Les Cahiers du GERAD

Combining Cross Entropy and MADS methods for inequality constrained optimization

C. Audet,
J. Bigeon, R. Couderc

G–2020–59

November 2020

Combining Cross Entropy and MADS methods for inequality constrained optimization

Charles Audet [a,b]

Jean Bigeon [a,c]

Romain Couderc [a,b,c]

[a] GERAD, Montréal (Québec), Canada, H3T 2A7

[b] Department of Mathematics and Industrial Engineering, Polytechnique Montréal (Québec) Canada, H3C 3A7

[c] Université Grenoble Alpes, CNRS, Grenoble INP (Institute of Engineering Univ. Grenoble Alpes), Laboratoire G-SCOP, 38000 Grenoble, France

charles.audet@gerad.ca
jean.bigeon@grenoble-inp.fr
romain.couderc@grenoble-inp.fr

Abstract: This paper proposes a way to combine the Mesh Adaptive Direct Search (Mads) algorithm with the Cross-Entropy (CE) method for non smooth constrained optimization. The CE method is used as a SEARCH step by the Mads algorithm. The result of this combination retains the convergence properties of Mads and allows an efficient exploration in order to move away from local minima. The CE method samples trial points according to a multivariate normal distribution whose mean and standard deviation are calculated from the best points found so far. Numerical experiments show an important improvement of this method to reach the feasible region and to escape local minima.

1 Introduction

This work studies inequality constrained blackbox optimization problems of the form:

$$\min_{\mathbf{x} \in \Omega \subseteq \mathbb{R}^n} f(\mathbf{x}), \quad f : \mathbf{x} \in \mathbb{R}^n \to \mathbb{R} \tag{1}$$

with

$$\Omega = \{\mathbf{x} \in \mathcal{X} : c_j(\mathbf{x}) \leq 0, j = 1, 2, ..., m\},$$

where $f : \mathbb{R}^n \to \bar{\mathbb{R}} = \mathbb{R} \cup \pm\infty$, $c : \mathbb{R}^n \to \bar{\mathbb{R}}^m$ and the set $\mathcal{X}$ represents bound constraints of type $\boldsymbol{\ell} \leq \mathbf{x} \leq \mathbf{u}$ with $\boldsymbol{\ell}, \mathbf{u} \in \bar{\mathbb{R}}^n$. The specificity of this work is due to the form of the objective function f and of the constraints c_j. They can be the result of a simulation of complex physical phenomena. These simulations can take an important amount of time or present some discontinuities and therefore classical optimization methods are difficult to apply. Especially, when the gradient of the objective function and/or of the constraints are not explicitly known, hard to compute or its estimation is time consuming. This field is called derivative-free optimization (DFO). In the worst case, the gradient does not even exist, which is called blackbox optimization (BBO).

Specialized BBO and DFO algorithms have been developed in order to solve this kind of problem. There are two main categories: model based algorithms [14] and direct search algorithms [5]. This work deals with direct search algorithms which benefit from theoretical convergence results and adding some modifications may improve their performance. In particular, the Mesh Adaptive Direct Search (Mads) algorithm [2] ensures convergence to a point satisfying necessary conditions based on the Clarke calculus [12]. This theoretical guarantee is a solid basis for blackbox optimization. However, blackbox optimization algorithms must take into account two other types of difficulties. First, algorithms must be efficient in terms of simulation evaluations (constraints and objective function). Indeed, the simulation in engineering context is often time consuming. Second, blackbox simulations may involve multi-extrema functions, so algorithms must be able to escape from local minimum. Mads may be trapped in a local minimum.

To address the second difficulty, heuristics such as Variable Neighborhood Search (VNS) [1] and Latin Hypercube Sampling (LHS) [41] were proposed to escape from local minimum. Other methods of global optimization with no convergence guarantees exist, including Simulated Annealing [15], Genetic Algorithm [18] or Tabu search [37]. However, these heuristic methods often require a large number of function evaluations which is incompatible with the first difficulty. Finally, there exists also particular method as DIRECT algorithm [24] allowing to explore the space well. Nevertheless, here again the number of function evaluations seems to be prohibitive. In contrast, some methods have been developed to address the first problem, by reducing the overall number of simulation evaluation such as: the use of ensembles of surrogate [6] or of quadratic models [13] or the integration of the Nelder-Mead (NM) algorithm [9]. These different methods improve the efficiency of the Mads algorithm, nevertheless, they do not address the difficulty of local optima.

The objective of the present research is to propose an alternative SEARCH strategy: the Cross Entropy (CE) [40] method. This method is a trade-off between a more global search and a limited number of blackbox evaluations. It was introduced in 1997 , first in a context of rare event in discrete optimization [39] and then adapted to continuous optimization [26]. The main benefit of using CE is it often converges rapidly to a promising region in the space of variables. However, this method does not benefit from theoretical guarantees, and once that it has found a promising region, it requires a large number of simulation evaluations to improve the local accuracy. The two aims of this work are: global exploration with limited number of iterations and keep the convergence guarantee. In this purpose, CE is used as a SEARCH step. The combination we propose respect the local theoretical convergence guarantee and may hope to escape from local minima with few evaluations.This combination will benefit from CE's ability to identify a promising region, with Mads's ability to perform an efficient local descent.

This paper is divided as follow, Section 2 proposes an overview of the Mads and Cross Entropy methods. Section 3 presents an algorithm combining CE and Mads. Finally, Section 4 shows the main numerical results comparing the proposed method with Mads, Mads-LHS and Mads-VNS. Section 5 concludes on future work and on the contribution of this paper.

2   Description of Mads and Cross Entropy algorithms

In this Section, the Mads algorithm and the CE algorithm are described.

2.1   The Mads constrained optimization algorithm

The present work considers the Mads algorithm with the progressive barrier (PB) [3] to handle inequality constraints and with dynamic scaling [7] to handle the varying magnitudes of the variables. The PB uses the constraint violation function [16].

$$
h(\mathbf{x}) := \begin{cases} \sum_{j=1}^{m} (\max\{c_j(\mathbf{x}), 0\})^2 & \text{if } \mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n \\ \infty & \text{otherwise.} \end{cases}
$$

The constraints violation function value $h(\mathbf{x})$ is equal to 0 if and only if the point $\mathbf{x}$ belongs to Ω and is nonnegative otherwise.

Mads is a direct search algorithm. Each iteration includes two steps. The SEARCH step where various strategies may be used to explore the space of variables and the POLL step where the space of variable is locally explored by following strict rules guaranteeing convergence. In practice, the SEARCH accelerates the convergence to an optimum and it may attempt to escape from local minimum. The POLL is confined to a region delimited by the so called poll size vector $\mathbf{\Delta}^k \in \mathbb{R}_+^n$. All points generated by the POLL and SEARCH steps are rounded on a discretization of the space of the variables called the mesh whose the fineness is controlled by the mesh size vector $\boldsymbol{\delta}^k \in \mathbb{R}_+^n$. In its simplest form, the mesh [7] is defined as follows:

$$
M^k = V^k + \{diag(\boldsymbol{\delta}^k)\mathbf{z} : \mathbf{z} \in \mathbb{Z}^n\}
$$

where the cache V^k contains all the points visited by the start of iteration k. The mesh and the poll size vectors are updated at the end of each iteration. The values of both vectors are reduced when an iteration fails to improve the current solution and they are increased or remain at the same value otherwise. Algorithm 1 provides a description of Mads, the reader may consult [7] for complete details.

Algorithm 1: The Mesh Adaptive Direct Search algorithm (Mads)

Given a user-defined set of starting point: $V^0 \subset \mathbb{R}^n$,
and initial mesh and poll size vectors: $\boldsymbol{\delta}_i^0$ and $\mathbf{\Delta}_i^0$
Set the iteration counter: $k \leftarrow 0$.
1. Search step (optional):
   | Launch the simulation on a finite set S^k of mesh points.
   | If successful, go to 3.
2. Poll step:
   | Launch the simulation on the set P^k of poll points.
3. Updates:
   | Update the cache V^{k+1}, the incumbent $\mathbf{x}^{k+1}$
   | and the mesh and poll size vectors $\boldsymbol{\delta^{k+1}}$ and $\mathbf{\Delta^{k+1}}$.
   | Increase the iteration counter $k \leftarrow k + 1$ and go to 1.

The fundamental convergence result [3] of the Mads algorithm states that if the entire sequence of trial points belongs to a bounded set, then there exists an accumulation point $\mathbf{x}^*$ such that the generalized directional derivative $f^\circ(\mathbf{x}^*; \mathbf{d})$ of Clarke [12] is nonnegative in every hypertangent [22] direction $\mathbf{d}$ to the domain Ω at $\mathbf{x}^*$ provided that $\mathbf{x}^*$ is feasible. A similar result holds for the function h over the set X in situations where the iterates never approach the feasible region.

2.2 The Cross Entropy method for continuous optimization

The Cross Entropy method was introduced by Rubinstein in 1997 in the context of a minimization algorithm for estimating probabilities of rare events [39]. Later, it was modified to solve combinatorial optimization problems [40] and then in 2006 to solve continuous problems [26]. The main idea of this method is as follows. First, each optimization problem is transformed into a rare event estimation problem called associated stochastic problem (ASP). For instance, the deterministic problem (1) is transformed into:

$$P(f(\mathbf{X}) \leq \gamma) = E(I_{\{f(\mathbf{X}) \leq \gamma\}}) \tag{2}$$

where $\mathbf{X}$ is a random vector. Then, this ASP is tackled efficiently by an adaptive algorithm. This algorithm constructs a sequence of solutions which converging to the optimal solution of the ASP. The CE method is composed of two iterative steps:

- generation a sample of random data according to a density of probability;
- density parameters update thanks to the data sampled to create a new sample in the next iteration.

It results of this method is that it often escape from local minima.

2.2.1 An introductory example

For clarity, consider the example from [26] of minimizing the function:

$$f(x) = -e^{-(x-2)^2} - 0.8e^{-(x+2)^2}, x \in \mathbb{R}. \tag{3}$$

The function f has two local minima and a single global minimum at $x = 2$.
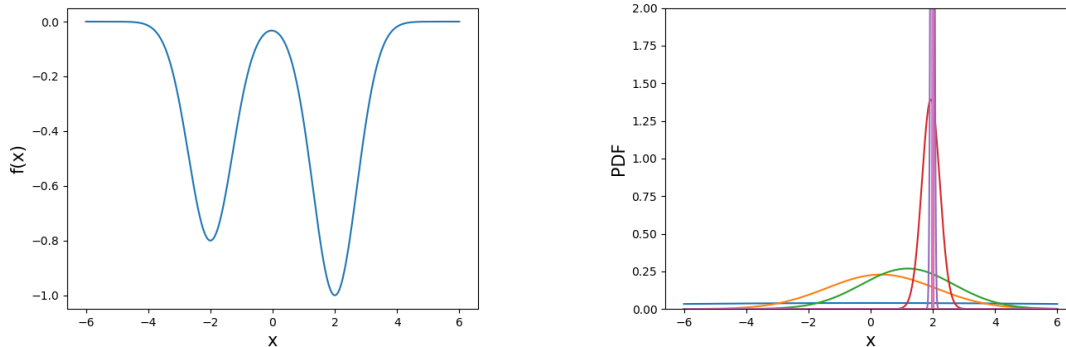
Figure 1: (Figure inspired by [26]) Graph of objective function f (left) and evolution of the normal distribution during the seven first iterations with $N_e = 10$ and $N_s = 50$ (right).

Using a normal distribution the CE procedure is the following:

- Initialization : at the first iteration $k = 0$, a mean $\boldsymbol{\mu}^0 \in \mathbb{R}^n$ and a standard deviation $\boldsymbol{\sigma}^0 \in \mathbb{R}^n$ (with n the dimension of the problem) are arbitrarily chosen. A large value of $\boldsymbol{\sigma}^0$ is taken in order to escape from local solutions.
- Iterative part: at each iteration $k \geq 1$:
  - First, a sample $\mathbf{X}_1, ..., \mathbf{X}_{N_s}$ of points in $\mathbb{R}^n$ is generated from a normal law $\mathcal{V}(\boldsymbol{\mu}^{k-1}, \boldsymbol{\sigma}^{k-1})$ where N_s is the number of samples.
  - Then, f is evaluated at each sampled points and a number of *elite* points N_e, with the lowest value of f. $\boldsymbol{\mu}^k$ and $\boldsymbol{\sigma}^k$ are the mean and standard deviation of these N_e points.
  - Termination: once the standard deviation becomes sufficiently small, the procedure is stopped.

The sequence of normal distribution is illustrated in the right part of Figure 1. This example shows how the CE procedure escapes from the local minimum at $x = -2$ and converges in seven iterations to the neighborhood of $x = 2$.

2.2.2   The general CE method

Before presenting the algorithm, the ASP is considered and the two iterative steps of the algorithm are precised. Problem 1 is transformed into an ASP. Using a family of probability distribution functions (pdf) $\{g(\cdot; \mathbf{v}) : \mathbf{v} \in \mathcal{V}\}$ where g is the law chosen to sample the different points at each iteration. $\mathcal{V}$ is the set of vector parameters of the pdf g which are calculated at each iteration. In the previous example g is taken as the normal law and the $v^k \in \mathcal{V}$ is composed of the mean and standard deviation $\mathbf{v}^k = (\boldsymbol{\mu}^k, \boldsymbol{\sigma}^k)$. Having explained the law and its parameters, the ASP related to problem 1 can be defined as follows:

$$P_{\mathbf{v}}(f(\mathbf{X}) \leq \gamma) = E_{\mathbf{v}}(I_{\{f(\mathbf{X}) \leq \gamma\}}) \tag{4}$$

where $\mathbf{v} \in \mathcal{V}$ is a vector of parameter, $\mathbf{X}$ is a random vector with a pdf $g(\cdot; \mathbf{v})$ and γ is a variable. At this stage, for a given value of γ, the parameter $\mathbf{v}$ may be estimated. Conversely, given a vector of parameters $\mathbf{v}$, the value γ may be also estimated. The CE method is based on these two estimations, at each iterations, the algorithm estimates one then the other. In the iterative part of Example 3, the first item corresponds to the estimation of γ and the second one to the estimation of $\mathbf{v}$. More precisely, we denote $\gamma^* \in \mathbb{R}$ as the infemum of the objective function, $\mathbf{v}^*$ the parameters and $g(\cdot; \mathbf{v}^*)$ the pdf associated to this infemum. The goal is to generate a sequence $(\gamma^k, \mathbf{v}^k)$ converging to $(\gamma^*, \mathbf{v}^*)$. To achieve this goal, a sequence of pdf $g(\cdot; \mathbf{v}^0), g(\cdot; \mathbf{v}^1), ...$ converging to $g(\cdot; \mathbf{v}^*)$ is created. To assure the convergence, one must have a "measure" of the difference between the iterate pdf $g(\cdot; \mathbf{v}^k)$ and the objective one $g(\cdot; \mathbf{v}^*)$. The Kullback-Leibler (KL) divergence [27] is used:

$$D(g(\cdot; \mathbf{v}^*) || g(\cdot; \mathbf{v}^k)) = \int_{-\infty}^{\infty} g(\mathbf{x}; \mathbf{v}^*) \ln \left(\frac{g(\mathbf{x}; \mathbf{v}^*)}{g(\mathbf{x}; \mathbf{v}^k)} \right) d\mathbf{x}. \tag{5}$$

The iterative steps may now be described. ρ is defined as a very small quantity, corresponding to the proportion of elite points which are kept from an iteration to another. The procedure is:

- **Adaptive update of γ^k.** With a fixed parameter of pdf $\mathbf{v}^{k-1}$, γ^k is defined such that it is the $(1 - \rho)$-quantile of $f(\mathbf{X})$ under v_{k-1}. Then, γ^k satisfies:

$$P_{\mathbf{v}^{k-1}}(f(\mathbf{X}) \leq \gamma^k) \geq \rho, \tag{6}$$

$$P_{\mathbf{v}^{k-1}}(f(\mathbf{X}) \geq \gamma^k) \geq 1 - \rho \tag{7}$$

  where $X \sim g(\cdot; \mathbf{v}^{k-1})$. The γ^k is denoted $\hat{\gamma}^k$. To obtain this estimator, a sample $\mathbf{X}_1, ..., \mathbf{X}_{N_s}$ is drawn from $g(\cdot; \mathbf{v}^{k-1})$ and evaluated. Then, the $(1 - \rho)$ quantile is:

$$\hat{\gamma}^k = f_{\lceil (1-\rho)N_s \rceil}. \tag{8}$$

- **Adaptive update of $\mathbf{v}^k$.** With a fixed γ^k and knowing $\mathbf{v}^{k-1}$, $\mathbf{v}^k$ is a solution of:

$$\begin{aligned} \max_{\mathbf{v}} D(\mathbf{v}) &= \max_{\mathbf{v}} E_{\mathbf{v}^{k-1}} I_{\{f(\mathbf{X}) \leq \gamma^k\}} \ln(g(\mathbf{X}; \mathbf{v})) \\ &= \min_{\mathbf{v}} E_{\mathbf{v}^{k-1}} I_{\{f(\mathbf{X}) \leq \gamma^k\}} \ln \left(\frac{I_{\{f(\mathbf{X}) \leq \gamma^k\}}}{g(\mathbf{X}; \mathbf{v})} \right) \end{aligned} \tag{9}$$

which is the minimization of the KL divergence at iteration k (with the convention $0 \ln(0) = 0$). Nevertheless, in practice, the real expectation and the real γ^k are not known, estimators must be used and the following equation is solved:

$$\tilde{\mathbf{v}}^k \in \operatorname*{argmax}_{\mathbf{v}} \widehat{D}(\mathbf{v}) = \max_{\mathbf{v}} \frac{1}{N_s} \sum_{i=1}^{N_s} I_{\{f(\mathbf{X}_i) \leq \hat{\gamma}^k\}} \ln(g(\mathbf{X}_i; v)). \tag{10}$$

Last but not least, $\hat{\mathbf{v}}^k$ is not set to $\tilde{\mathbf{v}}^k$. Indeed, some component of $\tilde{\mathbf{v}}^k$ could be set to 0 or 1 at the first few iterations and the transition between $\tilde{\mathbf{v}}^{k-1}$ and $\tilde{\mathbf{v}}^k$ could be discontinuous. To avoid these problems, the following convex combination is rather used:

$$\hat{\mathbf{v}}^k = \alpha\tilde{\mathbf{v}}^k + (1 - \alpha)\hat{\mathbf{v}}^{k-1} \tag{11}$$

with $0 < \alpha \leq 1$. Theoretically, any distribution converging in the neighborhood where the global maximum is attained can be used including normal, exponential or beta distribution. Nevertheless, in practice, the updating step is quite simple with the normal distribution that is often chosen. Therefore, the detailed algorithm is the following:

Algorithm 2: The Cross Entropy (CE) algorithm with normal law

---

Choose $\hat{\boldsymbol{\mu}}^0$ and $\hat{\boldsymbol{\sigma}}^0$
Set the iteration counter: $k \leftarrow 0$.
N_s number of sampled data at each iteration
N_e number of elite population
α the parameter of convex combination
1. Estimation of γ^k:
   Generate a random sample $\mathbf{X}_1, .., \mathbf{X}_{N_s}$ from $N(\hat{\boldsymbol{\mu}}^{k-1}, \hat{\boldsymbol{\sigma}}^{k-1})$ distribution.
   Evaluation of the N_s points by the simulation and then go to 2.
2. Estimation of mean and standard deviation
   Let E^k be the indices of the N_e best perfoming samples.
   Set $\tilde{\boldsymbol{\mu}}^k = \frac{1}{N_e} \sum_{i \in E^k} \mathbf{X}_i$
   and $(\tilde{\boldsymbol{\sigma}}^k)^2 = \frac{1}{N_e - 1} \sum_{i \in E^k} (\mathbf{X}_i - \boldsymbol{\mu}^k)^2$
3. Updates:
   Apply the convex combinations:
   $\hat{\boldsymbol{\mu}}^k = \alpha\tilde{\boldsymbol{\mu}}^k + (1 - \alpha)\hat{\boldsymbol{\mu}}^{k-1}$
   $\hat{\boldsymbol{\sigma}}^k = \alpha\tilde{\boldsymbol{\sigma}}^k + (1 - \alpha)\hat{\boldsymbol{\sigma}}^{k-1}$
   Increase the iteration counter $k \leftarrow k + 1$ and go to 1.

---

3 The CE-MADS constrained optimization algorithm

This section presents the CE-inspired SEARCH step of Mads. Section 3.1 describes how to handle constraints, the update of the mean $\boldsymbol{\mu}$ and the standard deviation $\boldsymbol{\sigma}$ and the condition to enter the CE search step. Section 3.2 presents the algorithm of the CE SEARCH step.

3.1 The CE-search step

3.1.1 Handling the constraints

Section 2 presented the CE method for unconstrained optimization. In [26], the bound constrained case is treated using a truncated normal law and a penalty approach is used for inequality constraints. In our work, the truncated normal law is also used to treat the bound constraints. For general inequality constraints, the algorithm does not use the penalty approach. In fact, when the algorithm chooses the elite sample, it uses the following function Best (defined in [9] and recalled here). Thus, any points in the cache may be selected even if its value of constraint violation is over the threshold of progressive barrier [3]. The definition relies on both the objective and the constraint violation functions f and h.

Definition 1 *The function* Best $: \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}^n$

$$\mathsf{Best}(\boldsymbol{x}, \boldsymbol{y}) = \begin{cases} \boldsymbol{x} & \text{if } \boldsymbol{x} \text{ dominates } \boldsymbol{y} \text{ or if } h(\boldsymbol{x}) < h(\boldsymbol{y}), \\ \boldsymbol{y} & \text{if } \boldsymbol{y} \text{ dominates } \boldsymbol{x} \text{ or if } h(\boldsymbol{y}) < h(\boldsymbol{x}), \\ Older(\boldsymbol{x}, \boldsymbol{y}) & \text{Otherwise} \end{cases}$$

returns the best *of two points.*

The function *Older* gives the point which was generated before the former one. Thanks to this definition, CE may treat the general inequality constraints with the terminology used in Mads.

3.1.2 Update of the mean and standard deviation

Three elements differ compared to classical CE method concerning the mean and the standard deviation. First, the elite points taken to compute the mean and the standard deviation are not only the N_s points generated by the normal law. The elite points are chosen from the cache at the iteration k, denotes $V^k \subset \mathbb{R}^n$, so any points generated by the Mads algorithm may be selected. This set is ordered with the function Best, in order to select the N_e elite points, it is sufficient to take the N_e first points of V^k.

Second, the mean and the standard deviation initialization procedure differs from the CE method proceeds. Indeed, Mads always begins with a starting point, thus there is at least one point in the cache (the set of evaluated points). Moreover, to avoid to field of exploration, bounds are always added on the problem as follows (using $\bar{\mathbf{x}}^k$ the poll center at iteration k):

$$\forall i \in [1,n] \ (\ell_i^k, u_i^k) = \begin{cases} (\ell_i, u_i) & \text{if } \ell_i \neq -\infty \text{ and } u_i \neq \infty, \\ (\bar{x}_i^k - 10 \times \Delta_i^k, u_i) & \text{if } \ell_i = -\infty \text{ and } u_i \neq \infty, \\ (\ell_i, \bar{x}_i^k + 10 \times \Delta_i^k) & \text{if } \ell_i \neq -\infty \text{ and } u_i = \infty. \end{cases} \tag{12}$$

Once the problem has finite bound constraints, there are two cases to calculate the mean and the standard deviation:

- In case where the number of points in the cache is too small to be relevant, i.e. fewer points that the number N_e required, then the mean and the standard deviation are determined such that:

$$\boldsymbol{\mu}^k = \bar{\mathbf{x}}^k \tag{13}$$

$$\boldsymbol{\sigma}^k = 2(\mathbf{u}^k - \boldsymbol{\ell}^k) \tag{14}$$

- In the others cases, the same calculations are made that in the original CE process:

$$\boldsymbol{\mu}^k = \frac{1}{N_e} \sum_{j \in E^k} \mathbf{X}_j$$

$$\boldsymbol{\sigma}^k = \sqrt{\frac{1}{N_e - 1} \sum_{j \in E^k} (\mathbf{X}_j - \boldsymbol{\mu}^k)^2}$$

Third, to generate the point during the CE search, the truncated normal law was always used with the bounds created in (12). Moreover, the elite points come not only from the previous normal sampling but also of the other kind of search step. That gives a vector of standard deviation which tends to zero very quickly, the other methods doing generally a local search. That is why, to avoid that the standard deviation is calculated as in (11) with a coefficient $\alpha = 0.7$.

3.1.3 The condition to pass in the CE-SEARCH step

The goal of the CE method is to explore in few evaluations the space to determine the promising region. The number of evaluations used by the CE SEARCH step must be quite small. For this purpose, Mads does not perform the CE SEARCH step at every iteration. The standard deviation can be seen as a measure of the incertitude on the data and is used to determine whether to launch the SEARCH step or not. First, a new variable called $\boldsymbol{\sigma}^p$ is introduced, it represents the incertitude measured the last time the algorithm passed through the CE SEARCH step and generated trial points. This variable is initialized to ∞. Then, the condition to launch the CE SEARCH is the following:

$$||\sigma^k|| < ||\sigma^p|| \tag{15}$$

This conditions means that the current incertitude is smaller than the previous one. Each time this conditions is respected, $\boldsymbol{\sigma}^p$ is updated with the standard deviation obtained after the CE step. Last but not least, there is a special case. The CE method being associate with Mads which is a local search, it is possible that the points become rapidly close to each others, reducing the standard deviation. In some cases, that avoids to escape from unfeasible region. Thus, in case where several iterations of Mads algorithm are passed and the feasible region is still not reached, then the CE-SEARCH is launched with a mean equal to the current best point and a standard deviation equal to 2 times the initial standard deviation until a feasible point is found.

3.2 The complete algorithm

The CE SEARCH step of Mads algorithm is presented here:

Algorithm 3: The CE SEARCH step

---

1. Calculation of $\boldsymbol{\mu}^k$ and $\boldsymbol{\sigma}^k$:
   if $card(V^k) < N_e$:
   $\quad \boldsymbol{\mu}^k = \bar{\mathbf{x}}_0$
   $\quad \boldsymbol{\sigma}^k$ update with 14
   else:
   $\quad \boldsymbol{\mu}^k = \frac{1}{N_e} \sum\limits_{j \in E^k} \mathbf{X}_j$
   $\quad (\boldsymbol{\sigma}^k)^2 = \frac{1}{N_e - 1} \sum\limits_{j \in E^k} (\mathbf{X}_j - \boldsymbol{\mu}^k)^2$
2. CE SEARCH
   If $||\sigma^k|| < ||\sigma^p||$:
   $\quad$ Generate a random sample $\mathbf{X}_1, .., \mathbf{X}_{N_s}$ from $\mathcal{N}(\boldsymbol{\mu}^k, 2\boldsymbol{\sigma}^k)$ distribution
   $\quad$ and project them on the mesh.
   $\quad$ Evaluation of the N_s points by the simulation.
   $\quad$ Update:
   $\quad \boldsymbol{\mu}^{k+1} = \frac{1}{N_e} \sum\limits_{j \in E^k} \mathbf{X}_j$
   $\quad (\boldsymbol{\sigma}^{k+1})^2 = \frac{1}{N_e - 1} \sum\limits_{j \in E^k} (\mathbf{X}_j - \boldsymbol{\mu}^{k+1})^2$
   $\quad (\boldsymbol{\sigma}^p)^2 = (\boldsymbol{\sigma}^{k+1})^2$

---

4 Computational experiments

The present work uses data profiles to compare the different algorithm. Data profiles [35] allow to assess if algorithms are successful in generating solution values close to the best objective function values. To identify a successful run, a convergence test is required. Let denote $\mathbf{x}_e$ the best iterates obtained by one algorithm on one problem after e evaluations, f_{fea} a common reference for a given problem obtained by taking the max feasible objective function values on all run instances of that problem for all algorithms and f^* the best solution obtained by all tested algorithms on all run instances of that problem. Then, the problem is said to be solved within the convergence tolerance τ when:

$$f_{fea} - f(\mathbf{x}_e) \geq (1 - \tau)(f_{fea} - f^*).$$

Different initial points constitute different problems. Moreover, an instance of a problem corresponds to a particular pseudo-random generator seeds. The horizontal axis of a data profile represents the number of evaluations for problems of fixed dimension, and represents group of $n + 1$ evaluations when problems of different dimension are involved. The vertical axis corresponds to the proportion of problems solved within a given tolerance τ. Each algorithm has its curve to allow comparison of algorithms capability to converge to the best objective function value.

This section presents the numerical experiments. It is divided in two subsections. The numerical experiments of Section 4.1 are performed on analytical test problems to calibrate the CE-SEARCH parameters. Section 4.2 compares Mads, LH-Mads, VNS-Mads and CE-Mads without the use of models on three real engineering problems.

4.1   Preliminary experiments to calibrate parameters

Computational experiments are conducted using the version 3.9.1 of NOMAD [28] software package. All tests use the Mads strategy with the use of the NM search [9] and without the use of models. When the CE-SEARCH is used, it is the first SEARCH step to be applied.

Numerical experiments on analytical test problems are conducted to set default values for the two algorithmic parameters: the number of sampled data at each iteration N_s and the number of elite population N_e. CE-Mads is tested on 100 analytical problems from the optimization literature. The characteristics and sources of these problems are summarized in Table 1 in appendix A. The number of variables ranges from 2 to 60; 28 problems have constraints other than bound constraints. In order to have a more precise idea of the effect between the hyper-parameters (n_e and n_s), three series of tests are conducted:

- A series of tests on the 69 unconstrained test problems having a dimension from 2 to 20.
- A series of tests on the 25 constrained test problems having a dimension from 2 to 20.
- A series of tests on the 6 larger problems in term of dimension (from 50 to 60), three are constrained and three are not.

For each test, the maximal number of function evaluations is set to $1000(n + 1)$, where n is the number of variables and each problem is run with 3 different random seeds. For each series of tests, the five following CE-MADS setup of hyper-parameters are compared: $(N_e, N_s) \in \{(2, n), (4, 2n), (6, 3n), (8, 4n), (10, 5n)\}$ with n the dimension of the test problem. A run called NO-MAD default is added in each series of test to compare our results with the current NOMAD software. Data profiles are presented on Figure 2, 3 and 4 with different values of the tolerance τ.

These results are analysed by series of problems:

- On the unconstrained problems (see Figure 2), no algorithm really stands out regardless of the value of τ, it is difficult to choose one hyper-parameter rather than another one even if the couple $N_e = 4$ and $N_s = 2n$ appears to be more efficient.
- On the constrained problems (see Figure 3), there are different behaviors according to the value of τ. For $\tau = 10^{-3}$, no algorithm appears to be dominant. However, for $\tau = 10^{-5}$, it happens that greater are the values of N_s and N_e, higher is the percentage of problems solved finally. That can be explained because great N_s and N_e allow a better exploration of the space, and so a more precise result at the end.
- Finally (see Figure 4), on the large test problems, and for small values of the tolerance τ the CE-Mads is outperformed by the Mads algorithm with default values. It seems that the CE method is not useful for problems with a large number of variables.

Inspection of the logs of the hyper-parameter calibration reveals the two following observations:

- The CE-Mads performance is not very sensitive to the hyper-parameter values. This allows to avoid some calibration experiments before applying the algorithm on a new test problem.
- For problems with a large number of variables, our tests suggest to avoid using of the CE-SEARCH. Nevertheless, this point has not been confirmed on real engineering problems given that we do not have access to engineering test problems with large dimension.

In the remainder of the paper, the CE-SEARCH values are set to $N_e = 4$ and $N_s = 2n$ as they often perform well.

4.2   Test on engineering problems

In this section, the CE-Mads algorithm is tested on three different engineering problems. Its results are compared with three algorithms: the Mads-default (without models), the VNS-Mads where a VNS-SEARCH is used and the LH-Mads which is a default Mads with in addition a LHS search. The
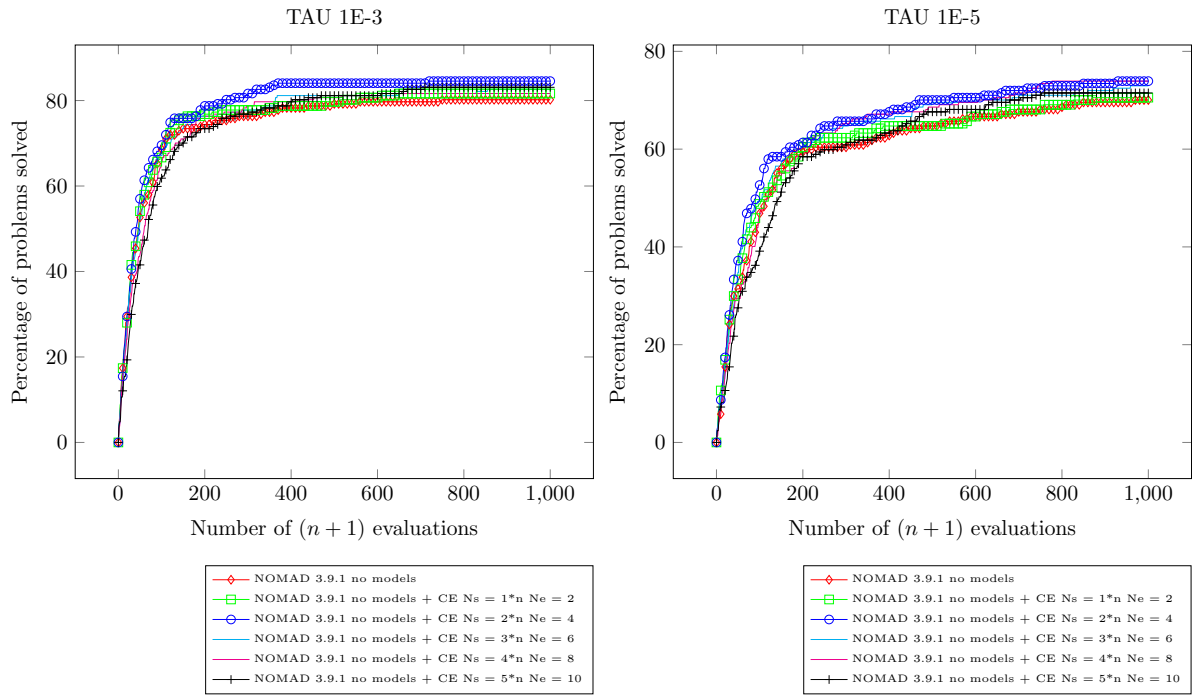
TAU 1E-3

TAU 1E-5

Figure 2: Result of calibration of the hyper-parameters of CE-MADS on the 69 unconstrained test problems.

TAU 1E-3

TAU 1E-5
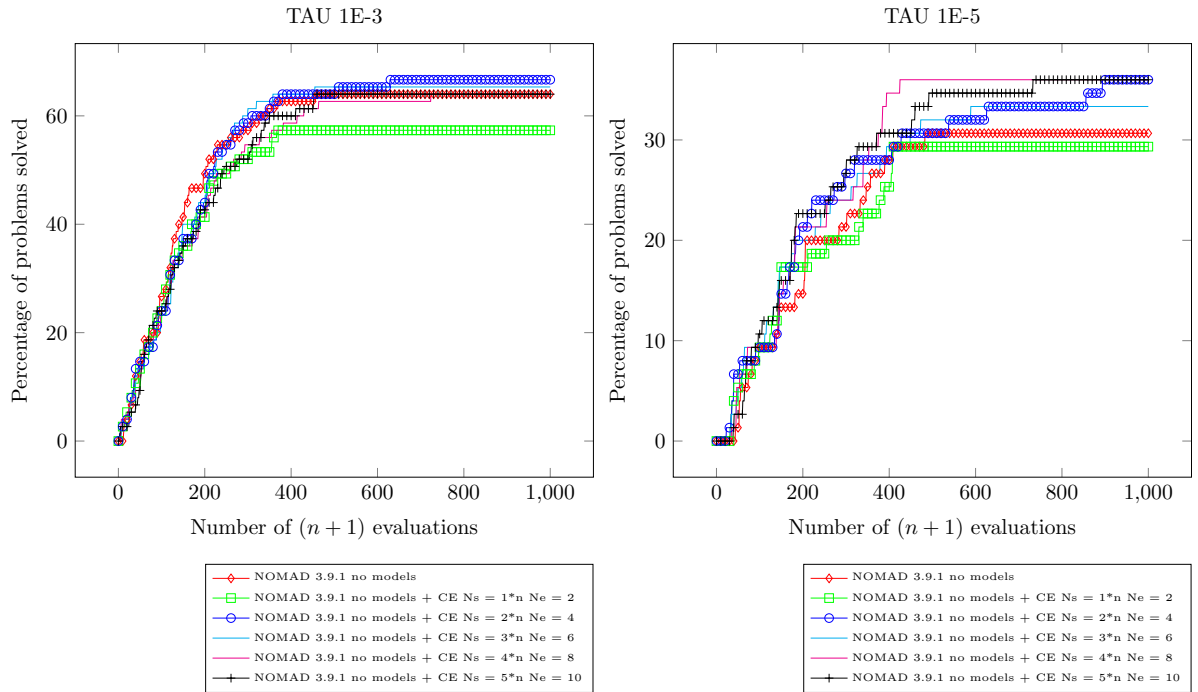
Figure 3: Result of calibration of the hyper-parameter of CE-MADS on the 25 constrained test problems.

comparison with the two last algorithms is crucial because they are methods aiming to explore the space of design variables. The Latin Hypercube SEARCH strategy is used with two parameters $n_{init} = 100$ and $n_{iter} = 10$: n_{init} is the number of LH trial points generated at the first iteration of Mads and n_{iter} the number of LH trial points generated at each subsequent iteration. The Variable Neighbour Search is used with the default parameters [1]. It is an metaheuristic allowing to explore distant neighborhoods of the current incumbent solution.

Figure 4: Result of calibration of the hyper-parameter of CE-MADS on the 6 large test problems.

4.2.1 The MDO problems

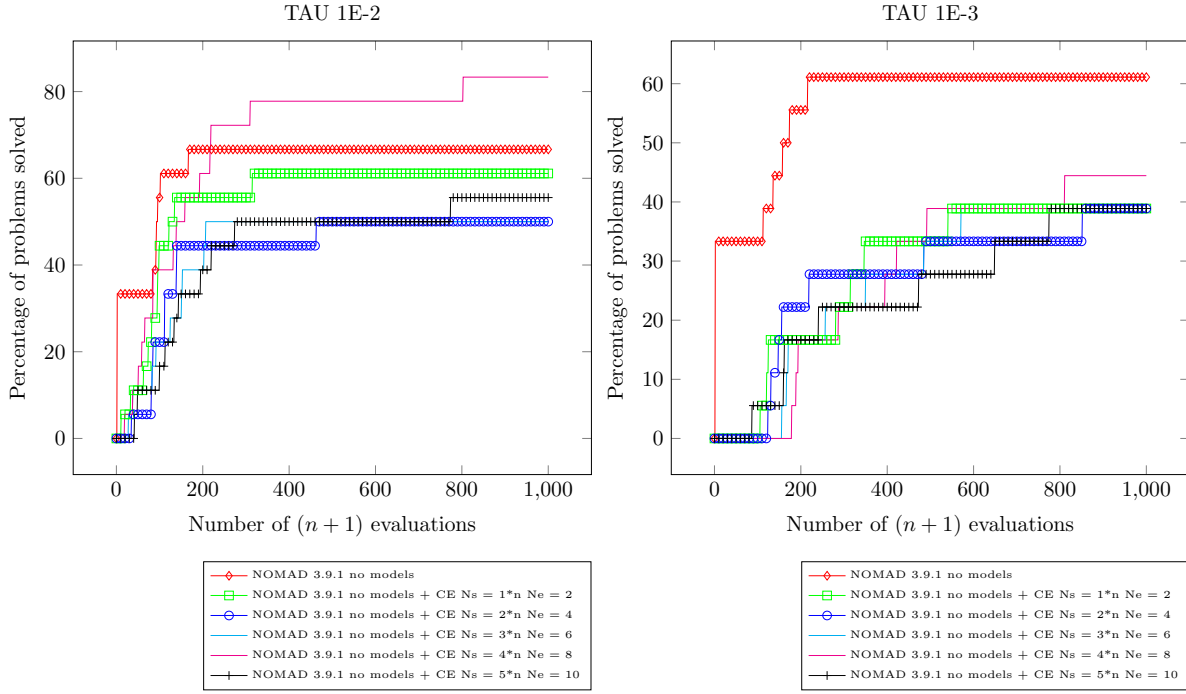The Mads-default (no models), CE-Mads, VNS-Mads and LH-Mads are tested to solve a simple multidisciplinary wing design optimization problem [17]. Each initial point defines a MDO problem. Solving the problem consists in maximizing the range of an aircraft subject to 10 general constraints. The problem has 10 scaled design variables bounded in [0; 100]. Figure 5 shows the result on a data profile when solving 20 MDO problems on different initial points using 3000 function evaluations or less. The initial points are real randomly selected within the bounds. Each run is done with three different seeds in order to minimize the impact of the seed.

Figure 5 shows that the CE-Mads outperforms the other algorithms with all values of τ.

4.2.2 The STYRENE problems

The Mads-default (no models), CE-Mads, VNS-Mads and LH-Mads algorithms are tested to optimize a styrene production process [1]. This problem is a simulation of a chemical process. This process relies on a series of interdependent calculation of blocks using common numerical tools as Runge-Kutta, Newton, fixed point and also chemical related solver. The particularity of this problem is the presence of "hidden" constraints, i.e. sometimes the process does not finish and just return an error. In the case where the chemical process ends, the constraints (not hidden) and the objective functions may be evaluated during a post-processing. The objective is to maximize the net value of the styrene production process with 9 industrial and environmental regulations constraints.

In this work, a STYRENE problem possesses eight independent variables influencing the styrene production process. The variables considered during the optimization process are all scaled and bounded in $X = [0, 100]^8$. As it was done for the MDO test problems, the four algorithms are tested with 20 different starting points taken in X. A maximal number of evaluations of 3000 is used and each problem is run with three different seeds. The STYRENE problems is particularly interesting in this study, because there are two minima as it is shown in [9]. The results with $\tau = 10^{-1}$ allow to know the percentage of problems having found the global minimum. The results are provided on Figure 6. On the left plot, it is interesting to notice that the CE-Mads algorithm find the global

minimum the same number of times that the LH-Mads algorithm but is more efficient. On the right plot, the CE-Mads algorithm seems to have the same accuracy that the VNS-Mads algorithm and is slightly more efficient.
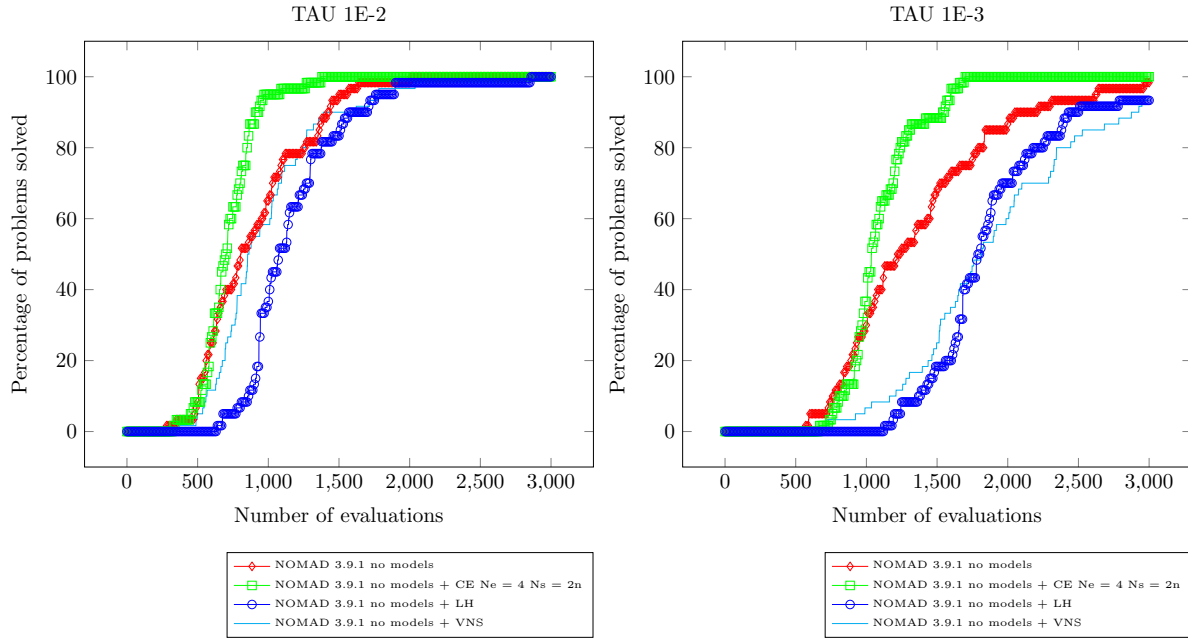

Figure 5: Result on the 60 MDO instances between Mads **(no models),** CE-Mads, VNS-Mads **and** LH-Mads **for** $\tau = 10^{-2}$ **(left) and** $\tau = 10^{-3}$ **(right).**
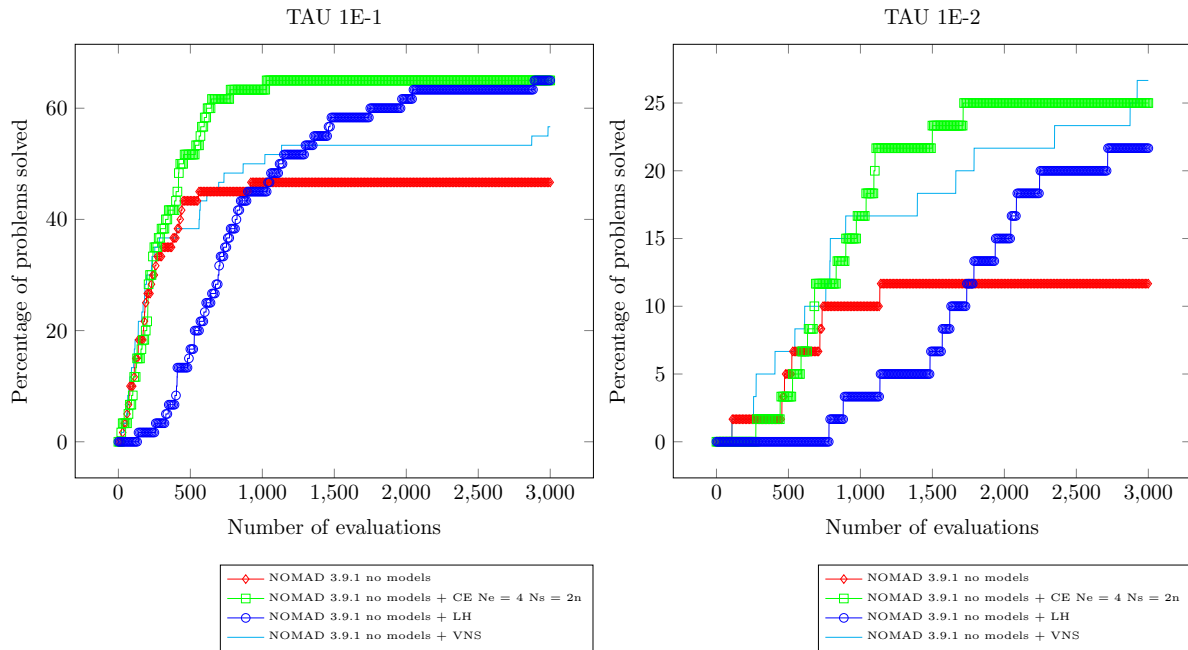

Figure 6: Result on the 60 STYRENE instances between Mads **(no models),** CE-Mads **and** LH-Mads **for** $\tau = 10^{-1}$ **(left) and** $\tau = 10^{-2}$ **(right).**

4.2.3 The LOCKWOOD problems

Finally, the Mads default (no models), LH-Mads and CE-Mads algorithms without quadratic models are tested to solve the basic version of a pump-and-treat groundwater remediation problem from Montana Lockwood Solvent Groundwater Plume Site [30]. The problem has 6 design variables bounded in $X = [0, 20000]^6$ and 4 constraints. A particularity of this problem is that each simulation run take several seconds, so the maximum number of blackbox evaluations is set to 1500. The algorithms are started from 20 different randomly selected initial points in X and three different seeds are used as previously. The results are provided on Figure 7.

In this problem, reach the feasible region is not easy. Here again, the results at $\tau = 10^{-1}$ allows to give an idea of the number of times the algorithm reach the feasible region. For instance, CE-Mads and LH-Mads always reach the feasible region while Mads default reaches the feasible only 41 times on 60 instances and VNS-Mads only 46 times. The efficiency of CE-Mads and LH-Mads is comparable. However, on the right plot, a better accuracy is reached with a greater efficiency by the CE-Mads algorithm.
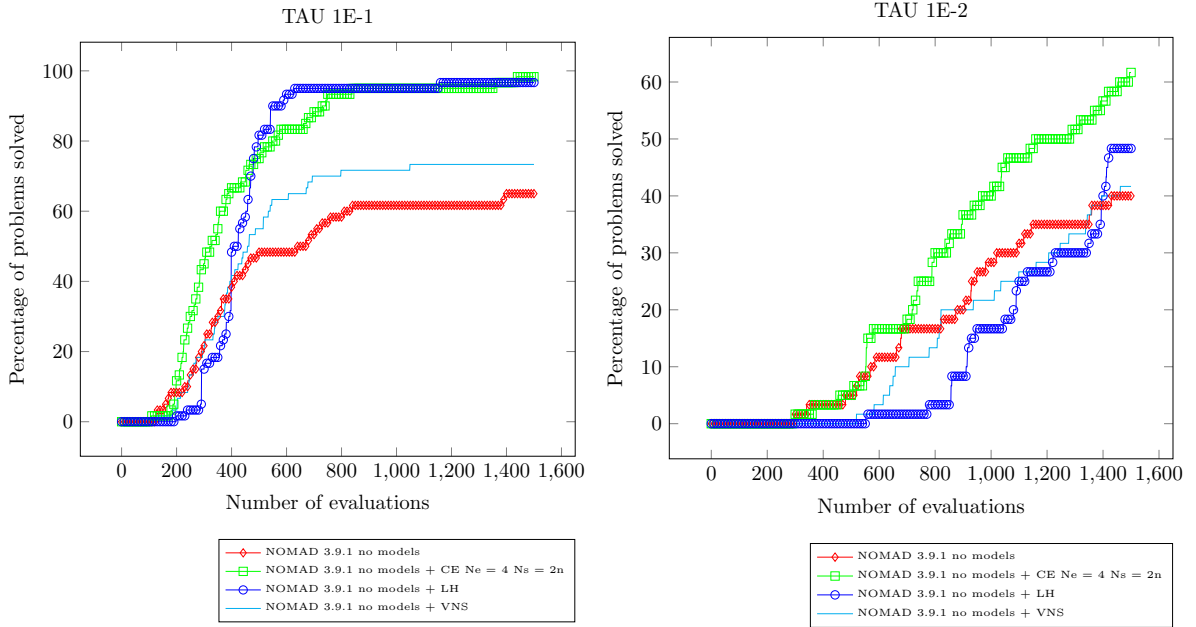

Figure 7: Result on the 20 LOCKWOOD instances between Mads-**default (no models),** CE-Mads, **VNS**-Mads **and LH**-Mads **for** $\tau = 10^{-1}$ **(left) and** $\tau = 10^{-2}$ **(right).**

5 Discussion

This paper introduces a way to combine the CE algorithm and the Mads algorithm so that it allows a better space exploration. This is achieved by defining a CE-SEARCH step within the Mads algorithm. The CE SEARCH generates some points according to a normal distribution whose mean and standard deviation is calculated from the best points stored in the cache. This approach allows to handle the constraints in a different ways. Moreover, the particularity of this SEARCH is that it is not performed at each iteration of the Mads algorithm, but according to a criterion based on the value of the norm of the standard deviation of the best points.

Numerical experiments show that in case where the problem has different minima or a feasible region hard to reach, the CE-Mads algorithm performs well. Indeed, it attains as often as the LH-Mads the feasible region or the global minimum but it is far more efficient, especially when a tight accuracy is considered. Finally, even on problem, as MDO, where the classical exploration SEARCH, LH and VNS, does not work well, the CE-Mads algorithm gives interesting results.

Further works will be devoted to improve the link between the Mads algorithm and the CE algorithm by adjusting the size of the mesh with the standard deviation calculated in CE.

A   Appendix

#	Name	Source	n	m	Bnds	#	Name	Source	n	m	Bnds
1	ARWHEAD10	[19]	10	0	no	51	MXHILB	[29]	50	0	no
2	ARWHEAD20	[19]	20	0	no	52	OPTENG_RBF	[25]	3	4	yes
3	BARD	[34]	3	0	no	53	OSBORNE1	[34]	5	0	no
4	BDQRTIC10	[19]	10	0	no	54	OSBORNE2	[29]	11	0	no
5	BDQRTIC20	[19]	20	0	no	55	PBC1	[29]	5	0	no
6	BEALE	[34]	2	0	no	56	PENALTY1_4	[19]	4	0	no
7	BIGGS	[19]	6	0	no	57	PENALTY1_10	[19]	10	0	no
8	BOX	[34]	3	0	no	58	PENALTY1_20	[19]	20	0	no
9	BRANIN	[20]	2	0	yes	59	PENALTY2_4	[19]	4	0	no
10	BROWNAL5	[19]	5	0	no	60	PENALTY2_10	[19]	10	0	no
11	BROWNAL7	[19]	7	0	no	61	PENALTY2_20	[19]	20	0	no
12	BROWNAL10	[19]	10	0	no	62	PENTAGON	[29]	6	15	no
13	BROWNAL20	[19]	20	0	no	63	PIGACHE_X00	[36]	4	11	yes
14	BROWNDENNIS	[34]	4	0	no	64	PIGACHE_X01	[36]	4	11	yes
15	BROWN_BS	[34]	2	0	no	65	POLAK2	[29]	10	0	no
16	B250	[10]	60	1	yes	66	POWELL_BS	[34]	2	0	no
17	B500	[10]	60	1	yes	67	POWELLSG4	[19]	4	0	no
18	CHENWANG_F2_X0	[11]	8	6	yes	68	POWELLSG8	[19]	8	0	no
19	CHENWANG_F2_X1	[11]	8	6	yes	69	POWELLSG12	[19]	12	0	no
20	CHENWANG_F3_X0	[11]	10	8	yes	70	POWELLSG20	[19]	20	0	no
21	CHENWANG_F3_X1	[11]	10	8	yes	71	RADAR	[33]	7	0	yes
22	CRESCENT	[3]	10	2	no	72	RANA	[23]	2	0	yes
23	DISK	[3]	10	1	no	73	RASTRIGIN	[20]	2	0	yes
24	DIFFICULT2	[3]	10	0	no	74	RHEOLOGY	[5]	3	0	no
25	ELATTAR	[29]	6	0	no	75	ROSENBROCK	[34]	2	0	yes
26	EVD61	[29]	6	0	no	76	SHOR	[29]	5	0	no
27	FILTER	[29]	9	0	no	77	SNAKE	[3]	2	2	no
28	FREUDENSTEINROTH	[34]	2	0	no	78	SPRING_X00	[38]	3	4	yes
29	GAUSSIAN	[34]	3	0	no	79	SPRING_X01	[38]	3	4	yes
30	G2_10	[4]	10	2	yes	80	SROSENBR6	[19]	6	0	no
31	G2_20	[4]	20	2	yes	81	SROSENBR8	[19]	8	0	no
32	G2_50	[4]	50	2	yes	82	SROSENBR10	[19]	10	0	no
33	GOFFIN	[29]	50	0	no	83	SROSENBR20	[19]	20	0	no
34	GRIEWANK	[20]	10	0	yes	84	TAOWANG_F2_X00	[42]	7	4	yes
35	GULFRD	[8]	3	0	no	85	TAOWANG_F2_X01	[42]	7	4	yes
36	HELICALVALLEY	[34]	3	0	no	86	TREFETHEN	[23]	2	0	yes
37	HS19	[21]	2	2	yes	87	TRIDIA10	[19]	10	0	no
38	HS78	[29]	5	0	no	88	TRIDIA20	[19]	20	0	no
39	HS83_X0	[21]	5	6	yes	89	TRIGONOMETRIC	[34]	10	0	no
40	HS83_X1	[21]	5	6	yes	90	VARDIM8	[19]	8	0	no
41	HS114_X0	[29]	9	6	yes	91	VARDIM10	[19]	10	0	no
42	HS114_X1	[29]	9	6	yes	92	VARDIM20	[19]	20	0	no
43	JENNRICHSAMPSON	[34]	2	0	no	93	WANGWANG_F3	[43]	2	0	yes
44	KOWALIKOSBORNE	[34]	4	0	no	94	WATSON9	[34]	9	0	no
45	L1HILB	[29]	50	0	no	95	WATSON12	[34]	12	0	yes
46	MAD6_X0	[29]	5	7	no	96	WONG1	[29]	7	0	no
47	MAD6_X1	[29]	5	7	no	97	WONG2	[29]	10	0	no
48	MCKINNON	[31]	2	0	no	98	WOODS4	[19]	4	0	no
49	MEYER	[34]	3	0	no	99	WOODS12	[19]	12	0	no
50	MEZMONTES	[32]	2	2	yes	100	WOODS20	[19]	20	0	no

Table 1: Description of the set of 100 analytical problems.

Nomenclature

The following list describes symbols used within the body of the document. In what follows, if the symbol is bold then it is a vector otherwise it is a scalar.

$\boldsymbol{\ell}$ The lower bound of a decision variable

Δ^k The frame size parameter at iteration k

δ^k The mesh size parameter at iteration k

ϵ The stopping criterion

γ A parameter to estimate in an associated stochastic problem

$\mathbf{u}$ The upper bound of a decision variable

$\mathcal{E}$ The expectation

$\mathcal{N}$ The normal distribution

$\mathcal{V}$ Set of parameters of a probability density function

$\mathcal{X}$ The bounded constraints set of type $\ell \leq x \leq u$

μ The mean

Ω The feasible set

ρ A percentage of quantile

σ The standard deviation

τ The mesh size adjustment parameter

$\mathbf{X}$ A random vector

c_j The jth constraint

D A positive spanning set

E^k The set of indices of elite points

F^k The frame at iteration k

$g(\cdot;\cdot)$ A probability density function

h The measure of constraints violation

I_x Indicator function of x

k The iteration counter

M^k The mesh at iteration k

n The dimension of a problem

N_s Number of sampled data at each iteration

N_e Number of elite population

V The cache

v A parameter of a probability density function

References

[1] Audet, C., Béchard, V., Le Digabel, S.: Nonsmooth optimization through Mesh Adaptive Direct Search and Variable Neighborhood Search. Journal of Global Optimization 41(2), 299–318 (2008). DOI 10.1007/s10898-007-9234-1. URL http://dx.doi.org/doi:10.1007/s10898-007-9234-1

[2] Audet, C., Dennis, Jr., J.: Mesh Adaptive Direct Search Algorithms for Constrained Optimization. SIAM Journal on Optimization 17(1), 188–217 (2006). DOI 10.1137/040603371. URL http://dx.doi.org/doi:10.1137/040603371

[3] Audet, C., Dennis, Jr., J.: A Progressive Barrier for Derivative-Free Nonlinear Programming. SIAM Journal on Optimization 20(1), 445–472 (2009). DOI 10.1137/070692662. URL http://dx.doi.org/10.1137/070692662

[4] Audet, C., Dennis, Jr., J., Le Digabel, S.: Parallel Space Decomposition of the Mesh Adaptive Direct Search Algorithm. SIAM Journal on Optimization 19(3), 1150–1170 (2008). DOI 10.1137/070707518. URL http://dx.doi.org/10.1137/070707518

[5] Audet, C., Hare, W.: Derivative-Free and Blackbox Optimization. Springer Series in Operations Research and Financial Engineering. Springer International Publishing, Berlin (2017). DOI 10.1007/978-3-319-68913-5. URL https://dx.doi.org/10.1007/978-3-319-68913-5

[6] Audet, C., Kokkolaras, M., Le Digabel, S., Talgorn, B.: Order-based error for managing ensembles of surrogates in mesh adaptive direct search. Journal of Global Optimization 70(3), 645–675 (2018). DOI 10.1007/s10898-017-0574-1. URL http://rdcu.be/wGt6

[7] Audet, C., Le Digabel, S., Tribes, C.: Dynamic scaling in the mesh adaptive direct search algorithm for blackbox optimization. Optimization and Engineering 17(2), 333–358 (2016). DOI 10.1007/s11081-015-9283-0. URL http://dx.doi.org/10.1007/s11081-015-9283-0

[8] Audet, C., Le Digabel, S., Tribes, C.: The Mesh Adaptive Direct Search Algorithm for Granular and Discrete Variables. SIAM Journal on Optimization 29(2), 1164–1189 (2019). DOI 10.1137/18M1175872. URL https://doi.org/10.1137/18M1175872

[9] Audet, C., Tribes, C.: Mesh-based Nelder-Mead algorithm for inequality constrained optimization. Computational Optimization and Applications 71(2), 331–352 (2018). DOI 10.1007/s10589-018-0016-0. URL https://link.springer.com/article/10.1007/s10589-018-0016-0

[10] Booker, A., Cramer, E., Frank, P., Gablonsky, J., Dennis, Jr., J.: Movars: Multidisciplinary optimization via adaptive response surfaces. AIAA Paper 2007–1927 (2007). URL http://pdf.aiaa.org/preview/CDReadyMSDM07_1473/PV2007_1927.pdf

[11] Chen, X., Wang, N.: Optimization of short-time gasoline blending scheduling problem with a DNA based hybrid genetic algorithm. Chemical Engineering and Processing: Process Intensification 49(10), 1076–1083 (2010). DOI 10.1016/j.cep.2010.07.014. URL http://dx.doi.org/10.1016/j.cep.2010.07.014

[12] Clarke, F.: Optimization and Nonsmooth Analysis. John Wiley & Sons, New York (1983). URL http://www.ec-securehost.com/SIAM/CL05.html. Reissued in 1990 by SIAM Publications, Philadelphia, as Vol. 5 in the series Classics in Applied Mathematics

[13] Conn, A., Le Digabel, S.: Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. Optimization Methods and Software 28(1), 139–158 (2013). DOI 10.1080/10556788.2011.623162. URL http://dx.doi.org/10.1080/10556788.2011.623162

[14] Conn, A., Scheinberg, K., Vicente, L.: Introduction to Derivative-Free Optimization. MOS-SIAM Series on Optimization. SIAM, Philadelphia (2009). DOI 10.1137/1.9780898718768. URL http://dx.doi.org/10.1137/1.9780898718768

[15] Coranan, A., Marchesi, M., Martini, C., Ridella, S.: Minimizing multimodel functions of continuous variables with "Simulated Annealing" algorithm. Transactions on Mathematical Software 13(3), 262–280 (1987)

[16] Fletcher, R., Leyffer, S.: Nonlinear programming without a penalty function. Mathematical Programming Series A, 91, 239–269 (2002). DOI 10.1007/s101070100244. URL http://dx.doi.org/10.1007/s101070100244

[17] Giunta, A.: Aircraft multidisciplinary optimization using design of experiments theory and response surface modeling methods. Ph.D. thesis, Virginia Tech, Houston, Texas (1997; available as Tech. Rep. MAD 97-05-01, May 1997, Department of Aerospace and Ocean Engineering, Virginia Tech, 215 Randolph Hall, Blacksburg, Virginia 24061)

[18] Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1989)

[19] Gould, N., Orban, D., Toint, P.: CUTEr (and SifDec): A constrained and unconstrained testing environment, revisited. ACM Transactions on Mathematical Software 29(4), 373–394 (2003). DOI 10.1145/962437.962439. URL http://dx.doi.org/10.1145/962437.962439

[20] Hedar, A.R.: Global Optimization Test Problems. http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO.htm. URL http://goo.gl/0vxil. (last accessed on 2017-10-20)

[21] Hock, W., Schittkowski, K.: Test Examples for Nonlinear Programming Codes, Lecture Notes in Economics and Mathematical Systems, vol. 187. Springer, Berlin, Germany (1981)

[22] Jahn, J.: Introduction to the Theory of Nonlinear Optimization. Springer, Berlin (1994). URL http://www.springer.com/mathematics/book/978-3-540-49378-5

[23] Jamil, M., Yang, X.S.: A literature survey of benchmark functions for global optimisation problems. International Journal of Mathematical Modelling and Numerical Optimisation 4(2), 150–194 (2013). DOI 10.1504/IJMMNO.2013.055204. URL http://dx.doi.org/10.1504/IJMMNO.2013.055204

[24] Jones, D., Perttunen, C., Stuckman, B.: Lipschitzian optimization without the Lipschitz constant. Journal of Optimization Theory and Application 79(1), 157–181 (1993). DOI 10.1007/BF00941892. URL http://dx.doi.org/10.1007/BF00941892

[25] Kitayama, S., Arakawa, M., Yamazaki, K.: Sequential approximate optimization using radial basis function network for engineering optimization. Optimization and Engineering 12(4), 535–557 (2011). DOI 10.1007/s11081-010-9118-y. URL http://dx.doi.org/10.1007/s11081-010-9118-y

[26] Kroese, D.P., S.Porotsky, Rubinstein, R.Y.: The Cross-Entropy method for continuous and multi-extremal optimization. Metodol Comput Appl Probab 8, 383–407 (2006). DOI 10.1007/s11009-006-9753-0

[27] Kullback, S., R.Leibler: On information and sufficiency. Annals of Mathematical Statistics 22, 79–86 (1951)

[28] Le Digabel, S.: Algorithm 909: NOMAD: Nonlinear Optimization with the MADS algorithm. ACM Transactions on Mathematical Software 37(4), 44:1–44:15 (2011). DOI 10.1145/1916461.1916468. URL http://dx.doi.org/10.1145/1916461.1916468

[29] Lukšan, L., Vlček, J.: Test problems for nonsmooth unconstrained and linearly constrained optimization. Tech. Rep. V–798, ICS AS CR (2000). URL http://www.cs.cas.cz/ics/reports/v798-00.ps

[30] Matott, L., Rabideau, A., Craig, J.: Pump-and-treat optimization using analytic element method flow models. Advances in Water Resources 29(5), 760–775 (2006). DOI 10.1016/j.advwatres.2005.07.009. URL http://dx.doi.org/10.1016/j.advwatres.2005.07.009

[31] McKinnon, K.: Convergence of the Nelder-Mead simplex method to a nonstationary point. SIAM Journal on Optimization 9(1), 148–158 (1998). DOI 10.1137/S1052623496303482. URL https://dx.doi.org/10.1137/S1052623496303482

[32] Mezura-Montes, E., Coello, C.: Useful Infeasible Solutions in Engineering Optimization with Evolutionary Algorithms. In: Proceedings of the 4th Mexican International Conference on Advances in Artificial Intelligence, MICAI'05, pp. 652–662. Springer-Verlag, Berlin, Heidelberg (2005). DOI 10.1007/11579427_66. URL http://dx.doi.org/10.1007/11579427_66

[33] Mladenović, N., Petrović, J., Kovačević-Vujčić, V., Čangalović, M.: Solving spread spectrum radar polyphase code design problem by tabu search and variable neighbourhood search. European Journal of Operational Research 151(2), 389–399 (2003). DOI 10.1016/S0377-2217(02)00833-0. URL http://dx.doi.org/10.1016/S0377-2217(02)00833-0

[34] Moré, J., Garbow, B., Hillstrom, K.E.: Testing unconstrained optimization software. ACM Transactions on Mathematical Software 7(1), 17–41 (1981). DOI 10.1145/355934.355936. URL http://dx.doi.org/10.1145/355934.355936

[35] Moré, J., Wild, S.: Benchmarking derivative-free optimization algorithms. SIAM Journal on Optimization 20(1), 172–191 (2009). DOI 10.1137/080724083. URL http://dx.doi.org/10.1137/080724083

[36] Pigache, F., Messine, F., Nogarede, B.: Optimal Design of Piezoelectric Transformers: A Rational Approach Based on an Analytical Model and a Deterministic Global Optimization. IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control 54(7), 1293–1302 (2007). DOI 10.1109/TUFFC.2007.390. URL http://dx.doi.org/10.1109/TUFFC.2007.390

[37] R. Chelouah, P.S.: A hybrid method combining continuous tabu search and Nelder-Mead simplex algorithm for the global optimization of multiminima functions. European Journal of Operational Research 161, 636–654 (2005)

[38] Rodríguez, J., Renaud, J., Watson, L.: Trust Region Augmented Lagrangian Methods for Sequential Response Surface Approximation and Optimization. Journal of Mechanical Design 120(1), 58–66 (1998). DOI 10.1115/1.2826677. URL http://dx.doi.org/10.1115/1.2826677

[39] Rubinstein, R.Y.: Optimization of computer simulation models with rare events . European Journal of Operational Research 99, 89–112 (1997)

[40] Rubinstein, R.Y., Kroese, D.P.: The Cross-Entropy Method: A unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning. Springer: Berlin Heidelberg, New York, USA (2004)

[41] Stein, M.: Large sample properties of simulations using latin hypercube sampling. Technometrics 29(2), 143–151 (1987). URL http://www.jstor.org/stable/1269769

[42] Tao, J., Wang, N.: DNA Double Helix Based Hybrid GA for the Gasoline Blending Recipe Optimization Problem. Chemical Engineering and Technology 31(3), 440–451 (2008). DOI 10.1002/ceat.200700322. URL http://dx.doi.org/10.1002/ceat.200700322

[43] Wang, K., Wang, N.: A novel RNA genetic algorithm for parameter estimation of dynamic systems. Chemical Engineering Research and Design 88(11), 1485–1493 (2010). DOI 10.1016/j.cherd.2010.03.005. URL https://doi.org/10.1016/j.cherd.2010.03.005