Les Cahiers du GERAD ISSN: 0711–2440

Robust optimization of noisy blackbox problems using the Mesh Adaptive Direct Search algorithm

C. Audet, A. Ihaddadene
S. Le Digabel, C. Tribes

G–2016–55

July 2016

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Robust optimization of noisy blackbox problems using the Mesh Adaptive Direct Search algorithm

Charles Audet
Amina Ihaddadene
Sébastien Le Digabel
Christophe Tribes

GERAD & Département de mathématiques et génie industriel, Polytechnique Montréal, Montréal (Québec) Canada H3C 3A7

Charles.Audet@gerad.ca
amina.ihaddadene@polymtl.ca
sebastien.le.digabel@gerad.ca
christophe.tribes@polymtl.ca

July 2016

Les Cahiers du GERAD
G–2016–55

Abstract: Blackbox optimization problems are often contaminated with numerical noise, and direct search methods such as the Mesh Adaptive Direct Search (MADS) algorithm may get stuck at solutions artificially created by the noise. We propose a way to smooth out the objective function of an unconstrained problem using previously evaluated function evaluations, rather than resampling points. The new algorithm, called Robust-MADS is applied to noisy problems from the literature.

Keywords: Robust optimization, direct search, blackbox optimization, MADS

Résumé : Les problèmes d'optimisation de boîtes noires sont souvent contaminés par du bruit numérique, et les méthodes de recherche directe telles que l'algorithme de recherche directe sur treillis adaptatif (MADS) peuvent rester bloquées dans des solutions créées artificiellement par le bruit. Ce travail propose un moyen de lisser la fonction objectif d'un problème sans contraintes en utilisant les évaluations déjà effectuées plutôt que de générer de nouveaux points par échantillonnage. Le nouvel algorithme, appelé Robust-MADS est testé sur des problèmes bruités de la littérature.

Mots clés : Optimisation robuste, recherche directe, optimisation de boîtes noires, algorithme MADS.

Acknowledgments: This work was supported in part by FRQNT grant 2015-PR-182098 and NSERC grant RDCPJ 490744-15 in collaboration with Rio Tinto and Hydro-Québec.

1 Optimization with noisy functions

This work studies the unconstrained optimization problem

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

in which the objective function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is computed by a blackbox simulation, contaminated with deterministic or stochastic noise. An objective function with a deterministic noise component may be written as:

$$f(x) = (1 + \phi(x))\, z(x) \tag{2}$$

where $z : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is a smoother function, but is unknown in practice, and $\phi : \mathbb{R}^n \to \mathbb{R}$ is a noise function that depends on x. In stochastic problems, ϕ is a random variable whose distribution may be independent of x or not.

There exist several robust optimization algorithms aiming to approach a minimizer of $z(x)$ when only having access to the noisy function values $f(x)$. In the algorithms proposed in [6, 11, 16, 22] quadratic models are constructed with regression techniques using previous evaluations of the noisy function values. The models are built on a trust region and weighted according to the evaluation uncertainty [6]. Repeated function evaluations reduce the quadratic model variance and increase the accuracy of the model parameters [22]. Several strategies for minimizing the quadratic model have been tested in previous work: a trust region algorithm in [7, 16], a modified version of the DIRECT algorithm in [14, 20] or a variant of the BOBYQA algorithm [22] to solve the stochastic problems [9]. Another technique for stochastic system optimization is described in [25], in which a ranking and selection strategy is used to determine the number of replicated responses to produce a sampling mean that satisfies a probability of correct selection. On the one hand, this procedure allows to determine the size of the sample which ensures obtaining a good estimation of the objective value. On the other hand, this classification allows to select the point having the minimum objective value. Furthermore, in [27] the smoothing function is constructed by elimination of all local optimal solutions worse than the best solution.

In [19], a smoothing technique approximates f by a smooth function $\widetilde{f}(x)$, constructed by the convolution with the probability density function v of the random variable that represents the noise:

$$\widetilde{f}(x) = \int_{\mathbb{R}^n} f(x+u)v(u)du.$$

An application of a smoothing technique on the energy function of a molecular model is described in [24]. The idea is to replace the value of the function at each point by a weighted average of nearby values. A Gaussian distribution is used to determine the weights:

$$\widetilde{f}_s(x) = \int_{\mathbb{R}^n} H(f(u), s) \exp\left(\frac{-\|x-u\|^2}{\sigma^2}\right) du,$$

where the standard deviation σ and s are the smoothing parameters, H is the function used to make the function f integrable, by using the energetic system transformation [28] based on the average value of f, or it can be also obtained by the diffusion equation method [15]. The motivation is to assign a low weight for values of f whose corresponding points are distant from the point x, and a high weight to values whose points are close to x.

In practice only a finite set of function values f is available for smoothing in blackbox optimization. Hence, the function $\widetilde{f}$ can be approximated by a discrete weighted sum of a set of observations. The function smoothing considered in this work is commonly known as a kernel regression. It is also used in [10] to estimate the convolution product.

The present work takes advantage of the function evaluations required by the optimization process to dynamically smooth out the objective function. The document is structured as follows. Section 2 proposes an algorithmic approach called Robust-MADS that dynamically smoothes the objective function by using newly obtained function evaluations. Section 3 proposes a convergence analysis of Robust-MADS and illustrates its performance on a collection of test problems from the literature.

2 Algorithmic approach

Our algorithm is designed to perform a robust optimization of Problem (1) by smoothing the noisy objective function. The approach is named Robust-MADS because it is embedded into the Mesh Adaptive Direct Search (MADS) blackbox optimization algorithm.

2.1 Mesh Adaptive Direct Search

MADS [3] is an iterative algorithm designed for a class of problems including unconstrained optimization of the form (1). The algorithm generates a sequence of trial points at which the objective function is evaluated. Each trial point is located on a mesh, a discretization of the space of variables, whose coarseness varies as the algorithm unfolds. The mesh becomes coarser when a better solution is found, and gets refined when local polling around the current best point fails to produce a better solution. The typical presentation of the MADS algorithm involves an iteration counter k that is incremented every time the mesh size parameter is updated.

The present work does not go into the details of how the trial points are generated, and simply considers the sequence of candidate points at which the objective function is evaluated. The reader is invited to consult [1, 5, 23, 26] for a thorough description of the way MADS may generate trial points.

Let $x^0 \in \mathbb{R}^n$ be the initial point, and denote x^ℓ the ℓ-th candidate point at which f is evaluated. In addition, $V^\ell = \{x^0, x^1, \ldots, x^\ell\}$ denotes the set of all trial points up to x^ℓ for which f has been successfully evaluated. Later, we will use the information $(V^\ell, f(V^\ell))$ to construct a smoothed version of the function f. For a given value of ℓ, we call $x^\ell_{\text{best}} \in \operatorname{argmin}\{f(v) : v \in V^\ell\}$ the *incumbent* solution, i.e., the best known solution. In the unlikely case that there are more than one candidate for the incumbent solution, we select the one that is the closest to the previous incumbent $x^{\ell-1}_{\text{best}}$.

The mesh M^k is updated at each iteration and is defined as follows. Let k be an iteration number, and let ℓ be the number of evaluations of f done by the start of iteration k. The mesh is

$$M^k \quad = \quad V^\ell + \left\{\delta^k D z \ : \ z \in \mathbb{Z}^n\right\} \tag{3}$$

where D is a matrix whose columns form a positive spanning set, and $\delta^k \in \mathbb{R}^n_+$ is called the *mesh size* parameter. At iteration k, trial points are generated during two separate steps, one global and one local, called the *search* and the *poll*. While search trial points can be constructed using any technique as long as they remain on the mesh, poll trial points are selected in a region around the incumbent x^ℓ_{best}, at a distance of at most Δ^k: $\{x \in M^k : \|x - x^\ell_{\text{best}}\|_\infty \leq \Delta^k\}$. The *poll size parameter* Δ^k is such that $\Delta^k \geq \delta^k$.

Typical values of the MADS parameters are to set the columns of $D \in \mathbb{R}^{n\times 2n}$ to be the positive and negative coordinate directions. When either the search or poll step is successful at improving the incumbent solution, the poll size parameter is doubled: $\Delta^{k+1} = 2\Delta^k$, and when the iteration is unsuccessful, the poll size parameter is halved: $\Delta^{k+1} = \frac{1}{2}\Delta^k$. The mesh size parameter is set to $\delta^{k+1} = \min\{(\Delta^{k+1})^2, \Delta^{k+1}\}$. This way, the mesh size parameter will go to zero much faster than the poll size parameter.

In Robust-MADS we introduce a smoothed function $\widetilde{f}$ to identify the incumbent solutions whereas MADS uses the original function f.

2.2 Function smoothing

We define the function $\widetilde{f}^\ell : \mathbb{R}^n \mapsto \mathbb{R}$ that uses the values $(V^\ell, f(V^\ell))$ to smooth f using a kernel function. For any $x \in \mathbb{R}^n$, we set

$$\widetilde{f}^\ell(x) \ = \ \frac{1}{P^\ell(x)} \sum_{v\in V^\ell} \psi(x,v) f(v) \quad \text{where} \quad P^\ell(x) \ = \ \sum_{v\in V^\ell} \psi(x,v) \tag{4}$$

and ψ is a kernel function.There are many kernel functions, for example, the Epanechnikov [12] or Gaussian functions. Here ψ is chosen to be a Gaussian function for a random variable with average $\mu = 0$ and

variance σ^2. The kernel function is given by:

$$\psi(x,v) \;=\; \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-\|x-v\|^2}{2\sigma^2}} \tag{5}$$

where $\|x-v\|$ is the Euclidean distance between x and v. In this work, the smoothing is dynamically adapted by setting the standard deviation as a multiple of the poll size parameter when the trial point x is generated at iteration k: $\sigma(x) = \beta\Delta^k$, where $\beta > 0$ is a fixed constant. Hence, $\psi(x,v)$ and $\psi(v,x)$ have the same value only if x and v have been produced with the same poll size Δ^k. Tying the standard deviation to the poll size ensures that as the algorithm unfolds and Δ^k gets small, more weight will be given to the neighbors of new trial points.

2.3 Updating the smoothed function

After evaluating f at a new trial point x and adding it to the set V^ℓ, the smoothed function $\widetilde{f}^\ell(x)$ is evaluated using (4) with a standard deviation value $\sigma(x)$. The value $\sigma(x)$ remains unchanged at ulterior iterations. The smoothed function needs to be recalculated for all points in $V^{\ell-1}$ when new information is available and the smoothed function quality increases.

Proposition 1 *Having a newly evaluated trial point x, the function $\widetilde{f}^\ell$ with $\ell > 1$ is calculated for each point $v \in V^{\ell-1}$ as follows:*

$$\widetilde{f}^\ell(v) \;=\; \frac{1}{P^\ell(v)} \left(P^{\ell-1}(v)\widetilde{f}^{\ell-1}(v) + \psi(v,x)f(x) \right) \tag{6}$$

where $P^\ell(v) \;=\; P^{\ell-1}(v) + \psi(v,x)$.

Proof. For each point $v \in V^{\ell-1} \;=\; V^\ell \setminus \{x\}$,

$$\widetilde{f}^\ell(v) \;=\; \frac{1}{P^\ell(v)} \sum_{w \in V^\ell} \psi(v,w)f(w) \;=\; \frac{1}{P^\ell(v)} \left(\sum_{w \in V^\ell \setminus \{x\}} \psi(v,w)f(w) + \psi(v,x)f(x) \right).$$

The function $\widetilde{f}^{\ell-1}$ has already been evaluated at v, and substituting the sum by $P^{\ell-1}(v)\widetilde{f}^{\ell-1}(v)$ in the previous equality gives (6). The quantity $P^\ell(v)$ is obtained by adding the weight corresponding to (v,x) to the previous weight $P^{\ell-1}(v)$. □

The above proposition holds because the value of σ for the kernel function $\psi(v,w)$ is obtained from the the poll size at the creation of v and is constant. In a blackbox optimization context, this updating technique reduces the computational effort by storing the values $\widetilde{f}^{\ell-1}$ and $P^{\ell-1}$ in a cache.

2.4 The Robust-MADS algorithm

The Robust-MADS algorithm proceeds as follows. At each iteration, denoted by k, the standard MADS algorithm generates a finite list X^k of search and poll trial points located on the mesh M^k. Then, the blackbox function f is evaluated at trial points in X^k. Due to hidden constraints [8, 18], the evaluation of f may fail, and the corresponding point is simply discarded. When the evaluation succeeds, the smoothed function $\tilde{f}^\ell$ is constructed and evaluated at each point of the set V^ℓ and the incumbent solution x^ℓ_{best} is determined. If the incumbent coincides with the last point where f was evaluated, then the iteration ends and is called successful. The poll and mesh size parameters are increased. Otherwise, the situation where the incumbent x^ℓ_{best} differs from $x^{\ell-1}_{\text{best}}$ is called a cache success because a previously evaluated trial points becomes the incumbent solution. This is due to the new smoothing function. The iteration is terminated, and the poll and mesh size parameters remain at their previous values. In both cases of success, all trial points in X^k may not be evaluated at iteration k. This is referred to as an *opportunistic strategy.* Finally, if $x^\ell_{\text{best}} = x^{\ell-1}_{\text{best}}$ then the algorithm proceeds to the next trial point. Algorithm 1 shows the pseudo-code of Robust-MADS. Figure 1 illustrates the $\tilde{f}^\ell(V^\ell)$ evaluation for a given $x \in X^k$.

Algorithm 1 Pseudocode of the Robust-MADS algorithm

procedure INITIALIZATION
  Select MADS parameters and β for smoothing
  Select an initial poll size $\Delta^0 > 0$, an initial point x^0 in $\mathbb{R}^n$ and evaluate f at x^0
  Set $\ell = 0$, $k = 0$, $x^\ell_{\text{best}} \leftarrow x^0$ and $V^0 \leftarrow \{x^0\}$
end procedure

procedure ROBUSTMADS
  repeat
    Prepare MADS iteration k: select the set X^k of search and poll trial points
    Set the iteration success $flag^k$ =FALSE
    Set the standard deviation smoothing parameter: $\sigma = \beta\Delta^k$
    for all $x \in X^k \setminus V_\ell$ **do**
      if f successfully evaluates at x **then**
        Increase evaluation counter $\ell \leftarrow \ell + 1$ and set $V^\ell \leftarrow V^{\ell-1} \cup \{x\}$
        Calculate $\widetilde{f}^\ell(v)$ and $P^\ell(v)$ using (4) or (6) for each $v \in V_\ell$
        Choose $x^\ell_{\text{best}} \in \text{argmin}\{\widetilde{f}^\ell(v) : v \in V^\ell\}$
        if $x = x^\ell_{\text{best}}$ **then**
          Iteration success: increase the MADS poll size: $\Delta^{k+1} > \Delta^k$
          Reset the iteration success $flag^k$ =TRUE
          Break **for all** loop on x
        else if $x^\ell_{\text{best}} \neq x^{\ell-1}_{\text{best}}$ **then**
          Cache update success: MADS poll size unchanged: $\Delta^{k+1} = \Delta^k$
          Reset the iteration success $flag^k$ =TRUE
          Break **for all** loop on x
        end if
      end if
    end for
    if $flag^k$ = FALSE **then**
      MADS iteration failure: decrease the poll size: $\Delta^{k+1} < \Delta^k$
    end if
    Increase the MADS iteration counter $k \leftarrow k + 1$
  until meeting a MADS termination criterion
end procedure

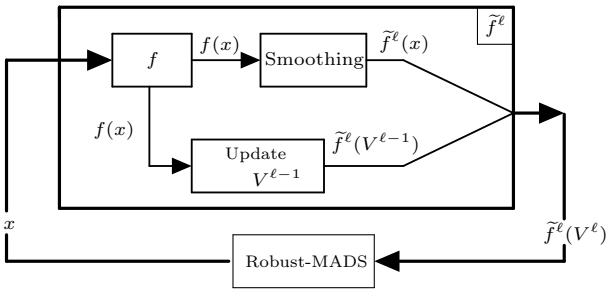


Figure 1: Function smoothing of a new point in Robust-MADS.

3 Convergence analysis and computational study

3.1 Convergence analysis of Robust-MADS

The convergence analysis of Robust-MADS is similar to that of MADS. The cornerstone of the analysis states that if the set of trial points generated by a MADS instance belongs to a bounded set, then the poll and mesh size parameters satisfy

$$\liminf_{k\to\infty} \delta^k \;=\; \liminf_{k\to\infty} \Delta^k = 0.$$

The requirement that the trial points belong to a bounded set is satisfied in particular when the level sets of f are bounded. The proof of the result relies on the fact that the mesh M^k contains finitely many points inside the bounded set. Consequently, the result trivially holds when MADS is replaced by Robust-MADS.

The mesh and poll size parameters are only reduced at unsuccessful iterations, i.e., when the incumbent solution remains unchanged after completing an iteration. The incumbent is said to be a mesh local optimizer.

Definition 1 *A convergent subsequence of mesh local optimizers, $\{x^k\}_{k\in K}$ (for some subset of indices K), is said to be a refining subsequence if $\{\Delta^k\}_{k\in K}$ converges to zero.*

It is shown in [2] that if the trial points generated by a MADS instance are in a bounded set, then refining subsequences exist. Consequently, the following corollary holds.

Corollary 1 *If $\hat{x}$ is the limit of a refining subsequence, then $\hat{x}$ is the limit of mesh local optimizers on meshes that get infinitely fine.*

The stronger convergence results of MADS do not hold in the context of Robust-MADS because of the fact that $\tilde{f}^\ell$ is redefined every time the noisy function f is evaluated.

3.2 Computational study

We analyze the performance of Robust-MADS on a collection of stochastic and deterministic noisy problems artificially created from smooth unconstrained problems from the derivative-free optimization literature [21] in the form of Equation (2). We study the effect of different values of the kernel function standard deviation factor β with respect to noise level.

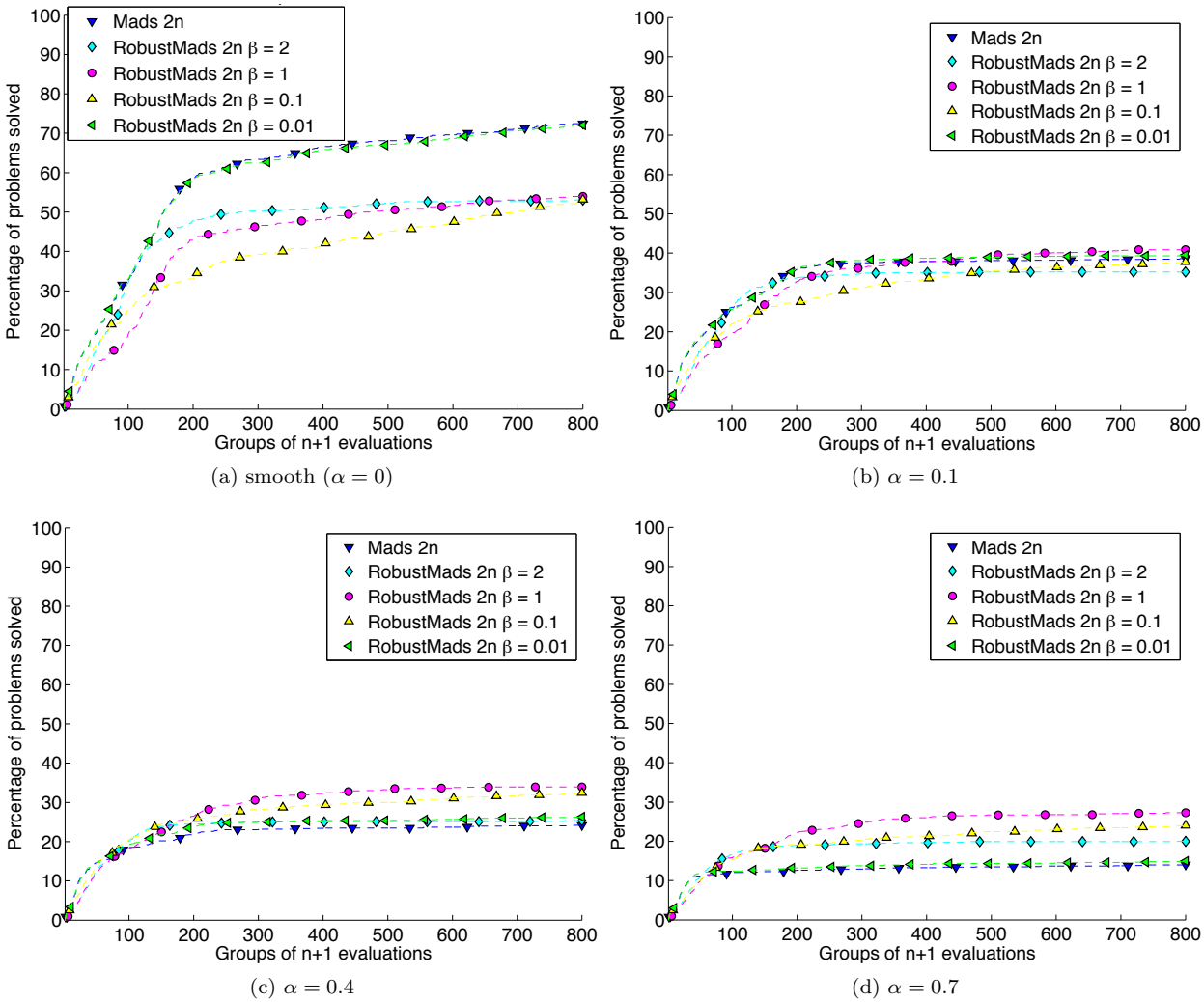
Both Robust-MADS and MADS are implemented in NOMAD [17] version 3.7.3 using the OrthoMADS $2n$ directions of MADS with quadratic models disabled. The stopping criteria of NOMAD are set to be a maximal number of evaluations and a minimal mesh size parameter value of 10^{-13}.

The analytical problems are generated from 22 different CUTEst [13] functions with different starting points giving a total of 53 smooth problems with the dimension n ranging from 2 to 12. Note that for 8 problems, some variables are constrained by a lower bound of zero. For these, infeasible trial points are simply rejected. We consider deterministic and random noisy variants of these functions in the form (2). A parameter $\alpha \in \mathbb{R}$ controls the noise amplitude: $\phi(x) = \alpha\psi(x)$. We study the influence of the noise level on the performance of the algorithm by considering other values than the default value ($\alpha = 0.001$) given in [21]. On the low end side of the noise level we obtain smooth functions $f(x) = z(x)$ for $\alpha = 0$. For the deterministic noise, above $\alpha \simeq 0.8$ the noise amplitude is so large that it dominates the variations of $f(x)$ due to $z(x)$ and NOMAD stops near the starting point. For this reason, we consider that $\alpha = 0.7$ corresponds to the strongest noise level worth studying.

Ten optimization runs are reported for each function with different pseudo-random number generator seeds. Having a large enough set of problems allows the presentation of results with *data profiles* [21] using the following convergence test with respect to the smooth objective function: $z(x^0) - z(x^e) \geq (1-\tau)\left(z(x^0) - z^*\right)$, where x^0 denotes the starting point, x^e corresponds to the best point obtained after e evaluations on one of the problems for all the compared methods, τ is the convergence tolerance, and z^* is the smooth objective function value of the best solution obtained by all the methods for all the runs of the considered problem.

In the figures, the horizontal axis is the number of evaluations divided by $n+1$, and the vertical axis corresponds to the proportion of problems solved according to a given τ. Each variant of MADS corresponds to one curve, so that graphic comparison of the performance is easy.

The data profiles of Figure 2 show that for smooth functions or for noiseless problems ($\alpha = 0$), MADS performs better than Robust-MADS, as expected. However, Robust-MADS is less sensitive to the noise level, and for $\alpha > 0.1$ it becomes more effective than MADS. The parameter β that controls the smoothing intensity impacts the performance of Robust-MADS. A small value of $\beta = 0.01$ makes Robust-MADS similar to MADS. Based on these preliminary results, $\beta = 1$ seems to be an adequate compromise.



(a) smooth ($\alpha = 0$)

(b) $\alpha = 0.1$

(c) $\alpha = 0.4$

(d) $\alpha = 0.7$

Figure 2: Data profiles obtained with MADS and Robust-MADS for a convergence tolerance of $\tau = 10^{-5}$ with test problems of various noise levels ($\alpha = 0$ is smooth and $\alpha = 0.7$ is the noisiest).

4 Discussion

We have proposed a straightforward way to smooth out an objective function by means of a weighted average involving a kernel function parameterized by the current poll size parameter. This means that as the algorithm generates points that converge to some region in the space of variables, the objective function is sampled more and more in that region, thereby increasing the number of sampling points. There are no additional mechanism to resample extra points. Future work will focus on extending this approach to constrained blackbox optimization, using the progressive barrier [4].

References

[1] M.A. Abramson, C. Audet, J.E. Dennis, Jr., and S. Le Digabel. OrthoMADS: A Deterministic MADS Instance with Orthogonal Directions. SIAM Journal on Optimization, 20(2):948–966, 2009.

[2] C. Audet and J.E. Dennis, Jr. Analysis of generalized pattern searches. SIAM Journal on Optimization, 13(3):889–903, 2003.

[3] C. Audet and J.E. Dennis, Jr. Mesh adaptive direct search algorithms for constrained optimization. SIAM Journal on Optimization, 17(1):188–217, 2006.

[4] C. Audet and J.E. Dennis, Jr. A Progressive Barrier for Derivative-Free Nonlinear Programming. SIAM Journal on Optimization, 20(1):445–472, 2009.

[5] C. Audet, A. Ianni, S. Le Digabel, and C. Tribes. Reducing the Number of Function Evaluations in Mesh Adaptive Direct Search Algorithms. SIAM Journal on Optimization, 24(2):621–642, 2014.

[6] S.C. Billups, J. Larson, and P. Graf. Derivative-Free Optimization of Expensive Functions with Computational Error Using Weighted Regression. SIAM Journal on Optimization, 23(1):27–53, 2013.

[7] R. Chen, M. Menickelly, and K Scheinberg. Stochastic Optimization Using a Trust-Region Method and Random Models. Technical report, arXiv, 2016.

[8] T.D. Choi and C.T. Kelley. Superlinear convergence and implicit filtering. SIAM Journal on Optimization, 10(4):1149–1162, 2000.

[9] G. Deng and M.C. Ferris. Adaptation of the UOBYQA Algorithm for Noisy Functions. In Proceedings of the 38th Conference on Winter Simulation, WSC '06, pages 312–319. Winter Simulation Conference, 2006.

[10] D.Yang and S.J. Flockton. Evolutionary algorithms with a coarse-to-fine function smoothing. Evolutionary Computation, IEEE International Conference, 2:657–662, 1995.

[11] C. Elster and A. Neumaier. A grid algorithm for bound constrained optimization of noisy functions. IMA Journal of Numerical Analysis, 15(4):585–608, 1995.

[12] V. A. Epanechnikov. Non-Parametric Estimation of a Multivariate Probability Density. Theory of Probability and Its Applications, 14(1):153–158, 1969.

[13] N.I.M. Gould, D. Orban, and Ph.L. Toint. CUTEst: a Constrained and Unconstrained Testing Environment with safe threads for mathematical optimization. Computational Optimization and Applications, 60(3):545–557, 2015. Code available at https://ccpforge.cse.rl.ac.uk/gf/project/cutest/wiki.

[14] D.R. Jones, C.D. Perttunen, and B.E. Stuckman. Lipschitzian optimization without the Lipschitz constant. Journal of Optimization Theory and Application, 79(1):157–181, 1993.

[15] J. Kostrowicki, L. Piela, B.J. Cherayil, and H.A. Scheraga. Performance of the diffusion equation method in searches for optimum structures of clusters of Lennard-Jones atoms. The Journal of Physical Chemistry, 95(10):4113–4119, 1991.

[16] J. Larson and S.C. Billups. Stochastic derivative-free optimization using a trust region framework. Computational Optimization and Applications, 64(3):619–645, 2016.

[17] S. Le Digabel. Algorithm 909: NOMAD: Nonlinear Optimization with the MADS algorithm. ACM Transactions on Mathematical Software, 37(4):44:1–44:15, 2011.

[18] S. Le Digabel and S.M. Wild. A Taxonomy of Constraints in Simulation-Based Optimization. Technical Report G–2015–57, Les cahiers du GERAD, 2015.

[19] J. Li, C. Wu, Z. Wu, and Q. Long. Gradient-free method for nonsmooth distributed optimization. Journal of Global Optimization, 61(2):325–340, 2015.

[20] Q. Liu, J. Zeng, and G. Yang. MrDIRECT: a multilevel robust DIRECT algorithm for global optimization problems. Journal of Global Optimization, 62(2):205–227, 2015.

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

[22] M.J.D. Powell. UOBYQA: Unconstrained optimization by quadratic approximation. Mathematical Programming, 92(3):555–582, 2002.

[23] S.E. Selvan, P.B. Borckmans, A. Chattopadhyay, and P.-A. Absil. Spherical mesh adaptive direct search for separating quasi-uncorrelated sources by range-based independent component analysis. Neural Computation, 25(9):2486–2522, 2013.

[24] C.-S. Shao, R.H. Byrd, E. Eskow, and R.B. Schnabel. Global Optimization for Molecular Clusters Using a New Smoothing Approach. In L. Biegler, T. Lorenz, A.R. Conn, T.F. Coleman, and F.N. Santosa, editors, Large-Scale Optimization with Applications, volume 94 of The IMA Volumes in Mathematics and its Applications, pages 163–199. Springer, 1997.

[25] Todd A. Sriver, James W. Chrissis, and Mark A. Abramson. Pattern search ranking and selection algorithms for mixed variable simulation-based optimization. European Journal of Operational Research, 198(3):878–890, 2009.

[26] B. Van Dyke and T.J. Asaki. Using QR Decomposition to Obtain a New Instance of Mesh Adaptive Direct Search with Uniformly Distributed Polling Directions. Journal of Optimization Theory and Applications, 159(3):805–821, 2013.

[27] F. Wei, Y. Wang, and Z. Meng. A smoothing function method with uniform design for global optimization. Pacific Journal of Optimization, 10(2):385–399, 2014.

[28] Z. Wu. The Effective Energy Transformation Scheme as a Special Continuation Approach to Global Optimization with Application to Molecular Conformation. SIAM Journal on Optimization, 6(3):748–768, 1996.