Hooke and Jeeves Based Multilevel Coordinate Search to Globally Solving Nonsmooth Problems

M. Fernanda P. Costa*, Ana Maria A.C. Rocha† and Edite M.G.P. Fernandes**

*Department of Mathematics and Applications and Centre of Mathematics
†Department of Production and Systems
**Algoritmi R&D Centre
University of Minho, Campus de Gualtar, 4710-057 Braga, Portugal

Abstract. In this paper, we present a derivative-free multilevel coordinate search (MCS) approach, that relies on the Hooke and Jeeves local search, for globally solving bound constrained optimization problems. Numerical experiments show that the proposed algorithm is effective in solving benchmark problems, when compared with the well-known solvers MCS and DIRECT.

Keywords: Global optimization, Derivative-free, Multilevel search
PACS: 02.60.Pn

INTRODUCTION

We consider solving the bound constrained global optimization problem (BCGOP)

$$\text{glob}\min_{x\in\Omega} f(x), \tag{1}$$

by a derivative-free deterministic method, where f is a continuous function and Ω is a hyperrectangle in $\mathbb{R}^n$: $\Omega = \{x \in \mathbb{R}^n : l \leq x \leq u\}$. We do not assume that f is differentiable and convex. Instead of searching for any local (non-global) solution we want the globally best point in the feasible region. Chemical equilibrium problems, safety verification problems, protein folding problems, semi-infinite programming problems, nonlinear least squares problems, can be formulated as shown in (1) [1, 2]. The subproblems, that appear in the penalty function approaches and augmented Lagrangian based multiplier algorithms when constrained global optimization problems need to be solved, have in general the form (1). Direct search methods might be suitable since we do not assume differentiability. However, they are only local optimization procedures and therefore there is no guarantee that a global solution is reached [3]. Deterministic and stochastic global search methods have been proposed to compute a solution to the problem (1). The most known deterministic algorithms are Lipschitzian-based partitioning and the multilevel coordinate search.

We are particularly interested in using a variant of the multilevel coordinate search (MCS) algorithm of Huyer and Neumaier [4] where smoothness is not assumed to be present. Analytic and numerical derivatives are not used. The MCS method is inspired by the DIRECT method of global optimization [5]. MCS is a popular method for bound constrained global optimization that only uses function values. To enhance the convergence result, the original method uses a local search based on a sequential quadratic programming. It consists of building a local quadratic model by triple searches, and defining afterwards a promising search direction by minimizing the quadratic model on a suitable box. Finally, a line search along this direction is carried out looking for the global minimizer. The method is guaranteed to converge if the objective function is continuous in the neighborhood of the global minimizer. An useful and promising method for local search is the Hooke and Jeeves (HJ) method [6]. This is a pattern search type method that does not require analytic and numerical derivatives in order to converge to stationary points. In this paper, we propose a new variant of the MCS algorithm where the local search carried out from a candidate good solution is the HJ search method.

This paper is organized as follows. First, we briefly describe the proposed MCS that uses the HJ algorithm as the local search procedure, henceforth denoted by MCS-HJ. Then, we show the numerical results, some comparisons and the conclusions.

HJ BASED MCS

The MCS algorithm seeks a global minimum of a multivariable function in a hyperbox defined by $l \leq x \leq u$, although we will simply use the term 'box' for simplicity. Like DIRECT, MCS searches for a global minimizer using branching recursively in order to divide, or split, the search space in a specific manner. They differ mainly in when and how a box is split. Appropriate splitting rules guarantee the convergence of box diameters to zero. Efficiency is ensured by a proper balance between global and local search. This balance is achieved in MCS by splitting the boxes for which the pair (s, f) (where s is a suitably assigned level and f is the base point function value) is not dominated by any other pair. This means that the global part of the algorithm explores boxes that have not been explored very often, while the local part splits boxes that have good objective function values. Additionally, the original MCS implements a local search procedure using line searches and sequential quadratic programming (SQP). It consists of building a local quadratic model by triple searches, and defining afterwards a promising search direction by minimizing the quadratic model on a suitable box. Details of the MCS algorithm, the underlying theory, and numerical comparisons can be found in [4]. The herein presented MCS algorithm contains the below described main procedures.

Initialization An initialization procedure generates a preliminary set of sub-boxes, using points input by the user or derived using a default generation procedure.

Splitting Information gained from already sampled points is used to determine the splitting coordinate as well as the position of the split. The level $s \in \{0, 1, \ldots, s_{\max}\}$ that is assigned to each box aims to measure the number of times a box has been processed. The following rules are considered: i) boxes with level $s_{\max}$ are considered too small for further splitting; ii) when a box of level s $(0 < s < s_{\max})$ is split, its level is set to zero, and its descendants get level $s+1$ or $\min(s+2, s_{\max})$. The golden section algorithm is used to split along a single coordinate and the larger fraction of the split gets level $s+1$, the smaller fraction gets level $\min\{s+2, s_{\max}\}$; iii) a level $s = 0$ indicates that a box has already been split and can be ignored. Thus, boxes with smaller levels mean that they are the larger boxes and have not been split very often yet. A box is split either by *rank* or by *expected gain* along a coordinate where the maximum gain in objective value is expected. MCS splits along a single coordinate at a time at chosen points. Only one new function evaluation is needed to split a box into two or three sub-boxes. Each sub-box is given a *base point*. *Base points* and objective function values of sub-boxes of level $s_{\max}$ are good starting points for the local search procedure.

Shopping Basket *Shopping basket* is another important concept in the MCS algorithm. It contains the minimizers that have been located so far. A *base point* of a box with level $s_{\max}$ is a candidate to the *shopping basket*. If the *base point* is likely to be in the basin of attraction of a minimizer already in the *shopping basket*, the point is discarded; otherwise, the local search procedure is started from it. The procedure that verifies if a point lies in the basin of attraction of a point already in the *shopping basket* is repeated from the output point of the local search procedure. If this point is really new then it will be put into the *shopping basket*.

Local Search When solving nonsmooth problems, methods that require analytic and numerical derivatives are not appropriate. In the present study, we propose the HJ method to replace the SQP within the local procedure aiming to refine the search for an accurate global solution. HJ is a derivative-free method from the pattern local search class [6]. Besides generating points along the coordinate directions with a fixed step size in the vicinity of a current point, also denoted as central point, x_k, the HJ algorithm also defines a pattern move whenever a successful iteration is found. This means that a new point, x_{k+1}, is found that improves over the central, in terms of objective function value. Then, a new trial point is defined using the pattern move, $x_{k+1} + (x_{k+1} - x_k)$, and a set of points are generated along the coordinate directions, using the trial point as the center of the search. If a new point with a better function value is found, the point is accepted and the step size is maintained; otherwise the search returns to x_{k+1}. Whenever an iteration does not provide a successful iterate, the step size of the search is reduced and the search process about the same central point is repeated. The algorithm stops when the step size falls below an error tolerance, ε_{tol}, since first-order convergence is guaranteed [3].

COMPUTATIONAL EXPERIMENTS AND CONCLUSIONS

In this section, a comparison between the DIRECT algorithm, the original MCS algorithm and the proposed MCS-HJ on a set of well-known BCGOP is presented. We report numerical results obtained with a set of twenty nonsmooth unconstrained problems (see [7] for a description of the functions) and consider the bounded set $\Omega = [-10, 10]^n$.

TABLE 1. Comparison results: DIRECT, MCS and MCS-HJ

Prob.				DIRECT		MCS			MCS-HJ		
		n	f^*	f_{best}	T_nfe	f_{best}	nfe_{loc}	T_nfe	f_{best}	nfe_{loc}	T_nfe
CB2	[7]	2	1.9522e+00	**1.9524e+00**	493	1.9637e+00	531	9671	1.9528e+00	129	9269
CB3	[7]	2	2.0000e+00	**2.0002e+00**	485	2.0010e+00	1098	10241	2.0002e+00	156	9293
DEM	[7]	2	-3.0000e+00	-2.9999e+00	259	-2.9999e+00	68	79	**-3.0000e+00**	9	20
EVD52	[7]	3	3.5997e+00	**3.5999e+00**	53661	3.7482e+00	2133	100000	3.7152e+00	2097	100000
Goffin†	[7]	50	0.0000e+00	0.7071e+00	169869	**2.8422e-14**	0	**101**	**2.8422e-14**	0	**101**
HS78	[7]	5	-2.9197e+00	-3.3528e+03	5109	9.6947e-03	2842	100000	**1.7976e-05**	6888	100000
L1HILB†	[7]	50	0.0000e+00	4.9368e-01	190269	1.8147e-04	1454	100001	**1.8223e-05**	2701	100001
LQ	[7]	2	-1.4142e00	**-1.4141e00**	403	-1.4127e00	226	12135	-1.4140e00	90	11999
Maxl†	[7]	20	0.0000e+00	0.7071e+00	179893	3.1234e-01	805	100000	3.2514e-01	2605	100000
Maxq†	[7]	20	0.0000e+00	0.5000e+00	179893	**1.1093e-31**	0	**99**	**1.1093e-31**	0	**99**
Maxquad	[7]	10	-8.4141e-01	**-7.5202e-01**	100081	-3.4291e-01	3162	100000	-6.0568e-01	2182	100000
Mifflin 1	[7]	2	-1.0000e+00	-0.9968e+00	118011	-1.0000e+00	1379	1539	**-1.0000e+00**	299	**459**
Mifflin 2	[7]	2	-1.0000e+00	-0.9999e+00	50355	-1.0000e+00	1379	1539	**-1.0000e+00**	299	**459**
MXHILB†	[7]	50	0.0000e+00	1.4142e-02	190269	8.5499e-07	1472	100000	**5.2203e-07**	2701	100000
OET6	[7]	4	2.0161e-03	**9.1009e-03**	100031	5.1195e-02	11939	100000	1.8728e-02	8057	100000
QL	[7]	2	7.2000e+00	**7.2002e+00**	923	7.3067e00	886	12765	7.2043e00	129	11998
Rosen-Suzuki	[7]	4	-4.4000e+01	**-4.3997e+01**	2127	-4.2374e+01	442	100000	-4.3814e+01	1171	100000
SPIRAL†	[7]	2	0.0000e+00	**0.0000e+00**	2587	3.7241e-12	297	311	5.1595e-11	202	216
Wolfe	[7]	2	-8.0000e+00	-7.9996e+00	14829	-7.9440e+00	2098	100000	**-8.0000e+00**	300	510
Wong 1	[7]	7	6.8063e+02	6.8839e+02	100081	6.9277e+02	5970	100000	**6.8549e+02**	733	100000
Ackley†	[8]	2	0.0000e+00	9.1114e-11	1861	**7.9884e-11**	442	492	5.0187e-06	3590	16700
Booth	[8]	2	0.0000e+00	**1.3726e-11**	1295	6.4428e-11	36	47	5.3644e-08	165	11767
Branin	[8]	2	3.9789e-01	3.9789e-01	195	**3.9789e-01**	29	41	3.9790e-01	101	113
Dixon-Price	[9]	2	0.0000e+00	3.3020e-11	1969	**3.9489e-19**	67	80	2.6828e-08	270	11947
Easom	[8]	2	-1.0000e+00	-9.9999e-01	32859	**-1.0000e+00**	42	**53**	-1.0000e+00	83	94
Goldstein-Price	[8]	2	3.0000e+00	3.0001e+00	191	3.0001e+00	29	40	**3.0000e+00**	89	100
Griewank†	[8]	2	0.0000e+00	2.4981e-04	100063	9.8647e-03	608	12537	**1.6255e-09**	1140	13118
Hartman3	[8]	3	-3.8628e+00	-3.8625e+00	199	-3.8624e+00	66	79	**-3.8625e+00**	160	173
Hartman6	[8]	6	-3.3224e+00	-3.3221e+00	571	**-3.3224e+00**	86	111	-3.3222e+00	324	349
S.Hump Camel	[9]	2	-1.0316e+00	-1.0316e+00	293	**-1.0316e+00**	33	**45**	-1.0316e+00	60	72
Levy†	[9]	30	0.0000e+00	2.9960e-02	106803	**1.5478e-16**	1214	1305	3.3192e-08	1711	100000
Perm	[9]	4	0.0000e+00	7.5234e-01	100015	**1.2388e-14**	60972	71735	2.6520e-04	95471	100094
Powell	[9]	24	0.0000e+00	4.2174e+00	101553	1.6061e-11	2885	3030	**0.0000e+00**	240	385
Power Sum	[8]	4	0.0000e+00	3.5730e-02	100113	3.3334e-12	45225	54497	**0.0000e+00**	16	35
Rastrigin†	[8]	2	0.0000e+00	6.4802e-11	1551	**0.0000e+00**	35	46	8.1442e-11	19443	25230
Shekel5	[8]	4	-1.0153e+01	-1.0152e+01	155	**-1.0153e+01**	65	**83**	-1.0153e+01	193	211
Shekel7	[8]	4	-1.0403e+01	-1.0402e+01	145	**-1.0403e+01**	88	**106**	-1.0403e+01	212	230
Shekel10	[8]	4	-1.0536e+01	-1.0535e+01	145	**-1.0536e+01**	85	**103**	-1.0536e+01	211	229
Shubert	[8]	2	-1.8673e+02	-1.8672e+02	2967	**-1.8673e+02**	58	69	-1.8672e+02	110	121
Trid	[9]	30	-2.1000e+02	1.0000e+01	102789	**-2.1000e+02**	1033	**1133**	-2.1000e+02	2888	2988
Zakharov	[9]	2	0.0000e+00	4.9791e-11	1419	2.9476e-14	28	41	**0.0000e+00**	19	32

Another set of twenty-one small and medium scale uni- and multi-modal differentiable benchmark functions (see [8, 9] for a description of the problems) is also used. Table 1 reports the name of the problem, 'Prob.'; the number of variables, 'n'; the known global minimum avialable in the literature, 'f^*'; the best function value obtained by the algorithm, 'f_{best}'; the total number of function evaluations,'T_nfe'; and the number of function evaluations in local search procedures, 'nfe_{loc}'. The algorithms in comparison – DIRECT, MCS and MCS-HJ – terminate when the following stopping condition is verified: $f_{best} - f^* \leq \max\left\{10^{-4}\,|f^*|, 10^{-10}\right\}$. If this condition does not hold, the algorithms also stop when the total number of function evaluations exceeds 10^5. In MCS and MCS-HJ, the following additional condition is used $s > s_{\max}$. For the problems marked with † in the table, the change of variables $y_i = x_i + 0.5\sqrt{2}$ is implemented since the global minimizer is the centroid of the Ω and all three methods, DIRECT, MCS and MCS-HJ, would find it in just one function evaluation. Other parameters from MCS are set as default, for example, $s_{\max} = 5n + 10$. In the original MCS, the number of iterations in the local search procedure is 50, and in MCS-HJ, the local HJ algorithm stops due to the tolerance $\varepsilon_{tol} = 10^{-3}$. Bold values show the best obtained results by DIRECT, MCS and MCS-HJ. From Table 1, we notice that none of the three algorithms is able to converge in 10^5 function evaluations when solving problem 'Maxl'. Analyzing the nonsmooth problems (first 20 rows of the table), we observe that MCS-HJ is better than the original MCS in terms of final objective function values on six problems ('DEM', 'HS78', 'L1HILB', 'MXHILB', 'Wolfe', 'Wong 1'), gives the same function value and number of function evaluations on two problems, 'Goffin' and 'Maxq', and the same function value with less function evaluations on 'Mifflin 1' and 'Mifflin 2'. DIRECT is better, in terms of final objective function values, than MCS and MCS-HJ on nine problems ('CB2', 'CB3', 'EVD52', 'LQ', 'Maxquad', 'OET6', 'QL', 'Rosen-Suzuki', 'SPIRAL') but it is not able to converge to a reasonable solution on five of the nonsmooth problems. Regarding the differentiable problems (last 21 rows of the table), we observe that in terms of final objective function values, the original MCS is better on eight problems ('Ackley', 'Branin', 'Dixon-Price', 'Hartman6', 'Levy', 'Perm', 'Rastrigin', 'Shubert'), MCS-HJ is better on six problems ('Goldstein-Price', 'Griewank', 'Hartman3', 'Powell', 'Power Sum', 'Zakharov') and DIRECT only on the problem 'Booth'. Furthermore, we note that both MCS and MCS-HJ reach the global solution of the problems 'Easom', 'S.Hump Camel', 'Shekel5', 'Shekel7', 'Shekel10' and 'Trid' although MCS is better in terms of function evaluations. We note that DIRECT fails to converge to the global solution on five problems of this set. The results reveal, in terms of final objective function values, that MCS and MCS-HJ are comparable and perform better than DIRECT. Overall MCS-HJ often outperforms MCS at the expense of additional function evaluations. Numerical testing involving large-dimensional problems and the implementation of an efficient technique to handle equality and inequality constraints are issues to be addressed in the near future.

ACKNOWLEDGMENTS

This work was financed by FEDER funds through COMPETE (Operational Programme Thematic Factors of Competitiveness) and by portuguese funds through FCT (Foundation for Science and Technology) within the project FCOMP-01-0124-FEDER-022674 and PEst-C/MAT/UI0013/2011.

REFERENCES

1. C.A. Floudas, P.M. Pardalos, C.S. Adjiman, W.R. Esposito, Z. Gümüs, S.T. Harding, J.L. Klepeis, C.A. Meyer, and C.A. Schweiger, *Handbook of Test Problems for Local and Global Optimization*, Kluwer, Dordrecht, 1999.
2. J.D. Pinter, *Global Optimization in Action*, Kluwer, Dordrecht, 1996.
3. T.G. Kolda, R.M. Lewis, and V. Torczon, *SIAM Review* **45**, 385–482 (2003).
4. W. Huyer and A. Neumaier, *Journal of Global Optimization* **14**, 331–355 (1999).
5. D.R. Jones, C.D. Perttunen, and B.E. Stuckman, *Journal of Optimization Theory and Applications* **79**, 157–181 (1993).
6. R. Hooke and T.A. Jeeves, *Journal on Associated Computation* **8**, 212–229 (1961).
7. V. Lukšan and J. Vlček, *Test Problems for Nonsmooth Unconstrained and Linearly Constrained Optimization*, Technical report, Institute of Computer Science, Academy of Sciences of the Czech Republic, 2000.
8. M.M. Ali, C. Khompatraporn, and Z.B. Zabinsky, *Journal of Global Optimization* **31**, 635–672 (2005).
9. A. Hedar, `http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO.htm`