On the performance of the ORTHOMADS algorithm on continuous and mixed-integer optimization problems

Marie-Ange Dahito, Laurent Genest, José Neto, Alessandro Maddaloni

On the performance of the OrthoMads algorithm on continuous and mixed-integer optimization problems

Marie-Ange Dahito[1,2], Laurent Genest[1], Alessandro Maddaloni[2], and José Neto[2]

[1] Stellantis, Route de Gisy, 78140 Vélizy-Villacoublay, France
marie-ange.dahito@polymtl.ca,laurent.genest@stellantis.com
[2] Samovar, Telecom SudParis, Institut Polytechnique de Paris, 19 place Marguerite Perey, 91120 Palaiseau, France
{alessandro.maddaloni,jose.neto}@telecom-sudparis.eu

Abstract. ORTHOMADS is an instantiation of the Mesh Adaptive Direct Search (MADS) algorithm used in derivative-free and blackbox optimization. We investigate the performance of the variants of ORTHOMADS on the `bbob` and `bbob-mixint`, respectively continuous and mixed-integer, testbeds of the COmparing Continuous Optimizers (COCO) platform and compare the considered best variants with heuristic and non-heuristic techniques. The results show a favourable performance of ORTHOMADS on the low-dimensional continuous problems used and advantages on the considered mixed-integer problems. Besides, a generally faster convergence is observed on all types of problems when the search phase of ORTHOMADS is enabled.

Keywords: Derivative-free optimization · Blackbox optimization · Benchmarking · Mesh Adaptive Direct Search · Mixed-integer blackbox.

1 Introduction

Derivative-free optimization (DFO) and blackbox optimization (BBO) are branches of numerical optimization that have known a fast growth in the past years, especially with the growing need to solve real-world application problems but also with the development of methods to deal with unavailable or numerically costly derivatives. DFO focuses on optimization techniques that make no use of derivatives while BBO deals with problems where the objective function is not analytically known, that is it is a blackbox. A regular blackbox objective is the output of a computer simulation: for instance, at Stellantis, the crash or acoustic outputs computed by the finite element simulation of a vehicle. The problems addressed in this paper are of the form:

$$\underset{x \in \mathcal{X}}{\text{minimize}} \quad f(x), \tag{1}$$

where $\mathcal{X}$ is a bounded domain of either $\mathbb{R}^n$ or $\mathbb{R}^c \times \mathbb{Z}^i$ with c and i respectively the number of continuous and integer variables. $n = c + i$ is the dimension of the

problem and f is a blackbox function. Heuristic and non-heuristic techniques can tackle this kind of problems. Among the main approaches used in DFO are direct local search methods. The latter are iterative methods that, at each iteration, evaluate a set of points in a certain radius that can be increased if a better solution is found or decreased if the incumbent remains the best point at the current iteration.

The Mesh Adaptive Direct Search (MADS) [1,4,5] is a famous direct local search method used in DFO and BBO that is an extension of the Generalized Pattern Search (GPS) introduced in [28]. MADS evolves on a mesh by first doing a global exploration called the search phase and then, if a better solution than the current iterate is not found, a local poll is performed. The points evaluated in the poll are defined by a finite set of poll directions that is updated at each iteration. The algorithm is derived in several instantiations available in the Nonlinear Optimization with the MADS algorithm (NOMAD) software [7,19] and its performance is evaluated in several papers. As examples, a broad comparison of DFO optimizers is performed on 502 problems in [25] and NOMAD is used in [24] with a DACE surrogate and compared with other local and global surrogate-based approaches in the context of constrained blackbox optimization on an automotive optimization problem and twenty two test problems.

Given the growing number of algorithms to deal with BBO problems, the choice of the most adapted method for solving a specific problem still remains complex. In order to help with this decision, some tools have been developed to compare the performance of algorithms. In particular, data profiles [20] are frequently used in DFO and BBO to benchmark algorithms: they show, given some precision or target value, the fraction of problems solved by an algorithm according to the number of function evaluations. There also exist suites of academic test problems: although the latter are treated as blackbox functions, they are analytically known, which is an advantage to understand the behaviour of an algorithm. There are also available industrial applications but they are rare.

Twenty two implementations of derivative-free algorithms for solving box-constrained optimization problems are benchmarked in [25] and compared with each other according to different criteria. They use a set of 502 problems that are categorized according to their convexity (convex or nonconvex), smoothness (smooth or non-smooth) and dimensions between 1 and 300. The algorithms tested include local-search methods such as MADS through NOMAD version 3.3 and global-search methods such as the NEW Unconstrained Optimization Algorithm (NEWUOA) [23] using trust regions and the Covariance Matrix Adaptation - Evolution Strategy (CMA-ES) [16] which is an evolutionary algorithm.

Simulation optimization deals with problems where at least some of the objective or constraints come from stochastic simulations. A review of algorithms to solve simulation optimization is presented in [2], among which the NOMAD software. However, this paper does not compare them due to a lack of standard comparison tools and large-enough testbeds in this optimization branch.

In [3], the MADS algorithm is used to optimize the treatment process of spent potliners in the production of aluminum. The problem is formalized as

a 7−dimensional non-linear blackbox problem with 4 inequality constraints. In particular, three strategies are compared using absolute displacements, relative displacements and the latter with a global Latin hypercube sampling search. They show that the use of scaling is particularly beneficial on the considered chemical application.

The instantiation ORTHOMADS is introduced in [1] and consists in using orthogonal directions in the poll step of MADS. It is compared to the initial LTMADS, where the poll directions are generated from a random lower triangular matrix, and to GPS algorithm on 45 problems from the literature. They show that MADS outperforms GPS and that the instantiation ORTHOMADS competes with LTMADS and has the advantage that its poll directions cover better the variable space.

The ORTHOMADS algorithm, which is the default MADS instantiation used in NOMAD, presents variants in the poll directions of the method. To our knowledge, the performance of these different variants has not been discussed in the literature. The purpose of this paper is to explore this aspect by performing experiments with the ORTHOMADS variants. This work is part of a project conducted with the automotive group Stellantis to develop new approaches for solving their blackbox optimization problems. Our contributions are first the evaluations of the ORTHOMADS variants on continuous and mixed-integer optimization problems. Besides, the contribution of the search phase is studied and shows a general deterioration of the performance when the search is turned off. The effect however decreases with increasing dimension. Two from the best variants of ORTHOMADS are identified on each of the used testbeds and their performance is compared with other algorithms including heuristic and non-heuristic techniques. Our experiments exhibit particular variants of ORTHOMADS performing best depending on problems features. Plots for analyses are available at the following link: https://github.com/DahitoMA/ResultsOrthoMADS.

The paper is organized as follows. Section 2 gives an overview of the MADS algorithm and its ORTHOMADS variants. In Section 3, the variants of ORTHOMADS are evaluated on the `bbob` and `bbob-mixint` suites that consist respectively of continuous and mixed-integer functions. Then, two from the best variants of ORTHOMADS are compared with other algorithms in Section 4. Finally, Section 5 discusses the results of the paper.

2 MADS and the variants of OrthoMads

This section gives an overview of the MADS algorithm and explains the differences among the ORTHOMADS variants.

2.1 The MADS algorithm

MADS is an iterative direct local search method used for DFO and BBO problems. The method relies on a mesh M_k updated at each iteration and determined

by the current iterate x_k, a mesh parameter size $\delta_k > 0$ and a matrix D whose columns consist of p positive spanning directions. The mesh is defined as follows:

$$M_k := \{x_k + \delta_k D y : y \in \mathbb{N}^p\}, \tag{2}$$

where the columns of D form a positive spanning set $\{D_1, D_2, \ldots, D_p\}$ and $\mathbb{N}$ stands for natural numbers.

The algorithm proceeds in two phases at each iteration: the search and the poll. The search phase is optional and similar to a design of experiment: a finite set of points S_k, stemming generally from a surrogate model prediction and a Nelder-Mead (N-M) search [21], are evaluated anywhere on the mesh. If the search fails at finding a better point, then a poll is performed. During the poll phase, a finite set of points are evaluated on the mesh in the neighbourhood of the incumbent. This neighbourhood is called the frame F_k and has a radius of $\Delta_k > 0$ that is called the poll size parameter. The frame is defined as follows:

$$F_k := \{x \in M_k : \|x - x_k\|_\infty \leq \Delta_k b\}, \tag{3}$$

where $b = \max\{\|d\|_\infty, d \in \mathbb{D}\}$ and $\mathbb{D} \subset \{D_1, D_2, \ldots, D_p\}$ is a finite set of poll directions. The latter are such that their union over iterations grows dense on the unit sphere.

The two size parameters are such that $\delta_k \leq \Delta_k$ and evolve after each iteration: if a better solution is found, they are increased and otherwise decreased. As the mesh size decreases more drastically than the poll size in case of an unsuccessful iteration, the choice of points to evaluate during the poll becomes greater with unsuccessful iteration. Usually, $\delta_k = \min\{\Delta_k, \Delta_k^2\}$. The description of the MADS algorithm is given in Algorithm 1 and inspired from [6].

Algorithm 1: Mesh Adaptive Direct Search (MADS)

Initialize $k = 0$, $x_0 \in \mathbb{R}^n$, $D \in \mathbb{R}^{n \times p}$, $\Delta_0 > 0$, $\tau \in (0,1) \cap \mathbb{Q}$, $\epsilon_{\text{stop}} > 0$
1. Update $\delta_k = \min\{\Delta_k, \Delta_k^2\}$
2. **Search**
  If $f(x) < f(x_k)$ for $x \in S_k$ then $x_{k+1} \leftarrow x$, $\Delta_{k+1} \leftarrow \tau^{-1}\Delta_k$ and go to 4
  Else go to 3
3. **Poll**
  Select $\mathbb{D}_{k,\Delta_k}$ such that $P_k := \{x_k + \delta_k d : d \in \mathbb{D}_{k,\Delta_k}\} \subset F_k$
  If $f(x) < f(x_k)$ for $x \in P_k$ then $x_{k+1} \leftarrow x$, $\Delta_{k+1} \leftarrow \tau^{-1}\Delta_k$ and go to 4
  Else $x_{k+1} \leftarrow x_k$ and $\Delta_{k+1} \leftarrow \tau\Delta_k$
4. **Termination**
  If $\Delta_{k+1} \geq \epsilon_{\text{stop}}$ then $k \leftarrow k + 1$ and go to 1
  Else stop

2.2 ORTHOMADS variants

MADS has two main instantiations called ORTHOMADS and LTMADS, the latter being the first developed. Both variants are implemented in the NOMAD

software but as ORTHOMADS is to be preferred for its coverage property in the variable space, it was used for the experiments of this paper with NOMAD version 3.9.1.

The NOMAD implementation of ORTHOMADS provides 6 variants of the algorithm according to the number of directions used in the poll or according to the way that the last poll direction is computed. They are listed below.

`ORTHO N + 1 NEG` computes $n+1$ directions among which n are orthogonal and the $(n+1)^{th}$ direction is the opposite sum of the n first ones.

`ORTHO N + 1 UNI` computes $n+1$ directions among which n are orthogonal and the $(n+1)^{th}$ direction is generated from a uniform distribution.

`ORTHO N + 1 QUAD` computes $n+1$ directions among which n are orthogonal and the $(n+1)^{th}$ direction is generated from the minimization of a local quadratic model of the objective.

`ORTHO 2N` computes $2n$ directions that are orthogonal. More precisely each direction is orthogonal to $2n-2$ directions and collinear with the remaining one.

`ORTHO 1` uses only one direction in the poll.

`ORTHO 2` uses two opposite directions in the poll.

In the plots, the variants will respectively be denoted using `Neg`, `Uni`, `Quad`, `2N`, `1` and `2`.

3 Test of the variants of OrthoMads

In this section, we try to identify potentially better direction types of ORTHOMADS and investigate the contribution of the search phase.

3.1 The COCO platform and the used testbeds

The COmparing Continuous Optimizers (COCO) platform [17] is a benchmarking framework for blackbox optimization. In this respect, several suites of standard test problems are provided and are declined in variants, also called instances. The latter are obtained from transformations in variable and objective space in order to make the functions less regular.

In particular, the `bbob` testbed [13] provides 24 continuous problems for blackbox optimization, each of them available in 15 instances and in dimensions 2, 3, 5, 10, 20 and 40. The problems are categorized in five subgroups: separable functions, functions with low or moderate conditioning, ill-conditioned functions, multi-modal functions with global structure and multi-modal weakly structured functions. All problems are known to have their global optima in $[-5, 5]^n$, where n is the size of a problem.

The mixed-integer suite of problems `bbob-mixint` [29] derives the `bbob` and `bbob-largescale` [30] problems by imposing integer constraints on some variables. It consists of the 24 functions of `bbob` available in 15 instances and in dimensions 5, 10, 20, 40, 80 and 160.

COCO also provides various tools for algorithm comparison, notably Empirical Cumulative Distribution Function (ECDF) plots (or data profiles) that are

used in this paper. They show the empirical runtimes, computed as the number of function evaluations to reach given function target values, divided by the dimension. A function target value is defined as $f_t = f^* + \Delta f_t$, where f^* is the minimum value of a function f and Δf_t is a target precision.

For the `bbob` and `bbob-mixint` testbeds, the target precisions are 51 values between 10^{-8} and 10^2. Thus, if a method reaches 1 in the ordinate axis of an ECDF plot, it means 100% of function target values have been reached, including the smallest one $f^* + 10^{-8}$. The presence of a cross on an ECDF curve indicates when the maximal budget of function evaluations is reached. After the cross, COCO estimates the runtimes: it is called simulated restarts.

For `bbob`, an artificial solver called *best 2009* is present on the plots and is used as reference solver. Its data comes from the BBOB-2009 workshop[3] comparing 31 solvers.

The statistical significance of the results is evaluated in COCO using the rank-sum test.

3.2 Parameter setting

In order to test the performance of the different variants of ORTHOMADS, the `bbob` and `bbob-mixint` suites of COCO were used, in particular the problems that have a dimension lower than or equal to 20. This limit in the dimensions has two main reasons: the first one is the computational cost required for the experiments and, with the perspective of solving real-world problems, 20 is already a high dimension in this expensive blackbox context. Only the first 5 instances of each function were used, that is a total of respectively 600 and 360 problems used from `bbob` and `bbob-mixint`. A maximal function evaluation budget of $2 \times 10^3 \times n$ was set, with n being the dimension of the considered problem.

To see the contribution of the search phase, the experiments on the variants were divided in two subgroups: the first one using the default search of ORTHOMADS and the second one where the search phase is disabled. The latter is obtained by setting the four parameters `NM_SEARCH`, `VNS_SEARCH`, `SPECULATIVE_SEARCH` and `MODEL_SEARCH` of NOMAD to the value `no`. In the plots, the label *NoSrch* is used when the search is turned off. The search notably includes the use of a quadratic model and of the N-M method. The minimal mesh size was set to 10^{-11}.

Experiments were run with restarts allowed for unsolved problems when the evaluation budget is not reached. This may happen due to internal stopping criteria of the solvers. The initial points used are suggested by COCO through the method `initial_solution_proposal()`.

3.3 Results

Continuous problems As said previously, the contribution of the search phase was studied. The results aggregated on all functions in dimensions $5, 10$ and

[3] https://coco.gforge.inria.fr/doku.php?id=bbob-2009-results

20 on the `bbob` suite are depicted on Figure 1. They show that enabling the search step in NOMAD generally leads to an equivalent or higher performance of the variants and this improvement can be important. Besides, using one or two directions with or without search is often far from being competitive with the other variants. In particular, `1 NoSrch` is often the worst or among the worsts, except on Discus which is an ill-conditioned quadratic function, where it competes with the variants that do not use the search. As mentioned in Section 1, the plots depicting the results described in the paper are available online.

Looking at the results aggregated on all functions for `ORTHO 2N`, `ORTHO N + 1 NEG`, `ORTHO N + 1 QUAD` and `ORTHO N + 1 UNI`, the search increases the success rate from nearly 70%, 55% and 40% up to 90%, 80% and 65% respectively in dimensions 2, 3 and 5, as shown in Figure 1a for dimension 5. From dimension 10, the advantage of the search decreases and the performance of `ORTHO N + 1 UNI` visibly stands out from the other three variants mentioned above since it decreases with or without the search, as illustrated in Figures 1b and 1c.

Focusing on some families of functions, `Neg NoSrch` seems slightly less impacted than the other `NoSrch` variants by the increase of the dimension.

On ill-conditioned problems, the variants using search are more sensitive to the increase of the dimension.

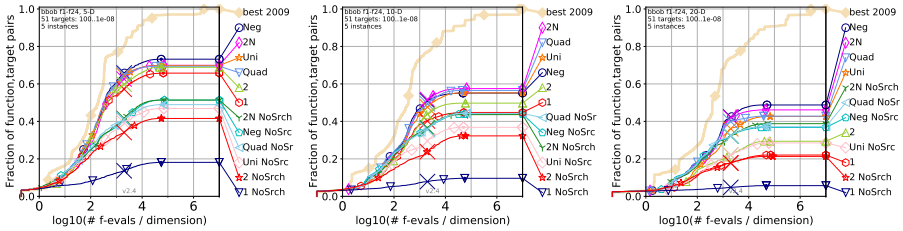
Considering multi-modal functions with adequate global structure, `2N NoSrch` solves 15% more problems than the other `NoSrch` variants in 2D. In this dimension, the variants using search have a better success rate than the *best 2009* up to a budget of 200 function evaluations. From 10D, all curves are rather flat: all ORTHOMADS variants tend to a local optimum.

With increasing dimension, `Neg` is competitive or better than the others on multi-modal problems without global structure, followed by `2N`. In particular, in dimension 20 both variants are competitive and outperform the remaining variants that use search on the Gallagher's Gaussian 101−me peaks function, and `Neg` outperforms them with a gap of more than 20% in their success rate on the Gallagher's Gaussian 21−hi peaks function which is also ill-conditioned.

Since `Neg` and `2N` are often among the best variants on the considered problems and have an advantage on some multi-modal weakly structured functions, they are chosen for comparison with other solvers.

Mixed-integer problems The experiments performed on the mixed-integer problems also show a similar or improved performance of the ORTHOMADS variants when the search step is enabled in NOMAD, as illustrated in Figure 2 in dimensions 5, 10 and 20. Looking at Figure 2a for instance, in the given budget of $2 \times 10^3 \times n$, the variant denoted as `2` solves 75% of the problems in dimension 5 against 42% for `2 NoSrch`.

However, it is not always the case: the only use of the poll directions is sometimes favourable. It is notably the case on the Schwefel function in dimension 20 where the curve `Neg NoSrch` solves 43% of the problems, which is the highest success rate when the search and non-search settings are compared together.



(a) 5D (b) 10D (c) 20D

Fig. 1: ECDF plots: the variants of ORTHOMADS with and without the search step on the `bbob` problems. Results aggregated on all functions in dimensions 5, 10 and 20.

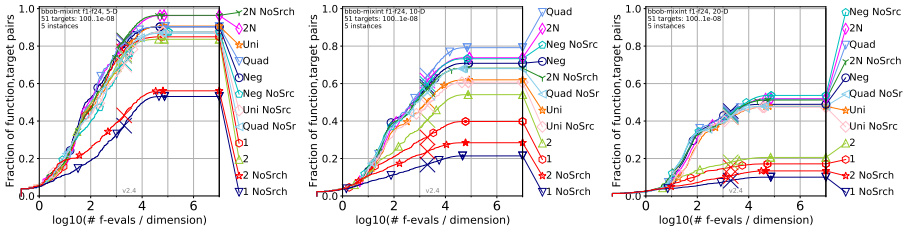
When the search is disabled, `ORTHO 2N` seems preferable in small dimension, namely here in 5D as presented in Figure 2a. In this dimension, it is sometimes the only variant that solves all the instances of a function in the given budget: it is the case for the step-ellipsoidal function, the two Rosenbrock functions (original and rotated), the Schaffer functions, and the Schwefel function. It also solves all the separable functions in 5D and can therefore solve the different types of problems. Although the difference is less noticeable with the search step enabled, this variant is still a good choice, especially on multi-modal problems with adequate global structure.

On the whole, looking at Figure 2, `ORTHO 1` and `ORTHO 2` solve less problems than the other variants and the gap in performance with the other direction types increases with the dimension, whether using the search phase or not. Although the use of the search helps solving some functions in low dimension such as the sphere or linear slope functions in 5D, both variants perform poorly in dimension 20 on second-order separable functions, even if the search enables the solution of linear slope which is a linear function. Among these two variants, using 2 poll directions also seems better than only one, especially in dimension 10 where `ORTHO 2` solves more than 23% and 40% of problems respectively without and with use of search, against 16% and 31% for `ORTHO 1` as presented in Figure 2b.

Among the four remaining variants, `ORTHO N + 1 UNI` reaches equivalent or less targets than the others whether considering the setting where the search is available or when only the poll directions are used, as depicted in Figure 2. In particular, in dimension 5, the four variants using more than $n+1$ poll directions solve more than 85% of the separable problems with or without search. But when the dimension increases, `ORTHO N + 1 UNI` has a disadvantage on the Rastrigin functions where the use of the search does not noticeably help the convergence of the algorithm.

Focusing on the different function types, no algorithm among the variants `ORTHO 2N`, `ORTHO N + 1 NEG` and `ORTHO N + 1 QUAD` seem to particularly outperform the others in dimensions 10 and 20. A higher success rate is however noticeable on multimodal weakly structured problems with search available for `ORTHO N + 1`

`NEG` in comparison with `ORTHO N + 1 QUAD` and for the latter in comparison with `ORTHO 2N`. Besides, `Neg` reaches more targets on problems with low or moderate conditioning. For these reasons, `ORTHO N + 1 NEG` was chosen for comparison with other solvers. Besides, the mentioned slight advantage of `ORTHO N + 1 QUAD` over `ORTHO 2N`, its equivalent or better performance on separable and ill-conditioned functions compared with the latter variant, makes it a good second choice to represent ORTHOMADS.



(a) 5D (b) 10D (c) 20D

Fig. 2: ECDF plots: the variants of ORTHOMADS with and without the search step on the `bbob-mixint` problems. Results aggregated on all functions in dimensions $5, 10$ and 20.

4 Comparison of OrthoMads with other solvers

The previous experiments showed the advantage of using the search step in ORTHOMADS to speed up convergence. They also revealed the effectiveness of some variants that are used here for comparisons with other algorithms on the continuous and mixed-integer suites.

4.1 Compared algorithms

Apart from ORTHOMADS, the other algorithms used for comparison on `bbob` are first, three deterministic algorithms: the quasi-Newton Broyden-Fletcher-Goldfarb-Shanno (BFGS) method [22], the quadratic model-based NEWUOA and the adaptive N-M [14] that is a simplicial search. Stochastic methods are also used among which a Random Search (RS) algorithm [10] and three population-based algorithms: a surrogate-assisted CMA-ES, Differential Evolution (DE) [27] and Particle Swarm Optimization (PSO) [18,11].

In order to perform algorithm comparisons on `bbob-mixint`, data from four stochastic methods were collected: RS, the mixed-integer variant of CMA-ES, DE and the Tree-structured Parzen Estimator (TPE) [8] that is a stochastic model-based technique.

BFGS is an iterative quasi-Newton linesearch method that uses approximations of the Hessian matrix of the objective. At iteration k, the search direction

p_k solves a linear system $B_k p_k = -\nabla f(x_k)$, where x_k is the iterate, f the objective function and $B_k \approx \nabla^2 f(x_k)$. The matrix B_k is then updated according to a formula. In the context of BBO, the derivatives are approximated with finite differences.

NEWUOA is the Powell's model-based algorithm for DFO. It is a trust-region method that uses sequential quadratic interpolation models to solve unconstrained derivative-free problems.

The N-M method is a heuristic DFO method that uses simplices. It begins with a non degenerated simplex. The algorithm identifies the worst point among the vertices of the simplex and tries to replace it by reflection, expansion or contraction. If none of these geometric transformations of the worst point enables to find a better point, a contraction preserving the best point is done. The adaptive N-M method uses the N-M technique with adaptation of parameters to the dimension, which is notably useful in high dimensions.

RS is a stochastic iterative method that performs a random selection of candidates: at each iteration, a random point is sampled and the best between this trial point and the incumbent is kept.

CMA-ES is a state-of-the art evolutionary algorithm used in DFO. Let $\mathcal{N}(m, C)$ denote a normal distribution of mean m and covariance matrix C. It can be represented by the ellipsoid $x^\top C^{-1} x = 1$. The main axes of the ellipsoid are the eigenvectors of C and the square roots of their lengths correspond to the associated eigenvalues. CMA-ES iteratively samples its populations from multivariate normal distributions. The method uses updates of the covariance matrices to learn a quadratic model of the objective.

DE is a meta-heuristic that creates a trial vector by combining the incumbent with randomly chosen individuals from a population. The trial vector is then sequentially filled with parameters from itself or the incumbent. Finally the best vector between the incumbent and the created vector is chosen.

PSO is an archive-based evolutionary algorithm where candidate solutions are called particles and the population is a swarm. The particles evolve according to the global best solution encountered but also according to their local best points.

TPE is an iterative model-based method for hyperparameter optimization. It sequentially builds a probabilistic model from already evaluated hyperparameters sets in order to suggest a new set of hyperparameters to evaluate on a score function that is to be minimized.

4.2 Parameter setting

To compare the considered best variants of ORTHOMADS with other methods, the 15 instances of each function were used and the maximal function evaluation budget was increased to $10^5 \times n$, with n being the dimension.

For the `bbob` problems, the data used for BFGS, DE and the adaptive N-M method comes from the experiments of [31]. CMA-ES was tested in [15], the data of NEWUOA is from [26], the one of PSO is from [12] and RS results come from [9]. The comparison data of CMA-ES, DE, RS and TPE used

on the `bbob-mixint` suite comes from the experiments of [29]. All are accessible from the data archives of COCO with the `cocopp.archives.bbob` and `cocopp.archives.bbob_mixint` methods.

4.3 Results

Continuous problems Figures 3 and 4 show the ECDF plots comparing the methods on the different function types and on all functions, respectively in dimensions 5 and 20 on the continuous suite. Compared with BFGS, CMA-ES, DE, the adaptive N-M method, NEWUOA, PSO and RS, ORTHOMADS often performs in the average for medium and high dimensions. For small dimensions 2 and 3, it is however among the most competitive.

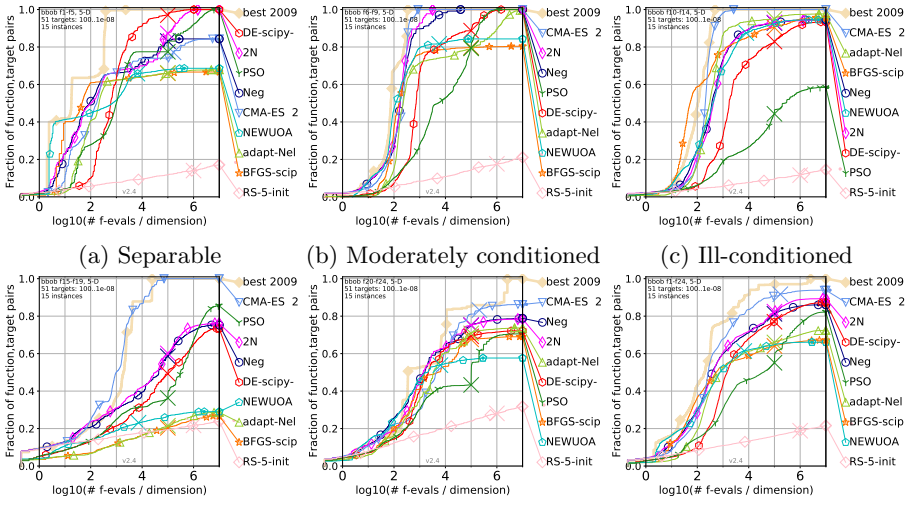
Considering the results aggregated on all functions and splitting them over all targets according to the function evaluations, they can be divided in three parts. The first one consists of very limited budgets (about $20 \times n$) where NEWUOA competes with or outperforms the others. After that, BFGS becomes the best for an average budget and CMA-ES outperforms the latter for high evaluation budgets (above the order of $10^2 \times n$), as shown in Figures 3f and 4f. The obtained performance restricted to a low budget is an important feature relevant to many applications for which each function evaluation may last hours or even days.

On multi-modal problems with adequate structure, there is a noticeable gap between the performance of CMA-ES, which is the best algorithm on this kind of problems, and the other algorithms as shown by Figures 3d and 4d. ORTHOMADS performs the best in the remaining methods and competes with CMA-ES for low budgets. It is even the best method up to a budget of $10^3 \times n$ in 2D and 3D while it competes with CMA-ES in higher dimensions for budgets lower than the order of $10^2 \times n$.

RS is often the worse algorithm to use on the considered problems.

Mixed-integer problems Figures 5 and 6 show the ECDF plots comparing the methods on the different function types and on all functions, respectively in dimensions 5 and 20 on the mixed-integer suite. The comparisons of `NEG` and `QUAD` with CMA-ES, DE, RS and TPE show an overall advantage of these ORTHOMADS variants over the other methods. A gap is especially visible on separable and ill-conditioned problems, respectively depicted in Figures 5a and 6a and Figures 5c and 6c in dimensions 5 and 20, but also on moderately conditioned problems as shown in Figures 5b and 6b in 5D and 20D. On multi-modal problems with global structure, ORTHOMADS is to prefer only in small dimensions: from 10D its performance highly deteriorates and CMA-ES and DE seem to be better choices. On multi-modal weakly structured functions, the advantages of ORTHOMADS compared to the others emerge when the dimension increases.

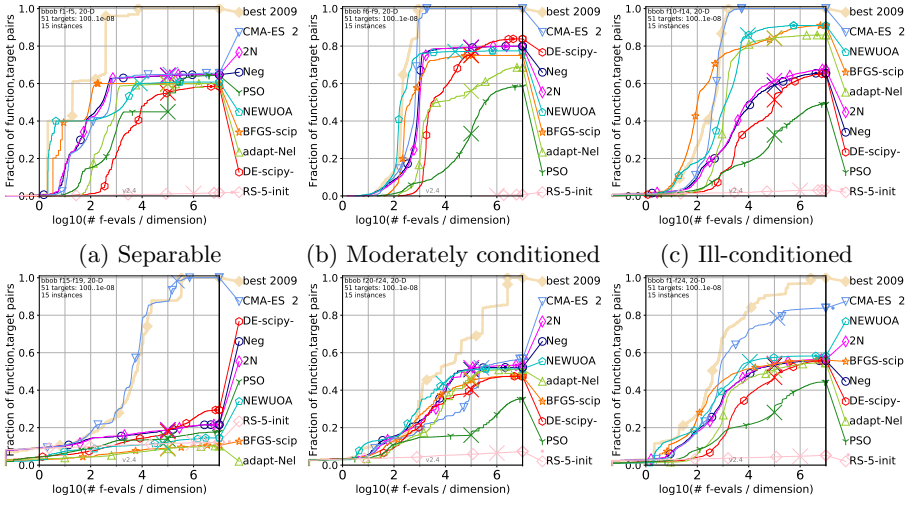
Besides, although the performance of all algorithms decreases with increasing dimensions, ORTHOMADS seems less sensitive to that. For instance, for a budget of $10^2 \times n$, ORTHOMADS reaches 15% more targets than CMA-ES and TPE that are the second best algorithms until this budget, and in dimension 20 this gap increases to 18% for CMA-ES and 25% for TPE.



(a) Separable (b) Moderately conditioned (c) Ill-conditioned

(d) Multi-modal (e) Weakly structured (f) All functions

Fig. 3: ECDF plots: comparison of the two variants `ORTHO 2N` and `ORTHO N + 1 NEG` of ORTHOMADS with BFGS, NEWUOA, adaptive N-M, RS, CMA-ES, DE and PSO on the `bbob` problems. Results aggregated on the function types and on all functions in dimension 5.

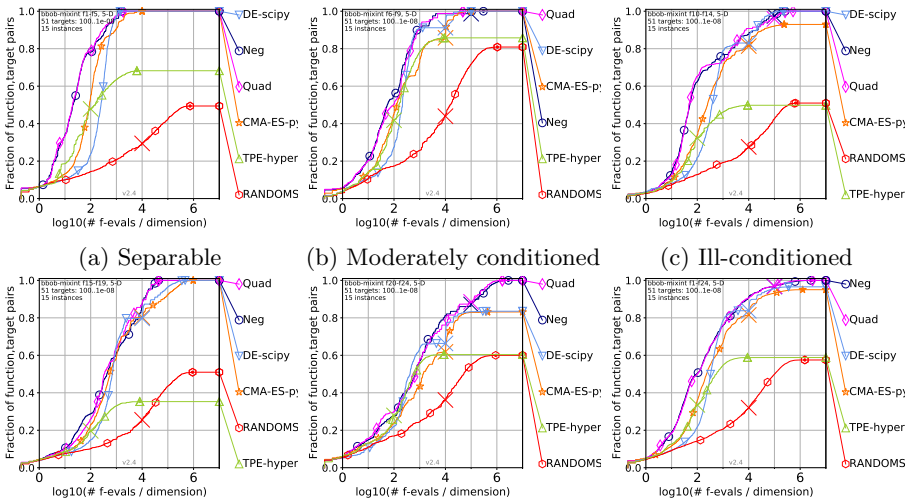


(a) Separable (b) Moderately conditioned (c) Ill-conditioned

(d) Multi-modal (e) Weakly structured (f) All functions

Fig. 4: ECDF plots: comparison of the two variants `ORTHO 2N` and `ORTHO N + 1 NEG` of ORTHOMADS with BFGS, NEWUOA, adaptive N-M, RS, CMA-ES, DE and PSO on the `bbob` problems. Results aggregated on the function types and on all functions in dimension 20.

On the overall picture, presented in Figures 5f and 6f, RS performs poorly. The budget allocated to TPE, which is only $10^2 \times n$, is way smaller than the ones allocated to the other methods. In this limited budget, TPE competes with CMA-ES in 5D and is better or competitive with DE in 10D and 20D. The latter competes with ORTHOMADS after a budget in the order of $10^3 \times n$. Thus, after 5×10^3 function evaluations, only DE competes with ORTHOMADS in 5D where both methods reach 70% of function-target pairs. Finally, CMA-ES competes with ORTHOMADS when the budget approaches $10^4 \times n$ function evaluations. Hence, restricted budgets seem to favour the direct local search method while expensive budgets favour the evolutionary algorithms CMA-ES and DE.



(a) Separable (b) Moderately conditioned (c) Ill-conditioned

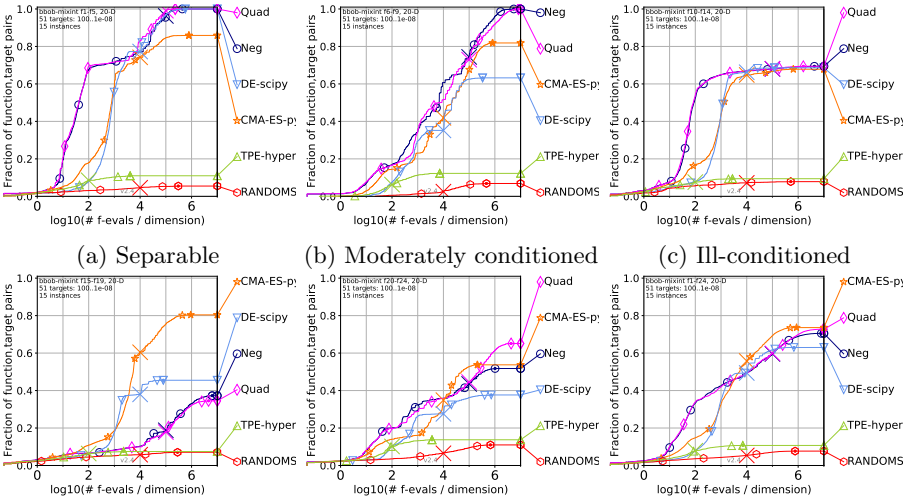
(d) Multi-modal (e) Weakly structured (f) All functions

Fig. 5: ECDF plots: comparison of the two variants `ORTHO N + 1 NEG` and `ORTHO N + 1 QUAD` of ORTHOMADS with RS, CMA-ES, DE and TPE on the `bbob-mixint` problems. Results aggregated on the function types and on all functions in dimension 5.

5 Conclusion

This paper investigates the performance of the different poll direction types available in ORTHOMADS on continuous and mixed-integer problems from the literature in a blackbox context. On these two types of problems, `ORTHO N + 1 NEG` competes with or outperforms the other variants of the algorithm whereas using only 1 or 2 directions is often far from being competitive.

On the continuous functions considered, the best poll direction types identified are `ORTHO N + 1 NEG` and `ORTHO 2N`, especially on multi-modal weakly struc-



(a) Separable (b) Moderately conditioned (c) Ill-conditioned

(d) Multi-modal (e) Weakly structured (f) All functions

Fig. 6: ECDF plots: comparison of the two variants `ORTHO N + 1 NEG` and `ORTHO N + 1 QUAD` of ORTHOMADS with RS, CMA-ES, DE and TPE on the `bbob-mixint` problems. Results aggregated on the function types and on all functions in dimension 20.

tured problems. ORTHOMADS is advantageous in small dimensions and achieves mean results for medium and high dimensions compared to the other algorithms. It also performs well on multi-modal problems with global structure where it competes with CMA-ES for limited budgets.

For very limited budgets, the trust-region method NEWUOA is favourable on continuous problems, followed by the linesearch method BFGS for a medium budget and finally the evolutionary algorithm CMA-ES for a high budget.

The results on the mixed-integer suite show that, among the poll direction types, `ORTHO 2N` is preferable in small dimension. Otherwise, `ORTHO N + 1 NEG` and `ORTHO N + 1 QUAD` are among the best direction types. Comparing them to other methods show that ORTHOMADS often outperforms the compared algorithms and seems more resilient to the increase of the dimension. For limited budgets, ORTHOMADS seems a good choice among the other considered algorithms to solve unconstrained mixed-integer blackbox problems. This is notably interesting regarding real-world application problems and, in particular, the mixed-integer optimization problems of Stellantis, where the number of allowed blackbox evaluations is often limited to a few hundreds. In the latter case, the variables are typically the thicknesses of the sheet metals, considered as continuous, and the materials that are categorical variables encoded as integers.

Finally, studying the contribution of the search step of ORTHOMADS shows that disabling it generally leads to a deteriorated performance of the algorithm. Indeed, the default search sequentially executes a N-M search and a quadratic

model search that enable a global exploration and accelerate the convergence. However, this effect softens when the dimension increases.

References

1. Abramson, M.A., Audet, C., Dennis, Jr., J.E., Le Digabel, S.: OrthoMADS: A Deterministic MADS Instance with Orthogonal Directions. SIAM Journal on Optimization **20**(2), 948–966 (2009). https://doi.org/10.1137/080716980
2. Amaran, S., Sahinidis, N., Sharda, B., Bury, S.: Simulation optimization: a review of algorithms and applications. 4OR **12**(4), 301–333 (2014). https://doi.org/10.1007/s10288-014-0275-2
3. Audet, C., Béchard, V., Chaouki, J.: Spent potliner treatment process optimization using a MADS algorithm. Optimization and Engineering **9**(2), 143–160 (2008)
4. Audet, C., Dennis, Jr., J.: Mesh Adaptive Direct Search Algorithms for Constrained Optimization. SIAM Journal on Optimization **17**(1), 188–217 (2006). https://doi.org/10.1137/040603371
5. Audet, C., Dennis, Jr., J.: A Progressive Barrier for Derivative-Free Nonlinear Programming. SIAM Journal on Optimization **20**(1), 445–472 (2009). https://doi.org/10.1137/070692662
6. Audet, C., Hare, W.: Derivative-Free and Blackbox Optimization. Springer Series in Operations Research and Financial Engineering, Springer, Cham, Switzerland (2017). https://doi.org/10.1007/978-3-319-68913-5
7. Audet, C., Le Digabel, S., Tribes, C.: NOMAD user guide. Tech. Rep. G-2009-37, Les cahiers du GERAD (2009), https://www.gerad.ca/nomad/Downloads/user_guide.pdf
8. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. Advances in neural information processing systems **24** (2011)
9. Brockhoff, D., Hansen, N.: The impact of sample volume in random search on the bbob test suite. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. p. 1912–1919. GECCO '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3319619.3326894
10. Brooks, S.H.: A discussion of random methods for seeking maxima. Operations research **6**(2), 244–251 (1958)
11. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. pp. 39–43. IEEE (1995). https://doi.org/10.1109/MHS.1995.494215
12. El-Abd, M., Kamel, M.S.: Black-box optimization benchmarking for noiseless function testbed using particle swarm optimization. In: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers. p. 2269–2274. GECCO '09, Association for Computing Machinery, New York, NY, USA (2009). https://doi.org/10.1145/1570256.1570316
13. Finck, S., Hansen, N., Ros, R., Auger, A.: Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Tech. Rep. 2009/20, Research Center PPE (2009)
14. Gao, F., Han, L.: Implementing the Nelder-Mead simplex algorithm with adaptive parameters. Comp. Opt. and Appl. **51**, 259–277 (2012)
15. Hansen, N.: A global surrogate assisted CMA-ES. In: Proceedings of the Genetic and Evolutionary Computation Conference. p. 664–672. GECCO

'19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3321707.3321842
16. Hansen, N., Auger, A.: Principled design of continuous stochastic search: From theory to practice. In: Theory and principled methods for the design of metaheuristics, pp. 145–180. Springer (2014)
17. Hansen, N., Auger, A., Ros, R., Mersmann, O., Tušar, T., Brockhoff, D.: COCO: A platform for comparing continuous optimizers in a black-box setting. Optimization Methods and Software **36**(1), 114–144 (2021). https://doi.org/10.1080/10556788.2020.1808977
18. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks. vol. 4, pp. 1942–1948. IEEE Service Center, Piscataway, Perth, Australia (1995). https://doi.org/10.1109/ICNN.1995.488968
19. Le Digabel, S.: Algorithm 909: NOMAD: Nonlinear Optimization with the MADS algorithm. ACM Transactions on Mathematical Software **37**(4), 44:1–44:15 (2011). https://doi.org/10.1145/1916461.1916468
20. Moré, J.J., Wild, S.M.: Benchmarking derivative-free optimization algorithms. SIAM Journal on Optimization **20**(1), 172–191 (2009). https://doi.org/10.1137/080724083
21. Nelder, J.A., Mead, R.: A simplex method for function minimization. The computer journal **7**(4), 308–313 (1965)
22. Nocedal, J., Wright, S.: Numerical optimization. Springer Science & Business Media (2006)
23. Powell, M.J.D.: The NEWUOA software for unconstrained optimization without derivatives. In: Large-scale nonlinear optimization, pp. 255–297. Springer (2006)
24. Regis, R.: Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. Engineering Optimization **46**(2), 218–243 (2014). https://doi.org/10.1080/0305215X.2013.765000
25. Rios, L., Sahinidis, N.: Derivative-free optimization: a review of algorithms and comparison of software implementations. Journal of Global Optimization **56**(3), 1247–1293 (2013). https://doi.org/10.1007/s10898-012-9951-y
26. Ros, R.: Benchmarking the NEWUOA on the bbob-2009 function testbed. In: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers. p. 2421–2428. GECCO '09, Association for Computing Machinery, New York, NY, USA (2009). https://doi.org/10.1145/1570256.1570338
27. Storn, R., Price, K.: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization **11**(4), 341–359 (1997). https://doi.org/10.1023/A:1008202821328
28. Torczon, V.: On the convergence of pattern search algorithms. SIAM Journal on Optimization **7**(1), 1–25 (1997). https://doi.org/10.1137/S1052623493250780
29. Tušar, T., Brockhoff, D., Hansen, N.: Mixed-integer benchmark problems for single- and bi-objective optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference. p. 718–726. GECCO '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3321707.3321868
30. Varelas, K., Auger, A., Brockhoff, D., Hansen, N., ElHara, O.A., Semet, Y., Kassab, R., Barbaresco, F.: A comparative study of large-scale variants of CMA-ES. In: Auger, A., Fonseca, C.M., Lourenço, N., Machado, P., Paquete, L., Whitley, D. (eds.) International Conference on Parallel Problem Solving from Nature. pp. 3–15. Springer, Springer International Publishing, Cham (2018)

31. Varelas, K., Dahito, M.A.: Benchmarking multivariate solvers of scipy on the noiseless testbed. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. p. 1946–1954. GECCO '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3319619.3326891