It follows that the number of free variables on which the optimization is applied is equal to $|J_\ell|$.

Algorithm 1: High-level presentation of STATS-MADS

```
Initialization: Let I = {1, 2, .., n};
Main loop:
for ℓ = 0, 1, 2, ... do
    Run MADS on the whole space I;
    Find J_ℓ ⊂ I using a statistical method;
    Run MADS on the subspace J_ℓ;
end
```

Algorithm 1 summarizes this high-level description of STATS-MADS. A more detailed description is presented in Sect. 3.3.

With this framework, information about all variables is collected while exploring the whole space, and rapid descents are achieved in the subspaces.

3.2 A statistical method to rank input variables

In this subsection, we present a statistical method to establish a hierarchy of input variables. No assumptions, such as normality or independence can be made about these inputs due to the blackbox nature of the problem.

We use sensitivity analyses to study the impact of the variation of each input on the output. More precisely we propose to estimate sensitivity indicators, known as sensitivity indexes, to classify the input variables based on their importance. According to [28], the sensitivity index S_j related to input variable x_j (for some index j between 1 and n) is a measure of its influence on the output $y = f(x)$ and is defined as:

$$S_j = \frac{\mathrm{Var}(\mathrm{E}(y \mid x_j))}{\mathrm{Var}(y)}. \tag{2}$$

The numerator in (2) is the variance of the conditional expectation of y given the j-th component of x, and the denominator is the unconditional variance of y. Equation (2) corresponds to the theoretical expression of the correlation ratio η^2 in [26]. This ratio is a measure of relationship between variables x_j and y, where x_j can be qualitative or quantitative. It varies between 0 and 1 and the closer it is to 1, the more x_j influences y.

To estimate the sensitivity index S_j for a variable x_j in the context of blackbox optimization, we only make use of the fundamental variance decomposition identity of ANOVA and do not perform any formal statistical inference (i.e., the F test, multiple comparisons procedures, etc.). Therefore, none of the usual ANOVA assumptions (normality, randomization, etc.) are made, since they are neither necessary nor conceivable in this context.

Let V denote the cache of points evaluated so far, excluding points with infinite function values, and let N be its cardinality. Each element of the cache is composed of a vector $x \in \mathbb{R}^n$ whose objective function value $f(x)$ is known. Define $\bar{f} = \frac{1}{N}\sum_{x\in V} f(x)$ to be the average objective function value from the cache elements.

Now, consider the $j-$th variable x_j, for some index j between 1 and n. Let $\{v_j^i : i = 1, 2, \ldots, r_j\}$ denote the r_j distinct values that the variable x_j takes in the cache V and define $A_j^i = \{x \in V : x_j = v_j^i\}$ to be the subset of cache points whose x_j-value equals v_j^i. The values v_j^i are called *treatments* of the *factor* x_j using the ANOVA terminology. Now, for each $i = 1, 2 \ldots, r_j$ define $\bar{f}_j^i = \frac{1}{n_j^i}\sum_{x\in A_j^i} f(x)$ to be the average objective function value associated to the v_j^i, where n_j^i is the cardinality of A_j^i.

Using x_j as a factor, the fundamental equation of variance decomposition for an unbalanced one-factor ANOVA model (e.g. in [27]) yields

$$\underbrace{\sum_{x\in V}\left(f(x)-\bar{f}\right)^2}_{SS_{total}} = \underbrace{\sum_{i=1}^{r_j} n_j^i\left(\bar{f}_j^i-\bar{f}\right)^2}_{SS_{between}} + \underbrace{\sum_{i=1}^{r_j}\sum_{x\in A_j^i}\left(f(x)-\bar{f}_i\right)^2}_{SS_{within}} \tag{3}$$

where SS_{total} is the total corrected sum of squares (or total variation), $SS_{between}$ is the sum of squares due to factor x_j (i.e., between treatments) and SS_{within} is the residual sum of squares (i.e, within treatments). Equation (3) is simply a partition of total variability into its component parts.

Dividing SS_{total} and $SS_{between}$ by N produces estimates for $\mathrm{Var}(y)$, and $\mathrm{Var}(\mathrm{E}(y \mid x_j))$ from Equation (2), respectively. Consequently, our estimate of the sensitivity index for the variable x_j is

$$\hat{S}_j = \frac{SS_{between}}{SS_{total}} = \frac{\sum_{i=1}^{r_j} n_j^i\left(\bar{f}_j^i-\bar{f}\right)^2}{\sum_{x\in V}\left(f(x)-\bar{f}\right)^2}.$$

Notice that these indices are easy to compute, as they only require to keep track of the objective function values. Sorting the variables by ascending order of their index ranks them from least to most important.

3.3 Algorithmic structure of Stats-Mads

We next discuss the way that the set J_ℓ is constructed during each iteration of Algorithm 1. The user must supply a parameter: p the proportion of variables to be fixed when exploring the subspace of variables. Algorithm 2 describes the method.

Algorithm 2: Identification of most important variables

Input : $p \in]0, 1[$ the user-defined proportion of variables to be fixed;
Load the cache and retrieve the data;
Compute the sensitivity indices of each variables $\hat{S}_1, \hat{S}_2, \ldots, \hat{S}_n$;
Sort indices in ascending order $\hat{S}_{i_1} \leq \hat{S}_{i_2} \leq \ldots \leq \hat{S}_{i_n}$;
Set $J_\ell = \{i_q, i_{q+1}, \ldots, i_n\}$ where $q = \lceil p\, n \rceil$;

Stats-Mads alternates between exploration in the entire space and subspaces. Each of these runs is initiated at the current best-known solution, and terminates after that a prescribed budget of evaluations is attained. Let n^I and n^J denote the maximal number of evaluations for each exploration in the whole space and in the subspaces, respectively. This notation allows us to summarize the algorithm as follows.

3.4 Convergence analysis

The convergence analysis of Stats-Mads is based on that of Mads.The convergence analysis of Mads along with its instantiations relies on the nonsmooth Clarke calculus [14] and is presented in a hierarchical manner [6], that depends on the assumptions put forth on f and X. In the present context, X is simply a bound-constrained domain.

Algorithm 3: Pseudo code of Stats-Mads

Input : x_0 initial point;
Δ_0 initial mesh size parameter;
n^I maximal number of blackbox evaluation while optimizing in entire space;
n^J maximal number of blackbox evaluation while optimizing in subspaces;
for $\ell = 0, 1, 2, \ldots$ **do**
  Run Mads on I from x_ℓ with initial mesh size parameter Δ_ℓ and with at most n^I calls to the blackbox;
  Let x_ℓ^I denote the best solution and $\Delta_{\ell+1}$ the final mesh size parameter;
  Apply Algorithm 2 to find $J_\ell \subset I$ using a statistical method;
  Run Mads on J_ℓ from x_ℓ^I with initial mesh size parameter Δ_ℓ and with at most n^J calls to the blackbox;
  Let $x_{\ell+1}$ denote the best solution;
end

Consider the optimization problem (1). Assume that a starting point in X with a finite objective function value is provided and that all iterates produced by Mads belong to a compact set. The existence of a convergent subsequence of mesh local

optimizers on meshes that get infinitely fine is proved in [6]. Let $\hat{x}$ denote one such limit point. The main convergence result is that if f is Lipschitz near $\hat{x}$ then the Clarke's generalized derivative $f^\circ(\hat{x}, d)$ is nonnegative for every direction d in $T_X(\hat{x})$, the tangent cone to X at $\hat{x}$. A consequence of this result is that if f is strictly differentiable near $\hat{x}$ then $\hat{x}$ is a KKT stationary point. Additional convergence results for both stronger and weaker smoothness assumptions are given in [1,30], respectively.

The next theorem shows that the same convergence analyses hold for the Stats-Mads algorithm.

Theorem 1 Stats-Mads *is an instance of the* Mads *class of algorithms.*

Proof As in the analysis of convergence of Mads, we examine the behaviour of the algorithm when no global stopping criterion is involved, i.e., when the iteration counter ℓ from Algorithm 3 goes to infinity.

Consider the sequence of iterates $\{x_\ell\}$ produced by the Stats-Mads algorithm. For a given value of ℓ, Mads is run twice with Δ_ℓ as the initial mesh size. During these runs, the mesh size will never go beneath the value $\Delta_\ell^{\min} := \tau^{\max\{n^I, n^J\}\Delta_\ell}$, where $\tau > 1$ is the internal Mads parameter used to control the variation of the mesh coarseness (see [6] for its formal definition). It follows that each trial point generated by Mads on I or on J_ℓ is generated on a mesh of size $\Delta_\ell^{\min}$. Therefore, the entire Stats-Mads algorithm can be seen as a Mads algorithm, where the sequence of mesh size parameters is $\{\Delta_\ell^{\min}\}$. It follows that Stats-Mads inherits all of the Mads convergence results. □

4 Numerical results

We ran Stats-Mads on test problems studied in [15]. The two nonsmooth problems, L1HILB and MXHILB, are taken from [25], and the eight smooth problems, BROWNAL, PENALTY1, PENALTY2, POWELLSG, SROSENBER, TRIDIA, VARDIM and WOODS, are taken from [20]. These problems were specifically chosen because their number of variables can be fixed to arbitrary values. A total of 60 instances of the problems were tested, 6 instances for each of the 10 problems.

The parameters used in the reported numerical experiments are tied to the dimension n of the problems, an overall budget of $100n$ blackbox evaluations, together with a maximum of $n^I = n^J = 10n$ evaluations for each run of Mads. Additional numerical experiments for different values of n^I and n^J can be found in the master's thesis [12].

Figure 1 illustrates the typical behaviour of Stats-Mads. The figure shows the best objective function value versus the number of function calls on the tridia test problem with 500 variables. The light circles correspond to the exploration in the entire space $\mathbb{R}^n$ and the dark ones represent the descents in the reduced subspaces where 90 % of the variables are fixed. It is clear that Stats-Mads outperforms the standard Mads method and that most of the progress is obtained during the subspace exploration. Similar behaviour is observed for the other problems.

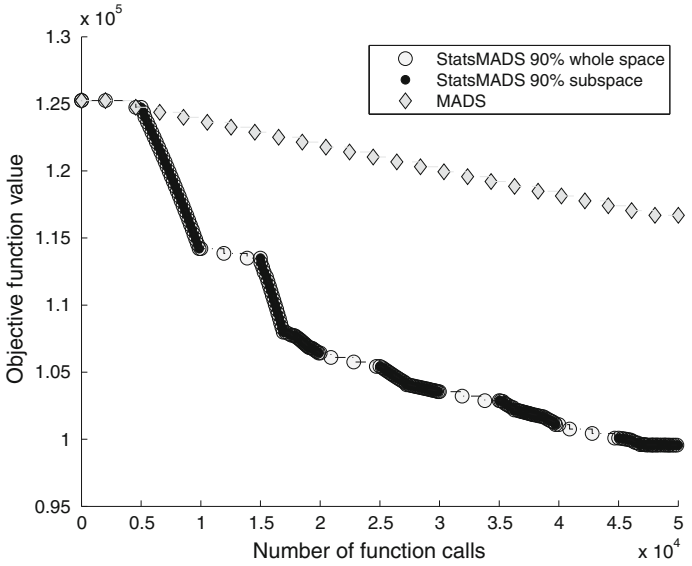

Fig. 1 Best objective function value versus number of evaluations

The plots shown in the next subsections are performance profiles, as introduced in [18]. They indicate the proportion of test problems solved within a given percentage α of the best solution found by the competing algorithmic strategies.

4.1 Stats-Mads for large-sized problems

By varying the number of variables from 250 to 500 by steps of 50, we generate a total of 60 instances. Figure 2 compares the standard Mads algorithm with Stats-Mads in which the proportion of fixed variables varies from 10 to 90 %. In fact, we have conducted tests for more values, but to make the figure readable we only reported the results for 10, 50, 80 and 90 %. More detailed results, including final objective function values, can be found in [12]. The figure shows that when the percentage is low, Stats-Mads and Mads behave in a very similar way. However, as it increases, Stats-Mads finds better solutions more rapidly.

Related work on Mads [8] exploit parallelism by assigning processors to perform descent in small subspaces. In that work, the variables defining the subspaces are randomly chosen. We compare Stats-Mads, Mads, Gps with a method where the subspaces are randomly chosen instead of being selected using the statistical analysis. Figure 3 shows the corresponding performance profile. Once again, Stats-Mads dominates other methods and, in particular, it is superior to the other strategies on 80 % of the test problems (as seen when $\alpha = 0$).

4.2 Stats-Mads for small-sized problems

Figures similar to those of the previous section are shown in [12], for problems of size 10, 20, 50 and 100. Not surprisingly, the gain in performance using Stats-Mads is

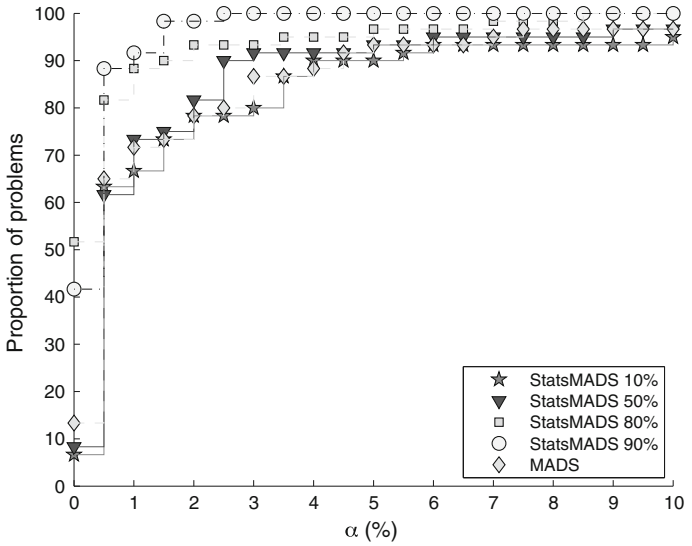

Fig. 2 Comparison of STATS-MADS and MADS on large problems

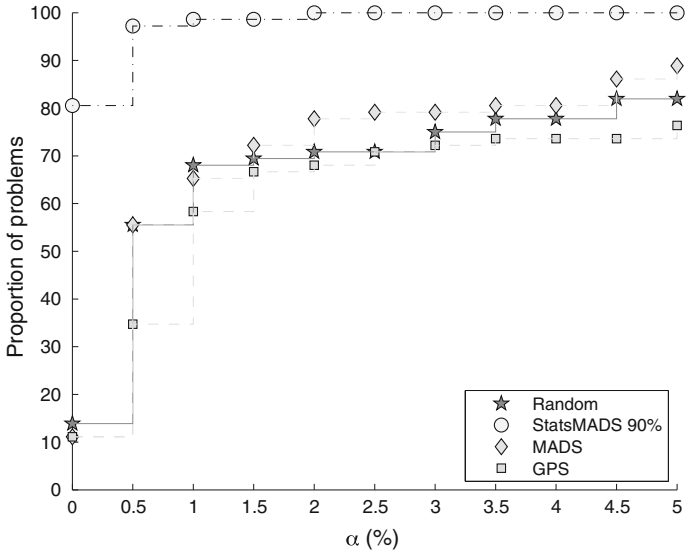

Fig. 3 Comparison of STATS-MADS, MADS, GPS and a variant with randomly chosen subspaces on large problems

more significant when the number of variables n is large. There is some benefit in using STATS-MADS when $n = 100$, but much less so than for $n = 250$. Figure 4 considers problems with $n = 20$ variables and compares MADS and GPS to STATS-MADS with 10, 50 and 90 % of the variables being fixed. In this case, there is no apparent benefit.

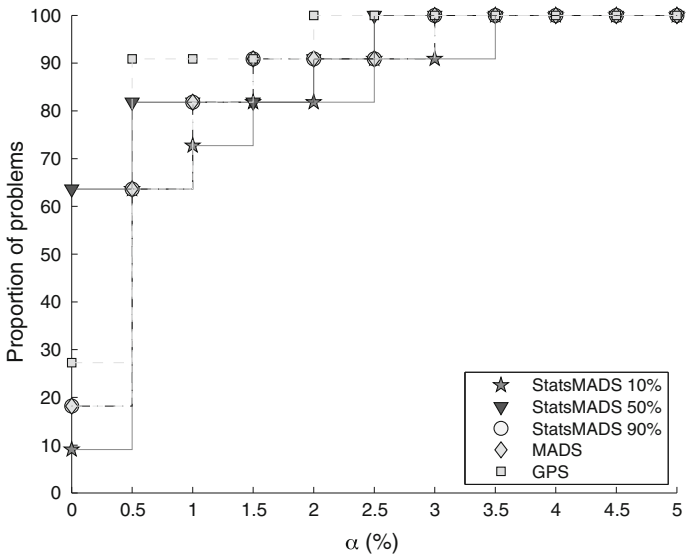

Fig. 4 Comparison between STATS-MADS, MADS and GPS on small problems

5 Discussion and future work

STATS-MADS handles large-sized problems by applying statistical analyses to identify important variables in which a sub-optimization is conducted. The approach is simple to implement, as it only needs to perform elementary operations to the cache of trial points. This approach is certainly not restricted to MADS, but it is easily implemented in the context of direct search methods. Three future research perspectives are considered.

In [8], the authors use parallelism to handle blackbox optimization problems. One possible research direction would be to replace their scheme in which they explore subspaces by the approach proposed in the present paper. Combining parallelism and statistical analyses might allow us to solve even larger problems.

The algorithm proposed in this paper is intended for bound constrained blackbox optimization. However, in many situations the domain also includes general inequality constraints. A promising research direction would be to extend the statistical analyses so that they take into account the sensitivity to the objective and to the constraints. However, we do not see an immediate extension of the present work to accomplish this in a useful way.

In very recent work, Bagirov et al. [11]1 use quasisecants to identify descent directions for nonsmooth unconstrained optimization. The convergence analysis ensures Clarke stationarity. Their method is applied to problems with up to 1,000 variables. In future work, we plan to combine statistical analyses with quasisecant directions.

Acknowledgments Work of C. Audet was supported by NSERC grant 239436 and AFOSR FA9550-12-1-0198. Research of C. Audet is partially supported by NSERC Discovery Grant 239436 and AFOSR FA9550-09-1-0160.

References

1. Abramson, M.A., Audet, C.: Convergence of Mesh Adaptive Direct Search to second-order stationary points. SIAM J. Optim. **17**(2), 606–619 (2006)
2. Abramson, M.A., Audet, C., Couture, G., Dennis, J.E. Jr., Le Digabel, S., Tribes, C.: The NOMAD project. Software available at http://www.gerad.ca/nomad
3. Abramson, M.A., Audet, C., Dennis, J.E. Jr., Le Digabel, S.: OrthoMADS: a deterministic MADS instance with orthogonal directions. SIAM J. Optim. **20**(2), 948–966 (2009)
4. Audet, C.: Convergence results for generalized pattern search algorithms are tight. Optim. Eng. **5**(2), 101–122 (2004)
5. Audet, C.: A survey on direct search methods for blackbox optimization and their applications. Technical Report G-2012-53, Les cahiers du GERAD. In: Pardalos, P.M., Rassias, Th.M. (eds.) Mathematics Without Boundaries - Surveys in Interdisciplinary Mathematics. Springer, New York, 2013 (in preparation)
6. Audet, C., Dennis, J.E. Jr.: Mesh adaptive direct search algorithms for constrained optimization. SIAM J. Optim. **17**(1), 188–217 (2006)
7. Audet, C., Dennis, J.E. Jr.: A progressive barrier for derivative-free nonlinear programming. SIAM J. Optim. **20**(1), 445–472 (2009)
8. Audet, C., Dennis, J.E. Jr., Le Digabel, S.: Parallel space decomposition of the mesh adaptive direct search algorithm. SIAM J. Optim. **19**(3), 1150–1170 (2008)
9. Audet, C., Dennis, J.E. Jr., Le Digabel, S.: Globalization strategies for mesh adaptive direct search. Comput. Optim. Appl. **46**(2), 193–215 (2010)
10. Audet, C., Le Digabel, S., Tribes, C.: NOMAD user guide. Technical Report G-2009-37, Les cahiers du GERAD (2009)
11. Bagirov, A.M., Jin, L., Karmitsa, N., Al Nuaimat, A., Sultanova, N.: Subgradient method for nonconvex nonsmooth optimization. J. Optim. Theory Appl. **157**(2), 416–435 (2013)
12. Ben Yahia, I.: Identification statistique de variables importantes pour l'optimisation de boîtes noires. Master's thesis, École Polytechnique de Montréal (2012)
13. Booker, A.J., Dennis, J.E. Jr., Frank, P.D., Serafini, D.B., Torczon, V.: Optimization using surrogate objectives on a helicopter test example. In: Borggaard, J., Burns, J., Cliff, E., Schreck, S. (eds.) Optimal Design and Control. Progress in Systems and Control Theory, pp 49–58. Birkhäuser, Cambridge, Massachusetts (1998)
14. Clarke, F.H.: Optimization and Nonsmooth Analysis. Wiley, New York (1990) Reissued In: Series Classics in Applied Mathematics, vol. 5. SIAM Publications, Philadelphia (1983)
15. Conn, A.R., Le Digabel, S.: Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. Optim. Methods Softw. **28**(1), 139–158 (2013)
16. Conn, A.R., Scheinberg, K., Vicente, L.N.: Introduction to derivative-free optimization. In: MOS/SIAM Series on Optimization. SIAM, Philadelphia (2009)
17. Dennis, J.E. Jr., Schnabel, R.B.: Numerical methods for unconstrained optimization and nonlinear equations. Prentice-Hall, Englewood Cliffs, NJ (1983). Reissued In: Series Classics in Applied Mathematics, vol. 16. SIAM Publications, Philadelphia (1996)
18. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Progr. **91**(2), 201–213 (2002)
19. Gilbert, J.C.: Automatic differentiation and iterative processes. Optim. Methods Softw. **1**(1), 13–21 (1992)
20. Gould, N.I.M., Orban, D., Toint, PhL.: CUTEr (and SifDec): a constrained and unconstrained testing environment, revisited. ACM Trans. Math. Softw. **29**(4), 373–394 (2003)
21. Gramacy, R.B., Le Digabel S.: The mesh adaptive direct search algorithm with treed Gaussian process surrogates. Technical Report G-2011-37, Les cahiers du GERAD (2011)
22. Gramacy, R.B., Taddy, M.A., Wild, S.M.: Variable selection and sensitivity analysis via dynamic trees with an application to computer code performance tuning. Ann. Appl. Stat. **7**(1), 51–80 (2013)
23. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black box functions. J. Glob. Optim. **13**(4), 455–492 (1998)
24. Le Digabel, S.: Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm. ACM Trans. Math. Softw. **3**7(4), 44:1–44:15 (2011)
25. Lukšan, L., Vlček, J.: Test problems for nonsmooth unconstrained and linearly constrained optimization. Technical Report V-798, ICS AS CR (2000)

26. McKay, M.: Nonparametric variance-based methods of assessing uncertainty importance. Reliab. Eng. Syst. Saf. **57**, 267–279 (1997)
27. Montgomery, D.C.: Design and Analysis of Experiments. Wiley, New York (2009)
28. Saltelli, A., Chan, K., Scott, E.M.: Sensitivity Analysis. Wiley, New York (2000)
29. Torczon, V.: On the convergence of pattern search algorithms. SIAM J. Optim. **7**(1), 1–25 (1997)
30. Vicente, L.N., Custódio, A.L.: Analysis of direct searches for discontinuous functions. Math. Progr. **133**(1–2), 299–325 (2012)