**Les Cahiers du GERAD** ISSN: 0711–2440

# Combining surrogate strategies with MADS for mixed-variable derivative-free optimization

A.-S. Crélot, C. Beauthier, D. Orban, C. Sainvitu, A. Sartenaer

G–2017–70

August 2017



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

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

# Combining surrogate strategies with MADS for mixed-variable derivative-free optimization

**Anne-Sophie Crélot** [a]

**Charlotte Beauthier** [b]

**Dominique Orban** [c]

**Caroline Sainvitu** [b]

**Annick Sartenaer** [a]

[a] *Department of Mathematics and naXys, University of Namur, 5000 Namur, Belgium*

[b] *Cenaero, Eole Building, 6041 Gosselies, Belgium*

[c] *GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal (Québec) Canada, H3C 3A7*

anne-sophie.crelot@unamur.be
charlotte.beauthier@cenaero.be
dominique.orban@gerad.ca
caroline.sainvitu@cenaero.be
annick.sartenaer@unamur.be

**August 2017**

**Les Cahiers du GERAD**

**G–2017–70**

**Abstract:** We consider the solution of derivative-free optimization problems with continuous, integer, discrete and categorical variables in the context of costly black-box mixed-variable industrial problems. Our approach is based on NOMAD, an implementation of the mesh-adaptive direct-search method (MADS), supplemented with surrogate models and strategies in the local poll and global search steps. The surrogate models are radial basis function interpolations managed by the surrogate-assisted evolutionary software MINAMO developed at Cenaero. The proposed approach is validated on a collection of problems from the literature and we compare several surrogate-based strategies. In the general mixed-variable case, the results show that employing MINAMO as a surrogate-based strategy within NOMAD in the poll and search steps increases both robustness and efficiency when compared to MINAMO's surrogate-based evolutionary algorithm alone or to NOMAD. On problems with mixed-integer variables only, we also experiment with the specialized mixed-integer solver BONMIN instead of MINAMO's evolutionary algorithm in the search step. It turns out to be slightly more efficient and substantially more robust when high accuracy is required.

**Keywords:** Derivative-free optimization, mixed-variable optimization, radial basis functions surrogates, categorical variables, mesh adaptive direct-search algorithm, evolutionary algorithm

**Résumé :** Nous considérons les problèmes d'optimisation sans dérivées avec variables continues, entières, discrètes et de catégorie dans le contexte d'applications industrielles avec objectif de type boîte noire coûteuse. Notre approche se base sur NOMAD, une implémentation de la méthode de recherche directe adaptative (MADS), agrémentée de fonctions substitut dans la sonde locale et la recherche globale. Les fonctions substitut sont des modèles d'interpolation à base radiale gérées par le logiciel évolutionnaire MINAMO développé à Cenaero. Nous validons notre approche sur un jeu de problèmes provenant de la littérature et comparons plusieurs stratégies liées aux fonctions substitut. Dans le cas des variables mixtes, l'utilisation de MINAMO pour gérer les substituts dans les étapes de sonde et de recherche améliore la robustesse et l'efficacité par rapport à MINAMO seul ou NOMAD seul. Dans le cas des variables continues et entières seulement, nous utilisons le logiciel d'optimisation en nombres entiers BONMIN au lieu de MINAMO, qui s'avère être plus efficace et substantiellement plus robuste lorsqu'une précision élevée est requise.

**Mots clés :** Optimisation sans dérivées, optimisation en variables mixtes, substituts à base radiale, variables de catégorie, algorithme de méthode de recherche directe adaptative, algorithme évolutionnaire

**Acknowledgments:** The present research benefited from computational resources made available on the Tier-1 supercomputer of the Fédération Wallonie–Bruxelles, infrastructure funded by the Walloon Region under the grant agreement number 1117545.

# 1 Introduction

We consider the mixed-variable derivative-free optimization (DFO) problem

$$\min_x f(x) \quad \text{s.t.} \quad c(x) \leq 0, \qquad l \leq \bar{x} \leq u, \tag{1}$$

where $c(x) = (c_1(x), c_2(x), \ldots, c_v(x))^T$, $x = (\bar{x}, x^c)$ and $\bar{x} = (x^r, x^i, x^d)$ with $x^r \in \mathbb{R}^{n_r}$, $x^i \in \mathbb{Z}^{n_i}$, $x^d \in D = D_1 \times \cdots \times D_{n_d}$, $x^c \in C = C_1 \times \cdots \times C_{n_c}$, $n_r + n_i + n_d = \bar{n}$ and $\bar{n} + n_c = n$. The sets $D_k$ and $C_k$ are at most countable ordered and unordered, respectively. We call $x^r$ the *continuous* variables, $x^i$ the *integer* variables, $x^d$ the *discrete* variables and $x^c$ the *categorical* variables. The vectors $l$ and $u \in \mathbb{R}^{\bar{n}}$ represent lower and upper bounds on the ordered variables $\bar{x}$. The constraints on the ordered and categorical variables are nonrelaxable, i.e., they must always be satisfied. We denote by $\Omega$ the feasible set of (1). We consider the objective $f : \mathbb{R}^{n_r} \times \mathbb{Z}^{n_i} \times D \times C \to \mathbb{R}$ and constraints $c : \mathbb{R}^{n_r} \times \mathbb{Z}^{n_i} \times D \times C \to \mathbb{R}^v$ as black-boxes, i.e., typically costly functions to evaluate, that may be nonconvex, discontinuous or noisy, and whose derivatives may not exist or be available.

In an industrial context, because of modelling choices, only nonintrusive DFO methods are considered. For example at the applied research center Cenaero, problems of the form (1) arise in the design of turbomachinery outlet guide vanes (Baert et al., 2015), cryogenic tanks (Mahajan et al., 2015) or composite materials (Madhavan and Martiny, 2013). The variety of fields of application is an additional argument to keep methods as general as possible. Modeling problems in such contexts often requires integer, discrete and/or categorical variables, e.g. the number of vanes in a turbine, their outlet angles chosen from a prescribed finite set of possible angles, or an insulation material chosen from a given catalog.

As computing capacity increases, so do the cost, complexity and size of black-box simulation problems. As detailed in Section 2, few existing solvers are able to solve the general mixed-variable problem (1). At Cenaero, many such problems are currently tackled by way of an online surrogate-assisted evolutionary algorithm (EA) (Forrester and Keane, 2009) implemented as part of the MINAMO optimization suite (Sainvitu et al., 2009; Baert et al., 2015). Surrogates play several roles in making such costly problems tractable. They act as inexpensive models for the time-consuming objective and constraints and are used to guide the search towards improved iterates during local exploitation and global exploration. Our goal, in this paper, is to propose a robust and efficient method to solve problem (1), i.e., supported by convergence guarantees that do not interfere with efficiency considerations. To this aim, we consider NOMAD, an implementation of the Mesh-Adaptive Direct-Search (MADS) family of methods (Audet et al., 2009; Le Digabel, 2011) proved to be globally convergent (cf. Section 3.1), and supplement it with Radial Basis Function (RBF) interpolation models constructed by MINAMO. Those models are used as surrogates in the local poll and global search stages of NOMAD. In addition in the search step, those surrogates are exploited using a mixed evolutionary algorithm or a branch-and-bound approach, depending on the nature of the variables.

The paper is organized as follows. Section 2 shortly reviews the DFO solvers literature and Section 3 outlines the purpose of NOMAD and MINAMO together with the convergence properties of NOMAD. In Section 4, we first describe our surrogate-based strategy. We next illustrate each stage of its construction and its performance on a set of analytical problems. Section 5 reports numerical experiments on the real-world Heatshield problem (Kokkolaras et al., 2001; Abramson, 2004). We conclude and give some perspectives in Section 6.

# 2 Related software and algorithms

When derivatives are not available, finite-difference approximations are effective in certain applications but can be unreliable in the presence of noise, and unrealistic for expensive functions. DFO algorithms only use function values and differ in the way they exploit this information to determine the new iterate. The Nelder and Mead (1965) simplex-reflection method is one of the most commonly-implemented DFO algorithms. Conn et al. (2009) describe both directional and model-based methods but focus on problems with continuous variables and only briefly mention the adaptation of methods to mixed-integer problems. Kolda et al. (2003) give a detailed survey on direct-search methods.

In the DFO algorithms and software survey of Rios and Sahinidis (2013), only four out of the 22 solvers considered can handle noncontinuous variables. Of those four, two are part of the TOMLAB Matlab toolbox (Holmström, 1997)

and are designed to solve problems with continuous and integer variables. The other two are NOMAD, which is the only one that accepts categorical variables, and HOPSPACK (Platenga, 2009), a generating set search algorithm that solves problems with continuous and integer variables.

New solvers later emerged. RBFOpt (Costa and Nannicini, 2014) is a free solver for unconstrained problems similar to the `rbfsolve` method of TOMLAB. Newby and Ali (2015) present HEMBOQA as well as two modifications of it, that are adaptations of BOBYQA (Powell, 2009) to handle integer variables. Müller et al. (2013b) develop SO-MI, an algorithm based on RBF surrogates, to identify a global minimizer of problems with costly black-box functions in real and integer variables. Müller et al. (2013a) introduce SO-I, which is dedicated to problems with integer variables only. Finally MISO (Müller, 2015) defines an optimization framework that adapts several continuous surrogate-based algorithms to mixed-integer problems.

Evolutionary and genetic algorithms have also been adapted to mixed-integer problems. NSGA-II (Deb et al., 2002) implements a single- and multi-objective genetic algorithm for problems in continuous and binary variables. However it is not well suited to solve problems involving expensive numerical simulations.

Other methods to solve mixed-variable problems are studied, without leading to software release. Bajer and Holeňa (2010) present an approach based on RBF Networks and genetic algorithms for problems with continuous and discrete variables. Herrera et al. (2014) develop a model-assisted method using multiple kernel regression, also designed for mixed-variable problems. Liao et al. (2014) solve mixed-variable problems with an ant Colony Optimization. The first two methods are well suited for problems with expensive functions while the last one may require many function evaluations to solve a problem.

# 3 Background on Nomad and Minamo

This section gives an overview of the two solvers that are used in this work. Section 3.1 is dedicated to NOMAD. We then describe MINAMO and its online surrogate-based optimization process in Section 3.2. Advantages and weaknesses of both are highlighted in order to motivate the combination proposed in Section 4.

## 3.1 Nomad

NOMAD is a `C++` implementation (Abramson, Audet, Couture, Dennis Jr., Le Digabel, and Tribes, https://www.gerad.ca/nomad/) of the Mesh Adaptive Direct Search (MADS) algorithm for nonlinear optimization (Audet et al., 2009; Le Digabel, 2011). It is designed for simulation-based constrained optimization, with black-box functions. NOMAD also implements a MADS adapted to mixed variables (Abramson et al., 2009a) and can handle continuous, integer, binary and categorical variables.

The MADS algorithm is a local iterative optimization method that relies on a mesh. To minimize $f$ over the feasible domain $\Omega$, NOMAD uses a progressive barrier function denoted by $(f,h)$, where $h$ is the constraint violation function

$$h(x) := \begin{cases} \sum_{j=1}^{v} (\max(c_j(x),0))^2 & \text{if } x \in X, \\ +\infty & \text{otherwise,} \end{cases} \tag{2}$$

and where $X$ is the subset of points $x = (\bar{x},\, x^c) \in \mathbb{R}^{n_r} \times \mathbb{Z}^{n_i} \times D \times C$ satisfying the non-relaxable constraints $l \leq \bar{x} \leq u$ and $x^c \in C$.

At each iteration $k$, $f$ and $h$ are evaluated at trial points with the aim to decrease one or both of them. An iteration (and consequently a point) may be *dominating*, *improving* or *unsuccessful*. It is *dominating* if at least one trial point dominates the current best feasible or current best infeasible point. A point $y$ dominates a point $x$ if and only if either $h(y) < h(x)$ and $f(y) \leq f(x)$, or $h(y) \leq h(x)$ and $f(y) < f(x)$. If the iteration is not dominating and there is an infeasible trial point improving the constraint violation but with higher objective value than the current best infeasible point, the iteration is *improving*. Otherwise it is *unsuccessful*. During iteration $k$, the (optional) search step takes place first. If it produces a dominating point, this point becomes the new current best solution and the next iteration is started, without

going through the poll step. Otherwise the poll step is performed. If the latter fails to generate a dominating point and the problem contains categorical variables, an extended poll step is performed. The next iteration then takes place.

The iterates generated by the MADS algorithm must always remain on the current, iteration dependent, mesh defined at iteration $k$ as

$$M_k = \left( \bigcup_{(\bar{w},w^c)\in S_k} \{\bar{w} + \Delta_k^m z \mid z \in \mathbb{Z}^{\bar{n}}\} \right) \times C,$$

where $S_k$ denotes the set of points previously evaluated up to iteration $k$ and $\Delta_k^m \in \mathbb{R}^+$ is the mesh size parameter. If a dominating point is found at the end of an iteration, the mesh size can be increased, while it remains unchanged after an improving iteration and is decreased if the iteration is unsuccessful. The whole optimization process is represented in the flowchart of Figure 1. We outline the main goals of each step in the remainder of this section.

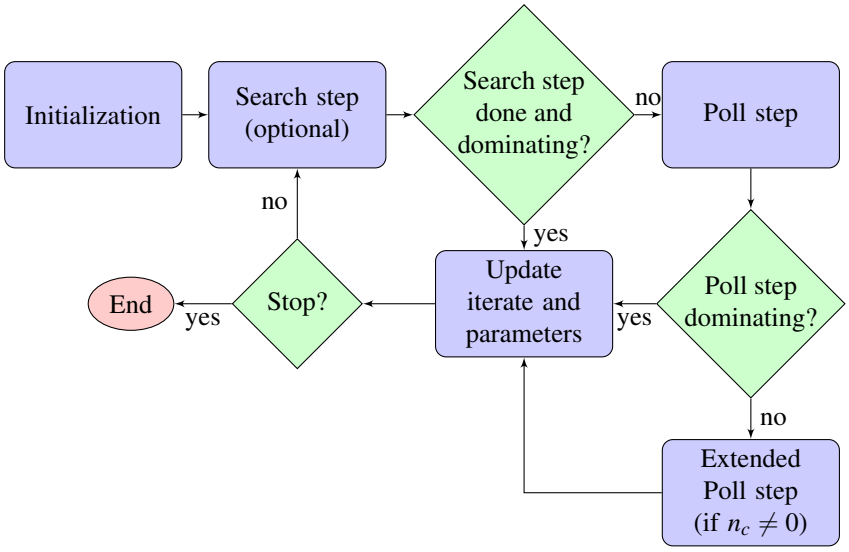


**Figure 1: Flowchart of Nomad.**

**Poll step** The poll step is the heart of the MADS philosophy to ensure convergence of the sequence of iterates, under appropriated assumptions, to a suitably defined stationary point. It generates a set of trial points around the current iterate, $x_k$, called the *frame*

$$F_k(x_k) = \{(\bar{x}_k + \Delta_k^m e, x_k^c) : e \in E_k\},$$

where $E_k$, the set of poll directions, is a positive spanning set satisfying technical conditions (Abramson et al., 2009a). The trial poll points must remain in a neighborhood of the current iterate (with respect to ordered variables $\bar{x}$) defined by the poll size parameter $\Delta_k^p \in \mathbb{R}^+$. The mesh and poll size parameters need to verify $\Delta_k^m \leq \Delta_k^p$ for all $k$ and $\lim_{k\in K} \Delta_k^m = 0$ if and only if $\lim_{k\in K} \Delta_k^p = 0$ for any infinite index set $K$.

Three different strategies are available in NOMAD for generating the poll directions $E_k$. The GPS option corresponds to coordinates directions, LTMADS (Audet and Dennis Jr, 2006) is a strategy using directions generated from a lower triangular matrix, and ORTHOMADS (Abramson et al., 2009b) uses orthogonal directions. For each strategy, the number of directions generated can be $n+1$ (minimal positive spanning set) or $2n$ (maximal positive spanning set). Figure 2 shows two examples of a MADS frame and mesh in $\mathbb{R}^2$, using ORTHOMADS directions.

Once the frame is generated, an *opportunistic* strategy is used by default to evaluate $f$ and $h$ at its points. This means that we stop scanning the frame as soon as a dominating point is found, if any. Prior to evaluate the true objective function and constraints, we consider in Section 4 the ordering of the poll points using surrogate values.

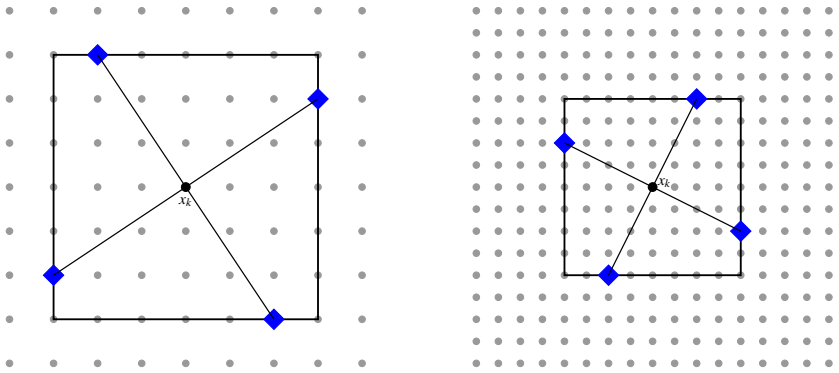


**Figure 2: Two examples of a MADS frame (blue diamonds) and mesh (gray circles) in $\mathbb{R}^2$ for the ordered variables. The square is the $\ell_\infty$-norm ball of radius $\Delta_k^p$ centered at the current iterate $x_k$. The spacing between mesh points is $\Delta_k^m$.**

In the presence of categorical variables, whenever the poll step does not find a dominating iterate, the extended poll step is performed. The latter consists in exploring new configurations for the categorical variables than the current $x_k^c$, based on a user-defined neighborhood structure (as there exists no natural order between the possible configurations). The neighborhood structure for categorical variables is modeled by way of a set-valued function $\mathcal{N} : \Omega \to 2^\Omega$, where $2^\Omega$ represents the set of all possible finite subsets of $\Omega$. Without loss of generality, we assume that $\mathcal{N}(x)$ is finite for all $x \in \Omega$. Among all neighbors $y = (\bar{y}, y^c) \in \mathcal{N}(x_k)$ with $x_k^c \neq y^c$, NOMAD selects the promising neighbors, i.e., those satisfying some technical conditions based on its barrier function value $(f,h)$ (Abramson et al., 2009a). From each of the selected points, a finite sequence of poll steps is performed. By default, each of these sequences starts with the initial mesh size $\Delta_0^m$, working on the ordered variables while keeping $y^c$ constant,[1] and stops when its own current mesh size reaches $\Delta_k^m$.

Note that when $\Delta_k^m$ becomes small, this could imply a large number of inner iterations for each of the selected points. Furthermore the descent is performed on surrogates when available. To avoid extra computation and evaluations of surrogates, we add a new stopping criterion: a maximal number of surrogate evaluations per extended-poll descent.

**Search step** The optional search step takes place before the poll step and allows to search for a dominating point lying on the current mesh using any strategy. Besides several independent strategies proposed by NOMAD, the users may code any strategy they wish, as long as it generates a finite number of points lying on the mesh. We explain our own strategies for that *user search* involving surrogates in Section 4. NOMAD's default search step is made of a *speculative search*, possibly followed by a *model search*. The former occurs when the previous iteration terminates with a successful (dominating or improving) poll point. The speculative search then further explores the most recently used successful direction. If the speculative search produces a new dominating point, a new iteration starts without performing a poll step. Otherwise a model search is performed in which a quadratic model of the exact function is built and minimized by NOMAD itself. Again if the model search produces a new dominating point, a new iteration starts. Otherwise, attention turns to the poll step.

**Handling discrete and categorical variables** At the beginning of this section, we pointed out that NOMAD is able to deal with continuous, integer, binary and categorical variables. Yet in the description of our problem (1), we also include discrete variables. We can adapt the representation of discrete variables in two ways: whether considering them as integer or as categorical variables. We choose the first option, that we implement using a mapping to match each discrete variable to a new integer one.

This choice is motivated by the way NOMAD deals with categorical variables. Indeed, during the poll step described above, only the non categorical variables are considered. The categorical variables are treated in the extended poll, based on information provided by the user for the categorical neighborhood. Choosing to consider the discrete variables as integer variables, we include their handling in the poll step instead of the extended poll step.

[1] The variable $y$ should be indexed by $k$ and with respect to the set of neighbors, but this is omitted here for clarity of notation.

**Convergence theory** The global convergence of a method refers to convergence to a local solution from any starting point. The global convergence properties of NOMAD rely on the poll step. Audet and Dennis Jr (2006) study the continuous case, while Abramson et al. (2009a, Theorem 10) generalize the results to mixed-variable problems, also considering the use of extended poll steps (Abramson et al., 2009a, Theorem 11). Those results rely on nonsmooth analysis concepts, including hypertangent vectors, several flavors of tangent cones, and generalized directional derivatives. The strongest results are obtained when $\Omega$ is a regular set, category that includes convex sets (Clarke, 1990, Proposition 2.4.4) and sets described by smooth functions. Clarke (1990, Theorem 2.4.7) discussed more general conditions for a regular feasible set for inequality constraints and Clarke et al. (1998, Chapter 2, Proposition 7.4) for equality constraints, for instance.

## 3.2 Minamo

The multi-disciplinary optimization software MINAMO performs online Surrogate-Based Optimization (SBO) and design space exploration. Though it was initially designed for continuous problems, MINAMO has been generalized to discrete and categorical variables (Baert et al., 2015). At each iteration of this process, MINAMO constructs surrogate models of the objective and constraint functions. An evolutionary algorithm (EA) is employed to identify an approximate global solution of the problem defined by those surrogates. MINAMO is appropriate for solving (1) directly. The main disadvantage of this approach is the lack of convergence properties of the evolutionary algorithm. In the remainder of this section, we detail the online SBO, the surrogates, the EA and other features of MINAMO. In Section 4, we exploit the MINAMO surrogates and SBO within NOMAD in order to accelerate the convergence and to retain global convergence properties.

**Online SBO** Surrogate-Based Optimization starts by generating a Design of Experiments (DoE). The DoE is a set of points covering the search space, where the objective and constraints are evaluated, and that forms an initial database. A first set of surrogates, $f_s$ for $f$ and $c_s$ for $c$, is built from the information in the database. The goal is to define surrogates that are cheap to evaluate yet capture features of the true objective and constraints.

As the iterations progress, MINAMO performs evaluations of the objective and constraints at new points, called candidates, that are entered into the database along with their corresponding function values. This in turn allows MINAMO to adaptively construct improved surrogate models.

MINAMO uses an EA to find an improved candidate $x^*$ as an approximate minimizer of a merit function (Booker et al., 1998) associated to

$$\min_x f_s(x) \quad \text{s.t. } c_s(x) \leq 0,\ l \leq \bar{x} \leq u, \tag{3}$$

that combines $f_s$, $c_s$, and additional terms meant to encourage exploration of the feasible space. The actual $f$ and $c$ are evaluated at $x^\dagger$ and optionally at other points computed during the optimization process. The new set of samples is then appended to the database. Updating the database during the process enables dynamic surrogates to represent $f$ and $c$ more accurately in the interesting zone as the algorithm proceeds. For this reason, SBO is said to be *online*. Once the database is updated, the next iteration is entered and the scheme is repeated until a stopping criterion is met (typically a maximal number of objective and constraint evaluations). Figure 3 represents a flowchart of online SBO. Note that the first pass in "update the database" means filling an empty database with information coming from the initial DoE.

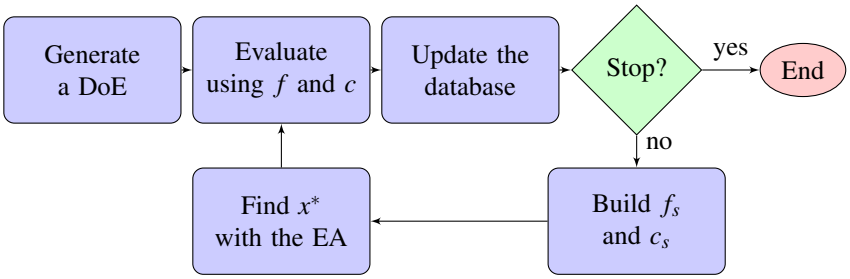


**Figure 3: Flowchart of online SBO.**

**Surrogates** MINAMO implements several types of surrogates, including RBF Networks and Kriging (Forrester et al., 2008; Conn et al., 2009). RBF are well suited to model highly multimodal functions, even when the size of the problem increases, and require fewer interpolation points than polynomial models.

An RBF interpolation model can be expressed as a linear combination of radial basis functions

$$RBF(x) = \sum_{i=1}^{s} \omega_i g_i(\|x - t_i\|, \sigma_i), \tag{4}$$

where $t_i$, $i = 1,\ldots,s$ are the interpolation points, $\omega_i$, $i = 1,\ldots,s$, are weights to be determined and $g_i$, $i = 1,\ldots,s$, are radial basis functions. The latter depend on the interpolation points $t_i$ and on parameters $\sigma_i > 0$. If there are only continuous and integer variables, a specific procedure, called *Move-Limit*, selects the interpolation points within a neighborhood of the current best iterate. The size of the neighborhood evolves with the iterations in a similar way as in the trust-region method (Conn et al., 2000). In the presence of discrete and/or categorical variables, all the points from the database are used as interpolation points.

The radial basis functions considered by MINAMO are of Gaussian type, $g(\theta,\sigma) = \exp(-\frac{1}{2}\theta^2/\sigma^2)$, or of multiquadric type, $g(\theta,\sigma) = (1+\theta^2/\sigma^2)^{\frac{1}{2}}$. MINAMO also proposes an automated *TunedRBF* that combines Gaussian and multiquadric basis functions. In such a TunedRBF model, $g_i$ and $\sigma_i$ are selected by MINAMO in order to determine accurate surrogates based on a correlation coefficient computed by a Leave-One-Out (LOO) procedure. More precisely, MINAMO implements an *Efficient LOO* suggested by Rippa (1999). Contrary to Kriging models, RBF surrogates' parameters are less expensive to tune. The weights $\omega_i$ are determined by solving a linear system $G\omega = Y$ that expresses the interpolation conditions.

The choice of the norm in (4) completes the definition of the RBF model. In the absence of categorical variables, the Euclidean norm is used. Otherwise, a specific heterogeneous distance is defined based on the work of McCane and Albert (2008) and Wilson and Martinez (1997).

**Evolutionary algorithm** SBO uses an EA (Goldberg, 1989; Affenzeller et al., 2009) to solve (3). This choice is motivated by several arguments: EAs are zero-th order methods, often able to identify global solution(s), robust and applicable to noisy, nonlinear, discontinuous, black-box functions. In addition, EAs can handle continuous, integer, discrete and categorical variables.

The EA implemented in MINAMO follows the general steps of a genetic algorithm: variation and selection of individuals. Variation is done through the application of cross-over and mutation operators while selection consists in the tournament method. In addition, elitism is also applied to a few individuals. Internally, individuals are represented by real numbers. The EA in MINAMO handles constrained problems through the constraint tournament selection (Deb, 2000), which is based on the comparison between two individuals. If both are feasible, the individual with a better objective function value is selected. If only one of the two is feasible, the feasible individual is retained. If neither is feasible, the one with the smallest constraint violation is selected.

# 4 Combining Nomad and Minamo

Surrogate models can be used at several stages during a NOMAD iteration. During the poll step, neighbors on the frame can be ordered according to surrogate objective and constraint values in hope of reducing the number of actual objective and constraint evaluations before a success. In addition, surrogates can help find an improved iterate as an approximate solution of a surrogate problem in the search step. By using an EA to minimize a surrogate, we encourage global exploration of the search space. In practice, we must often operate within a given budget, e.g., a fixed number of function evaluations or simulations. Employing surrogates at both the local and global levels possibly increases the chances of finding a better solution within the budget allowed.

In what follows, we first describe our testing environment. We next present and compare the numerical results.

## 4.1 Testing environment

Our experiments revolve around two sets of test problems. Set I contains problems without any categorical variable, while Set II contains problems with at least one categorical variable. The distinction is motivated by the fact that the heterogeneous distance used in the presence of categorical variables (mentioned on page 9) makes the RBF surrogates nondifferentiable. The use of methods that rely on derivatives of the surrogates will thus be limited to some problems of Set I, i.e. problems including only continuous and integer variables (this is imposed by the use of a mixed-integer solver). We employ such a method on mixed-integer problems, as detailed below.

**Test problems** Set I consists in the 17 problems listed in Table 1 and Set II consists in the 20 problems with categorical variables listed in Table 2. In the tables, problems of type A are classic continuous optimization test problems to which we impose that some variables be integer, discrete, or categorical (if in Set II). Problems of type B represent physical problems with mixed variables.

For each problem, Tables 1 and 2 report the number of continuous, integer, discrete and categorical variables as $n_r$, $n_i$, $n_d$ and $n_c$, respectively. The column $n$ gives the total number of variables and the column $v$ indicates the number of constraints. The source of each problem is indicated in the last column.

For problems involving categorical variables, the neighborhood structure of a point $x$ is given by $\mathcal{N}(x)$ as follows. A neighbor of $x = (\bar{x}, x^c)$ is a point of the form $(\bar{x}, y^c)$ where $y^c$ agrees with $x^c$ on all components but one, of index $i$ say, which is chosen as an element of $C_i \setminus \{x_i^c\}$. Note that, by convention, one assumes that fixed variables are eliminated from the problem. In order to allow a wide exploration of the search space, all possible neighbors of $x$ appear in $\mathcal{N}(x)$.

**Table 1: Test problems without categorical variables (Set I).**

| Type | Name | $n_r$ | $n_i$ | $n_d$ | $n_c$ | $n$ | $v$ | Reference |
|---|---|---|---|---|---|---|---|---|
| A | BarnesCase1 | 1 | 0 | 1 | 0 | 2 | 3 | (Robinson et al., 2006) |
| A | BarnesCase2 | 1 | 1 | 0 | 0 | 2 | 3 | (Robinson et al., 2006) |
| B | CarSideImpact | 9 | 0 | 2 | 0 | 11 | 10 | (Gandomi et al., 2011) |
| A | G07Case3 | 2 | 2 | 6 | 0 | 10 | 8 | (Regis, 2014) |
| A | G09 | 4 | 3 | 0 | 0 | 7 | 4 | (Regis, 2014) |
| A | MysteryCase1 | 1 | 0 | 1 | 0 | 2 | 0 | (Sasena, 2002) |
| A | MysteryCase2 | 1 | 1 | 0 | 0 | 2 | 0 | (Sasena, 2002) |
| A | MysteryCase6 | 1 | 0 | 1 | 0 | 2 | 0 | (Sasena, 2002) |
| B | PressureVessel | 2 | 2 | 0 | 0 | 4 | 3 | (Cagnina et al., 2008) |
| A | RastriginCase1 | 1 | 0 | 1 | 0 | 2 | 0 | (Yang, 2010) |
| A | RastriginCase2 | 1 | 1 | 0 | 0 | 2 | 0 | (Yang, 2010) |
| B | ReinforcedConcreteBeam | 1 | 1 | 1 | 0 | 3 | 2 | (Gandomi et al., 2011) |
| A | RosenbrockCase1 | 1 | 0 | 1 | 0 | 2 | 0 | (Yang, 2010) |
| A | RosenbrockCase2 | 1 | 1 | 0 | 0 | 2 | 0 | (Yang, 2010) |
| B | SpeedReducer | 6 | 1 | 0 | 0 | 7 | 11 | (Cagnina et al., 2008) |
| B | Spring | 2 | 1 | 0 | 0 | 3 | 4 | (Huang and Arora, 1997) |
| B | SteppedCantileverBeam | 4 | 2 | 4 | 0 | 10 | 11 | (Gandomi et al., 2011) |

**Running numerical experiments** The budget of black-box function and constraint evaluations is divided into two parts: one for building the first DoE and the other to perform the optimization. The initial DoE is built using $3n$ evaluations, while the number of evaluations available for the optimization phase is fixed at 100 for each problem.

In order to have meaningful results and to be able to make comparisons, we perform multiple runs for each method on each problem. Indeed, the optimization methods under consideration are sensitive to starting points (i.e., here, the DoEs) and the EA includes randomness. We thus generate 50 different DoEs for each problem, on which we run each method once.

The results are presented using the performance and data profiles (Moré and Wild, 2009) based on the number of function evaluations needed to find $x$ satisfying

$$f(x_0) - f(x) \geq (1 - \tau)(f(x_0) - f_{best}), \tag{5}$$

where $x_0$ denotes the initial point, $f_{best}$ is the best objective value across all methods for a given problem, and $\tau \in (0, 1)$ is a fixed tolerance. Smaller values of $\tau$ require a final objective value closer to $f_{best}$. As our test sets include constrained

**Table 2: Test problems with categorical variables (Set II).**

| Type | Name | $n_r$ | $n_i$ | $n_d$ | $n_c$ | $n$ | $v$ | Reference |
|---|---|---|---|---|---|---|---|---|
| A | BarnesCase3 | 1 | 0 | 0 | 1 | 2 | 3 | (Robinson et al., 2006) |
| B | CarSideImpactDC | 9 | 0 | 0 | 2 | 11 | 10 | (Gandomi et al., 2011) |
| A | G07Case4 | 2 | 2 | 0 | 6 | 10 | 8 | (Regis, 2014) |
| A | G09IC | 4 | 0 | 0 | 3 | 7 | 4 | (Regis, 2014) |
| A | MysteryCase3 | 1 | 0 | 0 | 1 | 2 | 0 | (Sasena, 2002) |
| B | PressureVesselIC | 2 | 0 | 0 | 2 | 4 | 3 | (Cagnina et al., 2008) |
| A | RastriginCase3 | 1 | 0 | 0 | 1 | 2 | 0 | (Yang, 2010) |
| A | RastriginCase12 | 2 | 2 | 3 | 3 | 10 | 0 | (Yang, 2010) |
| A | RastriginCase22 | 8 | 2 | 4 | 6 | 20 | 0 | (Yang, 2010) |
| B | ReinforcedConcreteBeamDC | 1 | 1 | 0 | 1 | 3 | 2 | (Gandomi et al., 2011) |
| B | ReinforcedConcreteBeamIC | 1 | 0 | 1 | 1 | 3 | 2 | (Gandomi et al., 2011) |
| B | ReinforcedConcreteBeamIDC | 1 | 0 | 0 | 2 | 3 | 2 | (Gandomi et al., 2011) |
| A | RosenbrockCase3 | 1 | 0 | 0 | 1 | 2 | 0 | (Yang, 2010) |
| A | RosenbrockCase12 | 2 | 2 | 3 | 3 | 10 | 0 | (Yang, 2010) |
| A | RosenbrockCase22 | 8 | 2 | 4 | 6 | 20 | 0 | (Yang, 2010) |
| B | SpeedReducerIC | 6 | 0 | 0 | 1 | 7 | 11 | (Cagnina et al., 2008) |
| B | SpringIC | 2 | 0 | 0 | 1 | 3 | 4 | (Huang and Arora, 1997) |
| B | SteppedCantileverBeamDC | 4 | 2 | 0 | 4 | 10 | 11 | (Gandomi et al., 2011) |
| B | SteppedCantileverBeamIC | 4 | 0 | 4 | 2 | 10 | 11 | (Gandomi et al., 2011) |
| B | SteppedCantileverBeamIDC | 4 | 0 | 0 | 6 | 10 | 11 | (Gandomi et al., 2011) |

problems and multiple runs to deal with randomness, we need to adapt the measures used to produce the profiles. On constrained problems, we replace $f(x_0)$ by $f_{feas}$ in (5), where $f_{feas}$ is defined as the objective function value at the first feasible iterate found during the optimization process. Note that such a feasible iterate exists for all our test problems. Otherwise no feasible solution is found and the run is viewed as a failure. One way of handling multiple runs for each problem is to produce profiles based on the mean or median of the results. However such data aggregation results in a loss of information and we prefer to consider each run as a different instance of the problem. For example, if 50 runs are performed on each of the 20 test problems, a profile is produced as if there were 1000 instances. Moreover each instance of a given problem has its own $f_{feas}$, as the initial DoE may differ, but only one $f_{best}$ is defined per problem, based on results across all methods and all instances of the problem.

## 4.2 Numerical results without using surrogates derivatives

In what follows, we compare four methods on the two sets of test problems. None of these methods rely on the possible availability of surrogates derivatives. The three first methods are variants of NOMAD, whose characteristics are described below. The fourth one is the default version of MINAMO SBO, as described in Section 3.2. We add this method, labeled MINAMO in the following, for comparison purposes.

**Nomad as reference method** In the default version of NOMAD, the search step differs slightly whether we solve problems of Set I or II. For problems of Set I, two types of search are activated: the speculative search and the model search. For problems involving categorical variables (Set II), the model search is disabled. The poll step is based on the generation of $2n$ directions using the ORTHOMADS method, along with the opportunistic strategy. As for the extended poll step, it uses the neighborhood structure previously described. Finally, all algorithmic parameters are set to their default value. We only impose a maximal number of 100 black-box evaluations, which typically serves as a stopping criterion. In what follows, we use version 3.8.0 of NOMAD and we refer to this method as NOMAD.

**Nomad+ as using surrogates in the poll step** At the beginning of the poll step, NOMAD+ calls MINAMO to build TunedRBF surrogates $f_s$ and $c_s$ based on all information stored in the database. Those surrogates are combined into a surrogate barrier function $(f_s, h_s)$. The surrogate constraint violation $h_s$ is defined by replacing $c_j$ with $c_{s_j}$ in (2). We evaluate $(f_s, h_s)$ at all trial poll points and order them in increasing values of this surrogate barrier function. The actual barrier function $(f, h)$ is then evaluated at each point in order. The opportunistic strategy stops the process as soon as an improvement is obtained in the actual barrier function. The search step strategy is unchanged.

**Nomad+/Minamo as using surrogates both in the poll and search steps** In order to measure the impact of the user search, the default search strategies i.e., speculative and model search are deactivated. The search step in NOMAD+/MINAMO is exclusively made of a user search in which (3) is based on TunedRBF surrogates and solved using the EA implemented in MINAMO. Indeed, in the presence of categorical variables, TunedRBF surrogates are not differentiable. Otherwise, a poll step takes place under the form described for NOMAD+ above, using updated surrogates taking account of the point generated by the search step. In this case, up to $2n$ black-box evaluations (at worst) can be spent in the poll step.

In addition, we propose variants of NOMAD+/MINAMO denoted NOMAD+/MINAMO S$k$. In NOMAD+/MINAMO the user search corresponds to a single iteration of MINAMO SBO. In the S$k$ variants, we allow $k$ iterations (meaning $k$ black-box evaluations) of MINAMO SBO. These variants give more freedom to MINAMO with respect to NOMAD. Note that NOMAD+/MINAMO S1 is identical to NOMAD+/MINAMO.

We summarize the features of each method in Table 3, where "I" and "I / II" indicate a feature activated for problems of Set I only, and of both sets, respectively. Note that experiments showed that adding speculative search in NOMAD+/MINAMO does not have a significant impact on accuracy or robustness.

**Table 3: Summary of the features of the four methods for problems of Sets I and II.**

| | Poll step | | Search step | | |
|---|---|---|---|---|---|
| Method | Default | Surrogate | Speculative | Quadratic | User |
| NOMAD | I / II | | I / II | I | |
| NOMAD+ | | I / II | I / II | I | |
| NOMAD+/MINAMO S$k$ | | I / II | | | I / II |
| MINAMO | | | | | |

**Performance profiles** Figure 4 shows the performance profiles comparing NOMAD, NOMAD+, NOMAD+/MINAMO, MINAMO and NOMAD+/MINAMO S10 for $\tau = 10^{-3}$, $\tau = 10^{-5}$ and $\tau = 10^{-7}$.

The use of a surrogate in the poll step slightly improves the efficiency and robustness, except for low values of $\tau$ in Set I. Building and evaluating $f_s$ and $c_s$ does not have any significant impact on the duration of the optimization. Building surrogates only needs to be done when the database is updated. The training time depends on the size of the problem and on the number of interpolating points, but always remains of the order of a second for problems up to 32 variables in our experiments using a 2.5GHz computer.

In all cases, NOMAD+/MINAMO (i.e. using surrogates in the search step) is substantially more efficient and more robust than both NOMAD and NOMAD+ and MINAMO in the absence of categorical variables. Surrogates have more impact when used in the search step than in the poll step. However it is interesting to keep them in both steps as their construction and evaluation time is small.

The S10 version of NOMAD+/MINAMO always helps improve efficiency. While it is also more robust on problems of Set II, the S1 version surpasses the S10 variant in terms of robustness on problems from Set I.

With $\tau = 10^{-3}$, NOMAD+/MINAMO is less efficient but substantially more robust than MINAMO. As $\tau$ decreases, the efficiency gap disappears and NOMAD+/MINAMO finds the lowest objective value on almost 30% of the problems of Set I, against 20% for MINAMO when $\tau = 10^{-7}$. Even when $\tau = 10^{-3}$, NOMAD+/MINAMO S10 remains the most efficient approach. In addition to being supported by convergence guarantees, NOMAD+/MINAMO S$k$ identifies lower objective values than MINAMO within the same evaluation budget. Choosing a smaller or larger value for the number $k$ of black-box evaluations per user search is a way of selecting a more robust or a more efficient method respectively in the case of Set I (problems without categorical variables). We recommend a larger value of $k$ for problems with categorical variables (set II).

Further analysis of the results shows that complementing NOMAD with surrogates increases the probability of finding a feasible solution within the allowed number of evaluations. The decrease of the objective value with respect to the number of evaluations is often faster for MINAMO than for NOMAD+/MINAMO. However the latter tends to find lower objective values.

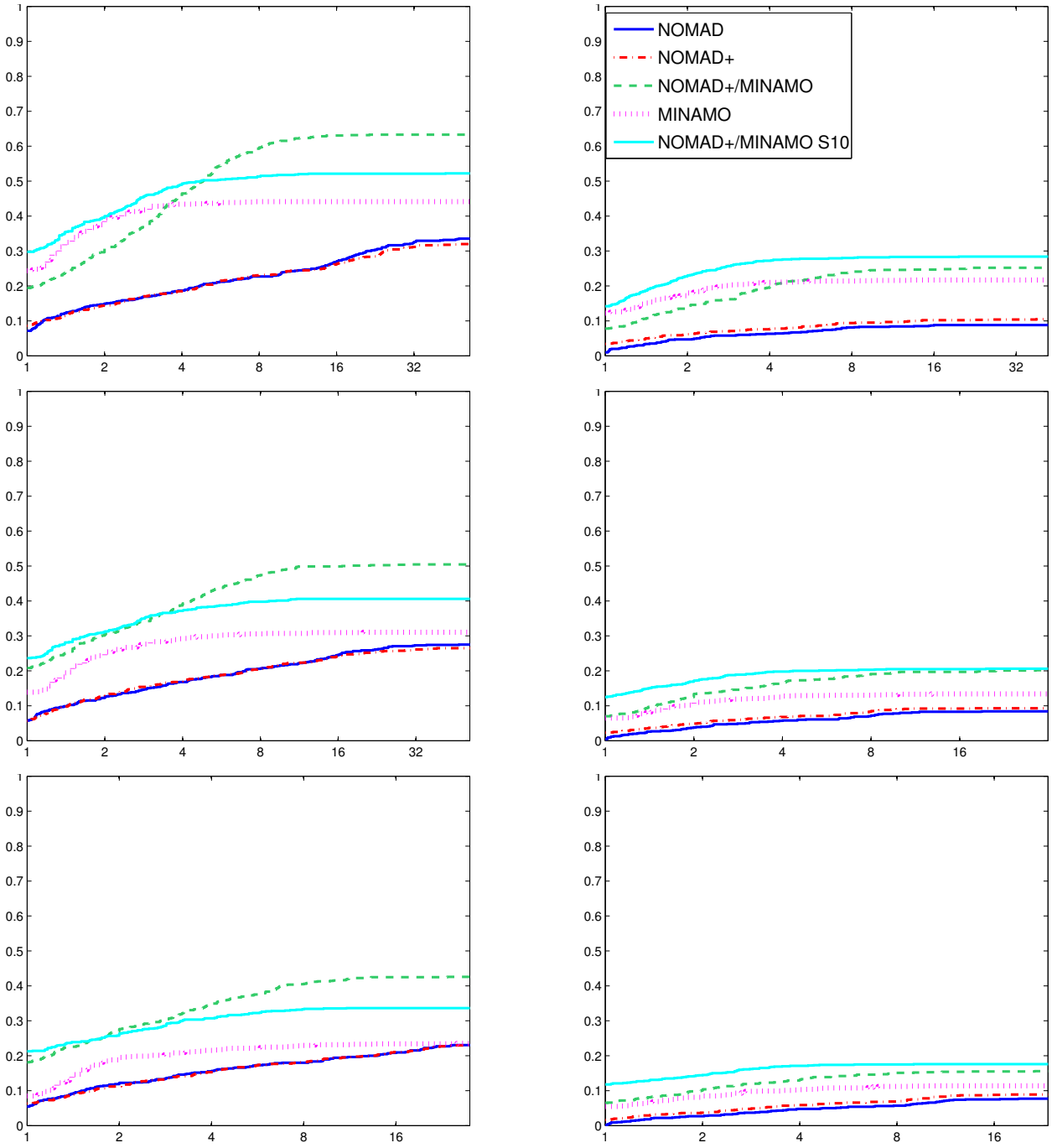


**Figure 4: Performance profiles for test sets I (left) and II (right). From top to bottom, the rows correspond to $\tau = 10^{-3}$, $\tau = 10^{-5}$ and $\tau = 10^{-7}$.**

When looking at profiles considering constrained and unconstrained problems separately, we observe a difference in the methods behaviour. When working on Set I (see Figure 5), all the curves involving NOMAD are higher for unconstrained problems. This means more efficient and more robust methods. For all accuracy levels $\tau$, NOMAD+/MINAMO is more efficient and robust on unconstrained problems, while on constrained problems MINAMO and NOMAD+/MINAMO *S*10 remains the most efficient. Similar observations apply to Set II, see Figure 6, but NOMAD+/MINAMO *S*10 surpasses NOMAD+/MINAMO in terms of efficiency for unconstrained problems in this case.

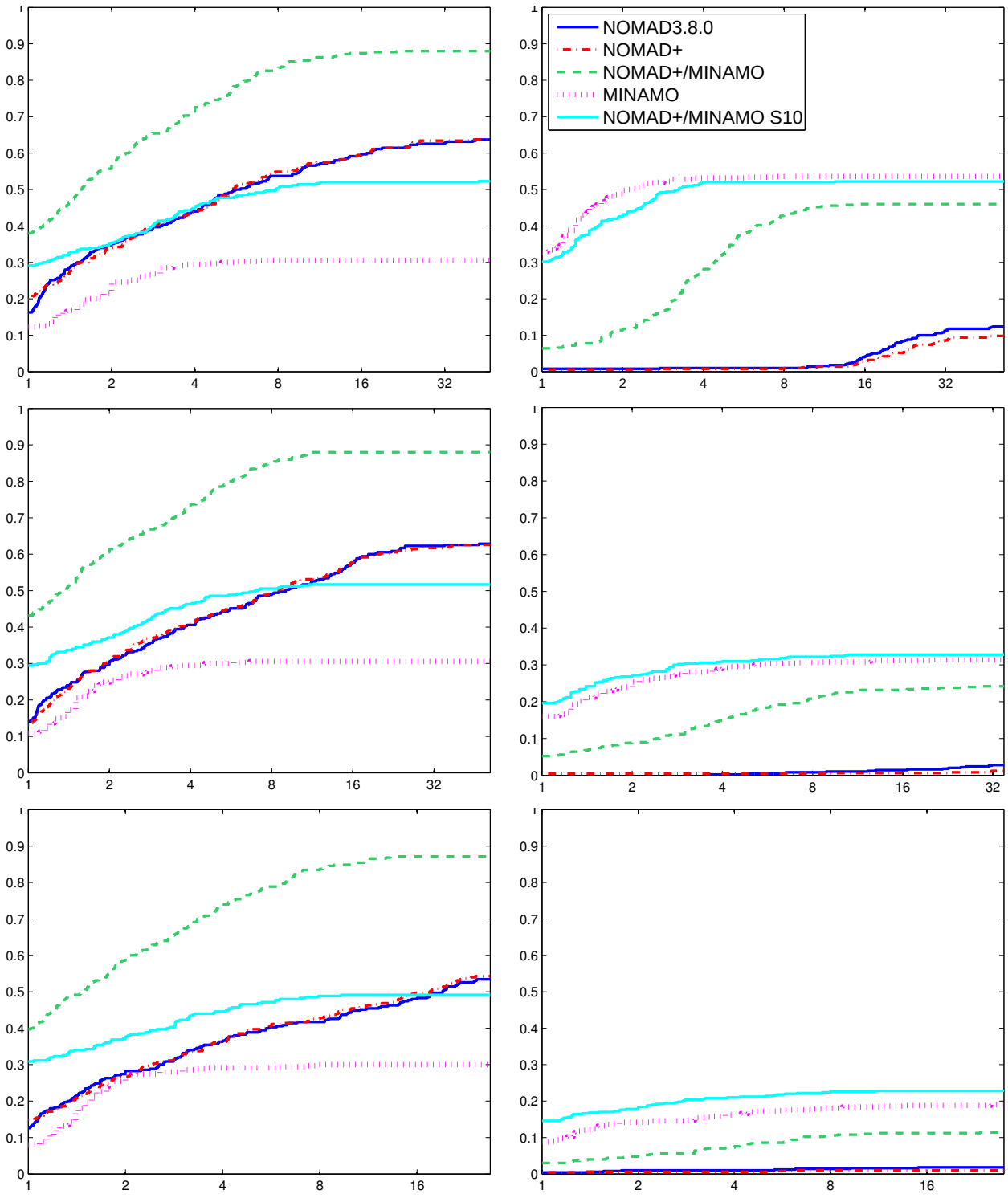


**Figure 5: Performance profiles for test sets I, unconstrained (left) and constrained (right) problems. From top to bottom, the rows correspond to $\tau = 10^{-3}$, $\tau = 10^{-5}$ and $\tau = 10^{-7}$.**

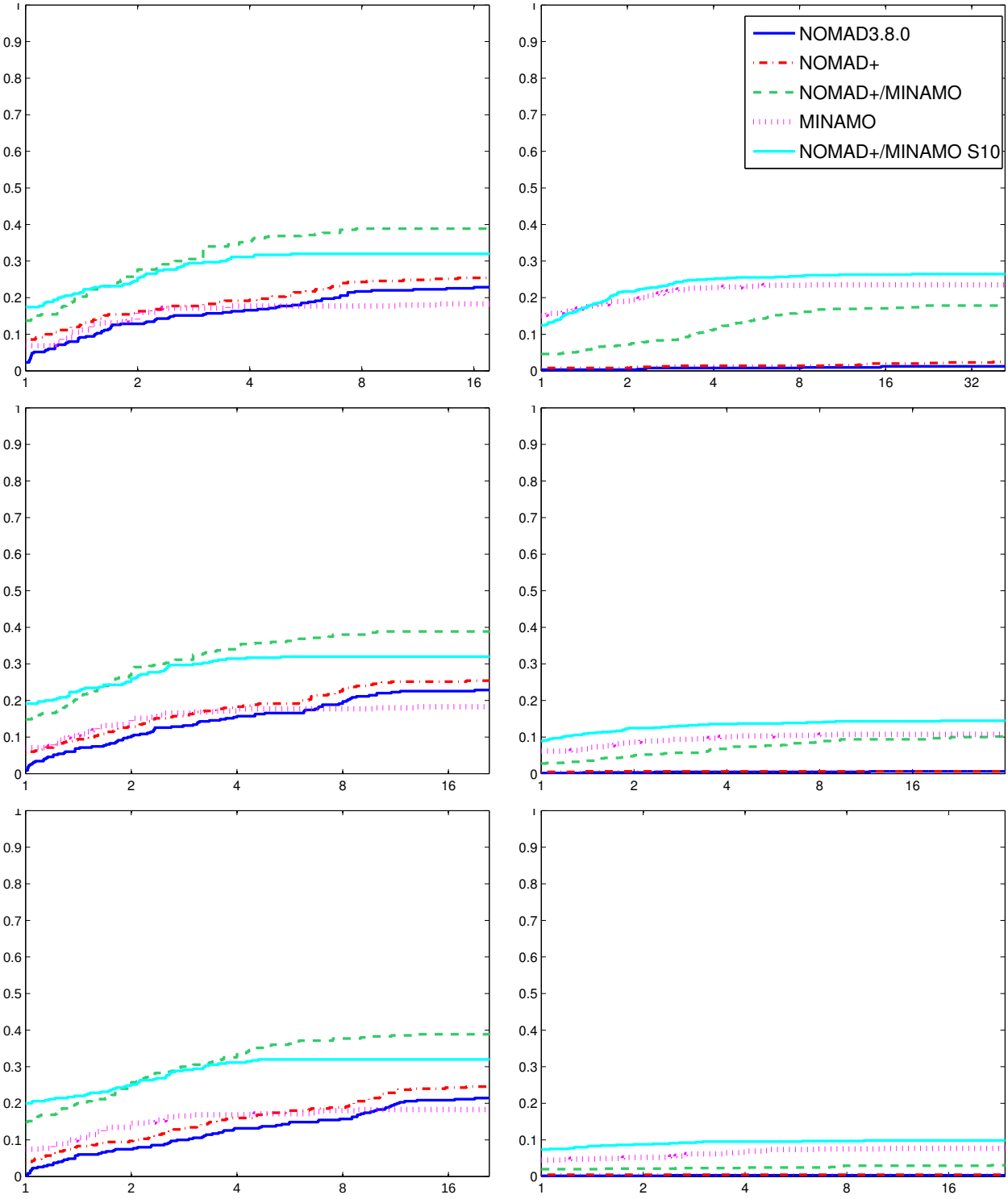


**Figure 6: Performance profiles for test sets II, unconstrained (left) and constrained (right) problems. From top to bottom, the rows correspond to $\tau = 10^{-3}$, $\tau = 10^{-5}$ and $\tau = 10^{-7}$.**

### 4.3 Numerical results exploiting surrogates derivatives

On problems without categorical variables (Set I), the TunedRBF surrogates computed by MINAMO are differentiable and we have access to their derivatives. In this section, we consider, as a sixth method, another variant of NOMAD in which an approximate solution of (3) is identified using the derivative-based mixed-integer solver BONMIN (Bonami et al., 2008).[2] We refer to this approach as NOMAD+/BONMIN and compare its performance with the five methods outlined in Section 4.2.

As BONMIN only handles continuous and integer variables, we need the application of NOMAD+/BONMIN to the eight problems of Table 1 for which $n_d = 0$. In this case, (3) becomes a mixed-integer nonlinear program (MINLP), which we solve with BONMIN with default parameters for nonconvex MINLP problems (relaxations are solved from three different starting points and softer branching rules are applied) (Bonami and Lee, 2013). Note that though discrete variables in the collections of problems we consider are treated as integers via a mapping (see Section 3.1) the approach used in BONMIN would be meaningless as it solves relaxations (with respect to integer constraints).

Figure 7 shows performance profiles on the eight problems of Table 1 for which $n_d = 0$ with NOMAD+/BONMIN added to the comparison, and with $\tau = 10^{-3}$ and $\tau = 10^{-5}$. When $\tau = 10^{-3}$ the efficiency of NOMAD+/MINAMO is similar to that of NOMAD+/BONMIN but NOMAD+/MINAMO is slightly more robust. When $\tau = 10^{-5}$ NOMAD+/BONMIN is both more efficient and more robust than NOMAD+/MINAMO. At that tolerance level, NOMAD+/BONMIN has approximately the same efficiency as MINAMO, but is substantially more robust. Overall, NOMAD+/BONMIN tends to behave similarly to NOMAD+/MINAMO but is able to identify solutions with a lower objective value. When $\tau = 10^{-7}$ the profile is similar to the profile for $\tau = 10^{-5}$.

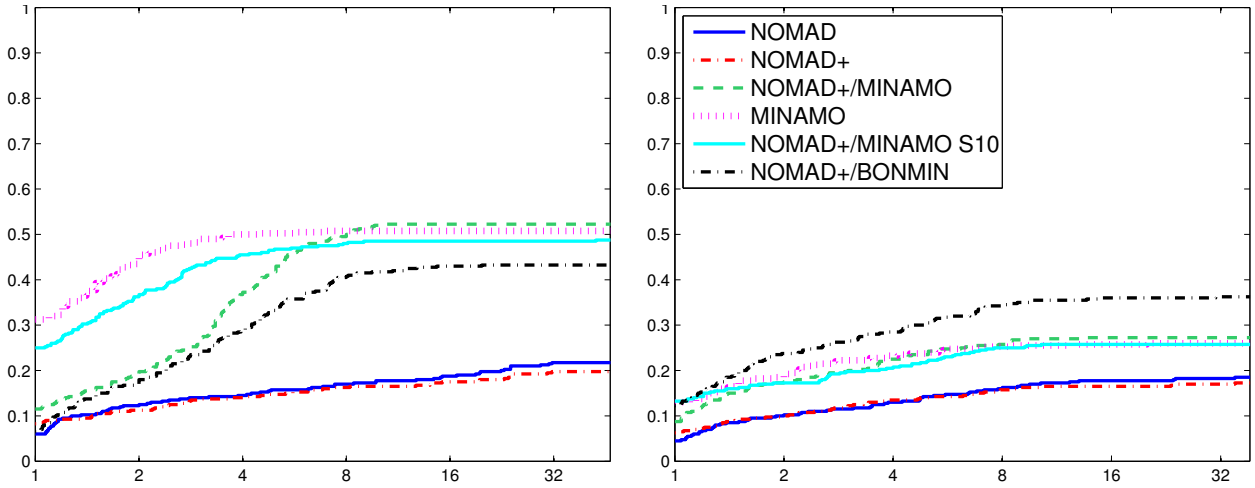


**Figure 7: Performance profiles on problems with continuous and integer variables for $\tau = 10^{-3}$ (left) and $\tau = 10^{-5}$ (right).**

## 5 Numerical experiments on the Heatshield problem

The Heatshield problem models a thermal insulation system. A number of heat intercepts are placed between a cold and a hot surface to be insulated from each other so that their temperature can be kept constant. The problem involves continuous, integer and categorical variables. The problem has been studied since the late 1970s (Hilal and Boom, 1977) but Kokkolaras et al. (2001) include categorical variables in the specification of the problem for the first time. Abramson (2004) solves the problem with categorical variables and additional nonlinear constraints. We first present the problem and related notation. Next we explain how we modify it with respect to previous studies and present numerical results.

[2] BONMIN is an open-source C++ solver available at https://www.coin-or.org/Bonmin/

## 5.1 Problem description

Four groups of variables describe the system. The number of intercepts $\ell$ is a positive integer. The cold and hot surfaces correspond to intercepts 0 and $\ell+1$, respectively. The temperature at intercept $i$ is $T_i$, $i = 0,\ldots,\ell+1$, with $T_0 = T_{\text{cold}}$ and $T_{\ell+1} = T_{\text{hot}} \geq T_{\text{cold}}$. The temperatures should be nondecreasing when going from the cold to the hot surface, i.e.

$$T_i \leq T_{i+1}, \quad 0 \leq i \leq \ell. \tag{6}$$

The insulators used between intercepts form a vector of categorical variables $I = [I_1, I_2, \ldots, I_{\ell+1}]$ whose $i$-th component corresponds to the insulator type between intercepts $i-1$ and $i$, $1 \leq i \leq \ell+1$. Insulator $i$ has a thickness denoted by $W_i \in \mathbb{R}^+$, $1 \leq i \leq \ell+1$. Thicknesses also have to verify

$$\sum_{i=1}^{\ell} W_i \leq L, \tag{7}$$

where $L > 0$ is the fixed distance between the cold and hot surfaces. The sizes of the vectors $T$, $I$ and $W$ depend on the variable $\ell$, i.e. the problem has one integer variable, $\ell+1$ categorical variables and $2\ell$ continuous variables.

The objective function represents the total refrigeration power required by the system. It depends on the heat flow from an intercept to the next given by Fourier's law. Kokkolaras et al. (2001) state the objective as

$$\sum_{i=1}^{\ell} TCE_i \left(\frac{T_{\text{hot}}}{T_i} - 1\right) \left[\frac{A_{i+1}}{W_{i+1}} \int_{T_i}^{T_{i+1}} \kappa(T, I_{i+1})\,\mathrm{d}T - \frac{A_i}{W_i} \int_{T_{i-1}}^{T_i} \kappa(T, I_i)\,\mathrm{d}T\right], \tag{8}$$

where $TCE_i$ is the thermodynamic cycle efficiency coefficient at intercept $i$, $A_i$ is the cross-sectional area of insulator $i$ and $\kappa$ is the thermal conductivity. Kokkolaras et al. (2001) minimize (8) subject to the bound constraints $W_i \geq 0$ for $1 \leq i \leq \ell$ and constraints (6) and (7).

Abramson (2004) starts from the same problem and adds three new nonlinear constraints to it. The problem then corresponds to a load-bearing system. The nonlinear constraints related to the load $\Gamma$ and stress of the system are the inequalities

$$\frac{\Gamma}{A_i} \leq \bar{\mathrm{o}}_i = \min\{\mathrm{o}_i(T, I_i) : T_{i-1} \leq T_i \leq T_{i+1}\}, \quad 1 \leq i \leq \ell+1, \tag{9}$$

where $\mathrm{o}_i(T, I_i)$ is the tensile yield strength at insulator $i$. Following Abramson (2004), (9) hold as equalities at optimality and can be used to substitute $A_i = \frac{\Gamma}{\bar{\mathrm{o}}_i}$ in the problem. A second nonlinear constraint added by Abramson (2004) imposes that the total mass of the system cannot exceed the maximal allowable mass $\mu_{\max}$. Its expression is

$$\sum_{i=1}^{\ell} \rho_i(I_i) \frac{W_i}{\bar{\mathrm{o}}_i} \leq \frac{\mu_{\max}}{\Gamma}, \tag{10}$$

where $\rho_i(I_i)$ is the density of the material used as insulator $i$. Finally the constraint associated with the system's thermal expansion or contraction can be expressed as

$$\sum_{i=1}^{\ell} \left[\frac{\int_{T_{i-1}}^{T_i} \phi(T, I_i)\kappa(T, I_i)\,\mathrm{d}T}{\int_{T_{i-1}}^{T_i} \kappa(T, I_i)\,\mathrm{d}T}\right] \left(\frac{W_i}{L}\right) \leq \frac{\delta}{100}, \tag{11}$$

where $\phi(T, I_i)$ is the unit thermal contraction from intercept $i$ to any point between intercepts $i$ and $i-1$ and $\delta$ is a limit on the total contraction of the system expressed as a percentage. This optimization problem, denoted Heatshield, is made of the objective function (8) where $A_i$ is replaced by $F/\bar{\mathrm{o}}_i$, constraints (6)–(7), (10)–(11) and the bound constraints.

## 5.2 Problem formulation

In this section, we test the methods of Table 3 on the Heatshield problem. We use the same insulator types and problem parameters as Kokkolaras et al. (2001) and Abramson (2004).

Because our surrogates and EA only apply to optimization problems of constant size, we fix the number of intercepts to $\ell = 10$, based on the choice discussed by Kokkolaras et al. (2001), and solve the problem with respect to variables $T$, $I$ and $W$. With $\ell$ fixed, we must adjust our definition of neighborhood structure. Kokkolaras et al. (2001) and Abramson (2004) consider three classes of neighbors: adding an intercept, removing an intercept and modifying an insulator type. We only retain the third strategy, where the neighbors of a choice of material are all other possible material choices.

Preliminary numerical experiments on Heatshield showed that the linear constraints (6) and (7) are well handled by NOMAD but cannot be satisfied when optimizing with the EA. The initial DoE and all points generated by the EA were infeasible with respect to (6) and (7). In such a case, the simulation code returns a constant value that denotes infeasibility, regardless of the constraint violation, and this results in meaningless surrogates. The failure is due to the very low probability of finding a point satisfying (6) and (7) in the space defined by the bound constraints during the random-based exploration of EA and DoE methods.

To overcome this difficulty we slightly modify the model in order to only generate points satisfying (6) and (7). These 2 constraints are removed from the problem formulation, resulting in an optimization problems with only 2 nonlinear constraints. In this new model the value $\tilde{W}_i$ is bounded between 0 and 100 and represents a percentage. The actual thickness $W_i$ of insulator $i$, $1 \leq i \leq \ell$, can be determined by multiplying $\tilde{W}_i$ by the remaining free space between intercept $i-1$ and the hot surface, i.e., $L - \sum_{k=1}^{i-1} W_k$. We apply the same reasoning to temperature variables: $\tilde{T}_i$ denotes a percentage, $1 \leq i \leq \ell$. The current gap between $T_{i-1}$ and $T_{\text{hot}}$, multiplied by $\tilde{T}_i$ is added to $T_{i-1}$ to obtain $T_i$, implying that (6) is always satisfied.

The advantage of the updated formulation is highlighted by running NOMAD (the only method that does not involve surrogates and/or EA) to solve both formulations. We compare the solution obtained after 200 function evaluations for both modeling approaches. The number of variables of Heatshield ($n = 32$) is slightly higher than that of the problems used in the previous section ($n \leq 20$). To keep a small budget of function evaluations while still adapting it to Heatshield size, we fix the maximal number of function evaluations to 200.

Figure 8 reports comparative results of NOMAD for 50 instances of Heatshield. We compute $y$, the logarithm of the ratio of the final objective value on the original model to that on the updated model. Positive vertical bars indicate a lower final objective value in the updated model, meaning that the updated formulation provides better results. A lower objective value is found on 50% of the instances using the updated model. Therefore we favor working with the updated formulation in the following.

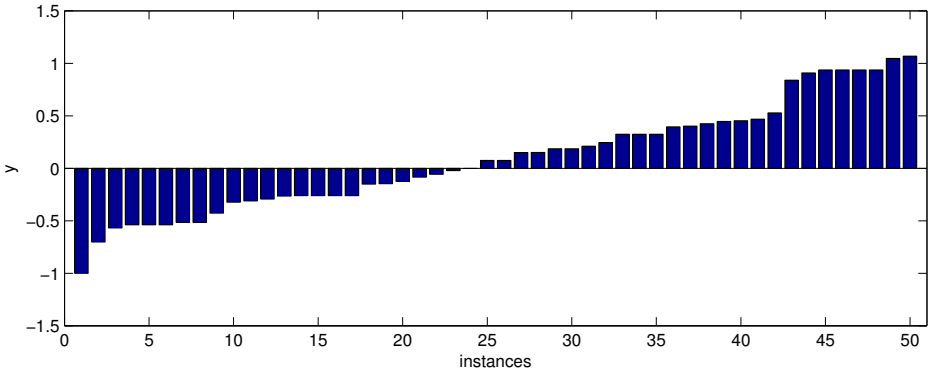


**Figure 8: Ratios of final objective values using Nomad 3.8.0 on Heatshield.**

We also have observed a large difference between the order of the objective function and those of the constraints. To overcome this issue, we choose to optimize the logarithm of the objective function (8).

When we generate random points within the bounds and satisfying (6) and (7), most of them are highly infeasible with respect to the thermal expansion constraint (11). On the contrary, the points generated during the optimization process have a value close to zero for (11), i.e., feasible or close to feasibility. Such points with a very high value for constraint (11) are unwanted and we choose to flag them as failures for surrogate building.

## 5.3 Numerical results

We now compare NOMAD, NOMAD+, NOMAD+/MINAMO, MINAMO and the S10, S20 and S50 versions of NOMAD+/MINAMO on the updated Heatshield model. We build 50 DoEs comprised of 320 points each, i.e., 10 times the number of variables. We then perform the optimization with a budget of 600 function evaluations. For each method we plot the current best feasible objective value as a function of the number of objective evaluations. The left-hand subplot corresponds to a mean on the 50 runs, the middle subplot to the best solution and the right-hand subplot to the worst solution.

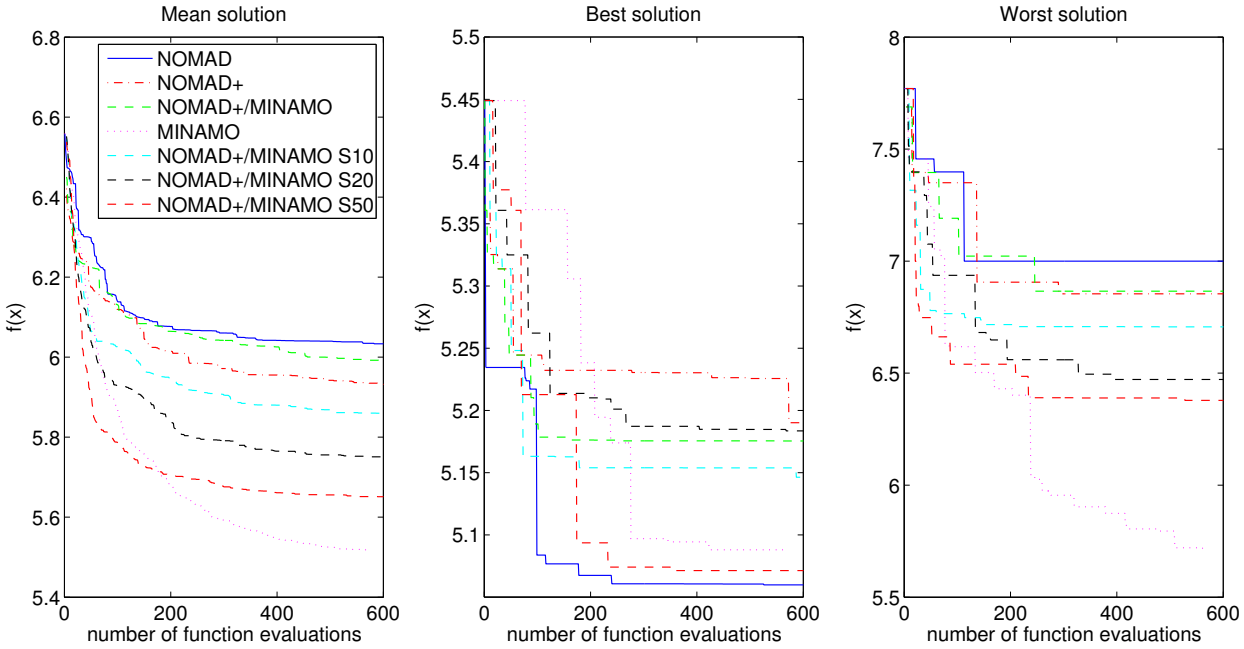


**Figure 9: Mean, best and worst function value with respect to the number of function evaluations on Heatshield.**

Heatshield is a constrained problem. As stated in Section 4.2, MINAMO tends to find a better solution for such problems. On average, we can observe on Figure 9 that MINAMO's curve is the lowest one. However, MINAMO is the slowest approach on heatshield, taking 75 hours of CPU time on average while 3 hours are needed by NOMAD+ and 6 hours for NOMAD+/MINAMO.

Figure 9 also shows the different levels of improvement brought by the surrogates in NOMAD. Using surrogates in the poll step (NOMAD+) implies a faster decrease of the objective function with respect to NOMAD. NOMAD+/MINAMO is only slightly better than NOMAD. Detailed results show that MINAMO needs a few iterations to find an improved point. A single MINAMO iteration is permitted in the NOMAD+/MINAMO search step, and it is unsuccessful most of the time. This limits the efficiency of the S1 approach. On the contrary, the S10, S20 and S50 approaches show increasing improvement as more and more iterations are allowed during the user search. In the early iterations, S10 and S20 behave in a similair way to MINAMO. NOMAD+/MINAMO S50 even decreases faster than MINAMO.

This corroborates the importance of supplementing NOMAD with surrogates, both in the poll and search steps. On the other hand MINAMO still provides solutions with lower objective. This leaves room for exploring new surrogate strategies, taking into account efficiency and speed of the process.

We now compare our best solutions to those of Abramson (2004), which he obtained by a mixed-variable generalized pattern-search algorithm. In Table 4, we compare its solution to the best solution obtained with NOMAD 3.8.0, for the same problem modelling and parameters. NOMAD 3.8.0 can even find a better solution. Note that only the normalized power value is given in the litterature. For comparison purpose, we compute the unnormalized power for

the litterature solution and the normalized value for our solution. This computation involves variables, meaning that a lower power does not imply a lower normalized power.

**Table 4: Best solutions from Abramson (2004) and from Nomad with our modelling, without any limit in term of function evaluations. Materials are abbreviated by the following: A = aluminum, E = epoxy (normal).**

| | NOMAD from Abramson (2004) | | | NOMAD 3.8.0 | | |
|---|---|---|---|---|---|---|
| Intercept | $I$ | $W$ | $T$ | $I$ | $W$ | $T$ |
| 1 | E | 0.625 | 4.25 | A | 0.0263 | 4.2 |
| 2 | E | 8.125 | 7.7375 | E | 8.3221 | 7.6592 |
| 3 | E | 7.9688 | 12.369 | E | 8.9516 | 12.6251 |
| 4 | E | 7.8125 | 18.094 | E | 3.2780 | 14.9481 |
| 5 | E | 12.344 | 29.912 | E | 17.0828 | 29.9322 |
| 6 | E | 26.094 | 71.094 | E | 23.9056 | 71 |
| 7 | E | 8.125 | 105.94 | A | 0.1059 | 71 |
| 8 | E | 5.3125 | 135.47 | E | 10.5150 | 115.5042 |
| 9 | E | 5 | 165.94 | E | 7.1791 | 156.3061 |
| 10 | E | 5.625 | 202.03 | E | 8.4931 | 207.4460 |
| 11 | E | | | E | | |
| Power | 106.35509 | | | 101.80249 | | |
| Norm. Power | 23.768 | | | 9.2759 | | |

As we work with a fixed number of intercepts (i.e., $\ell$ is fixed), we have to start from a different point than Kokkolaras et al. (2001) and Abramson (2004). Fixing $\ell$ also prohibits escaping local minima by exploring solutions with fewer or more intercepts. This, along with the poll directions defined in a different way and a very limited number of function evaluations (600), leads to different final solutions in terms of materials, thicknesses and temperatures. The way we reformulate the problem has an impact on the final solution as well. We observe in Table 5 larger thicknesses and temperatures in the first components of the solution. Remember that the power values given in Table 5 are expressed in logarithm.

**Table 5: Best solutions from Minamo, Nomad and Nomad+/Minamo S50, allowing 600 function evaluations. Materials are abbreviated by the following: N = nylon, E = epoxy (normal), Ep = epoxy (plane), T = teflon, S = steel and C = carbon steel.**

| | MINAMO | | | NOMAD | | | NOMAD+/MINAMO S50 | | |
|---|---|---|---|---|---|---|---|---|---|
| Intercept | $I$ | $W$ | $T$ | $I$ | $W$ | $T$ | $I$ | $W$ | $T$ |
| 1 | E | 46.11 | 6.31 | E | 32.52 | 6.15 | E | 24.03 | 5.31 |
| 2 | E | 31.30 | 78.82 | E | 42.79 | 92.05 | E | 39.50 | 67.02 |
| 3 | E | 11.99 | 169.96 | E | 17.01 | 212.72 | E | 15.55 | 209.76 |
| 4 | E | 7.47 | 299.78 | E | 4.79 | 295.29 | E | 7.94 | 227.42 |
| 5 | N | 1.40 | 299.92 | E | 1.72 | 299.69 | E | 8.60 | 270.89 |
| 6 | N | 0.45 | 299.98 | E | 0.24 | 299.76 | E | 1.91 | 294.60 |
| 7 | S | 0.74 | 299.99 | E | 0.40 | 299.92 | Ep | 2.09 | 298.72 |
| 8 | Ep | 0.48 | 299.99 | E | 0.43 | 299.97 | E | 0.18 | 299.10 |
| 9 | C | 0.02 | 299.99 | E | 0.05 | 299.99 | Ep | 0.15 | 299.64 |
| 10 | Ep | 0.02 | 299.99 | N | 0.00 | 299.99 | N | 0.04 | 299.79 |
| 11 | S | | | T | | | N | | |
| Power | 5.088088 | | | 5.049059 | | | 5.071348 | | |
| Norm. Power | 7.55792 | | | 6.61567 | | | 7.42807 | | |

Using our simulation code, we find the same normalized objective values for the full model solution. In terms of normalized power, best solutions are produced by NOMAD, NOMAD+/MINAMO S50 and MINAMO, all of them are better than Abramson (2004) solution. However the unnormalized power defined by (8) at the solution is higher because of the variables involved in the normalization computation.

# 6 Conclusion

In the context of mixed-variable optimization problems with expensive black-box function evaluations and no access to derivatives, few solvers are available. We propose new approaches to meet the demand of industry in solving such challenging problems.

We develop a method based on NOMAD supplemented by RBF surrogate models minimized by way of the EA implemented as part of MINAMO. Our method is supported by convergence guarantees and improves the efficiency and robustness of NOMAD and MINAMO alone. This is a framework combining NOMAD and MINAMO, and we could consider using other surrogates and EA than MINAMO.

Using surrogates in the poll and/or search steps makes our approach flexible. For mixed-integer problems we also propose an extra strategy using BONMIN to minimize the surrogate in the search step. Numerical experiments with a very limited number of expensive black-box evaluations show that all variants are efficient and improve the solution obtained with NOMAD alone. We are also competitive with respect to MINAMO on analytical problems as we provide comparable solutions in terms of accuracy while being more robust. Experiments on the mixed-variable problem Heatshield show the importance of problem formulation and encourage us to study new surrogates strategies. In particular, we could improve the approach to be able to solve problems of variable size effectively.

In order to further improve our approach, we plan to investigate the building and exploitation of surrogates, especially in the presence of categorical variables. Surrogates such as Gaussian Process Models (Qian et al., 2008) can be designed efficiently to represent functions involving mixed variables.

# A Test problems description

## A.1 Barnes

Minimize

$$\begin{aligned} f(x) &= 75.196 + 3.81x_1 - 0.126x_1^2 + 2.5056*10^{-3}x_1^3 - 1.034*10^{-5}x_1^4 6.83x_2 - 0.0302x_1x_2 \\ &\quad +1.281*10^{-3}x_2x_1^2 - 3.525*10^{-5}x_2x_1^3 + 2.266*10^{-}7x_2x_1^4 - 0.256x_2^2 3.46*10^{-}3x_2^3 \\ &\quad -1.35*10^{-}5x_2^4 + \frac{28.106}{x_2+1} + 5.237*10^{-}6x_1^2x_2^2 + 6.3*10^{-}8x_1^3x_2^2 + 1.663*10^{-}6x_1x_2^3 \\ &\quad +2.867e^{0.0005x_1x_2} \end{aligned}$$

Subject to

$$\begin{aligned} g_1(x) &= -\left(\frac{x_1x_2}{700} - 1\right) \leq 0 \\ g_2(x) &= -\left(\frac{x_2}{5} - \frac{x_1^2}{625}\right) \leq 0 \\ g_3(x) &= -\left(\frac{x_2}{50} - 1\right)^2 - \left(\frac{x_1}{500} - 0.11\right) \leq 0 \end{aligned}$$

Bound constraints and variables types for

Case1 $x_1 \in \{3,9,26,49,60,78\},\ 0.0 \leq x_2 \leq 60.0,\ x_1$ discrete, $x_2 \in \mathbb{R}$
Case2 $0 \leq x_1 \leq 80,\ 0.0 \leq x_2 \leq 60.0,\ x_1 \in \mathbb{N},\ x_2 \in \mathbb{R}$
Case3 $x_1 \in \{0,10,20,30,40,50,60,70,80\},\ 0.0 \leq x_2 \leq 60.0,\ x_1$ categorical, $x_2 \in \mathbb{R}$

## A.2 CarSideImpact

Minimize

$$f(x) = 1.98 + 4.90x_1 + 6.67x_2 + 6.98x_3 + 4.01x_4 + 1.78x_5 + 2.73x_7$$

Subject to

$$\begin{aligned} g_1(x) &= 1.16 - 0.3717x_2x_4 - 0.00931x_2x_{10} - 0.484x_3x_9 + 0.01343x_6x_{10} - 1 \leq 0 \\ g_2(x) &= 0.261 - 0.0159x_1x_2 - 0.188x_1x_8 - 0.019x_2x_7 + 0.0144x_3x_5 + 0.0008757x_5x_{10} \\ &\quad +0.08045x_6x_9 + 0.00139x_8x_{11} + 0.00001575x_{10}x_{11} - 0.32 \leq 0 \end{aligned}$$

$$
\begin{aligned}
g_3(x) &= 0.214 + 0.00817x_5 - 0.131x_1x_8 - 0.0704x_1x_9 + 0.03099x_2x_6 - 0.018x_2x_7 + 0.0208x_3x_8 \\
&\quad +0.121x_3x_9 - 0.00364x_5x_6 + 0.0007715x_5x_{10} - 0.0005354x_6x_{10} + 0.00121x_8x_{11} - 0.32 \leq 0 \\
g_4(x) &= 0.74 - 0.061x_2 - 0.163x_3x_8 + 0.001232x_3x_{10} - 0.166x_7x_9 + 0.227x_2^2 - 0.32 \leq 0 \\
g_5(x) &= 28.98 + 3.818x_3 - 4.2x_1x_2 + 0.0207x_5x_{10} + 6.63x_6x_9 - 7.7x_7x_8 + 0.32x_9x_{10} - 32 \leq 0 \\
g_6(x) &= 33.86 + 2.95x_3 + 0.1792x_{10} - 5.057x_1x_2 - 11.0x_2x_8 - 0.0215x_5x_{10} - 9.98x_7x_8 + 22.0x_8x_9 - 32 \leq 0 \\
g_7(x) &= 46.36 - 9.9x_2 - 12.9x_1x_8 + 0.1107x_3x_{10} - 32 \leq 0 \\
g_8(x) &= 4.72 - 0.5x_4 - 0.19x_2x_3 - 0.0122x_4x_{10} + 0.009325x_6x_{10} + 0.000191x_{11}^2 - 4 \leq 0 \\
g_9(x) &= 10.58 - 0.674x_1x_2 - 1.95x_2x_8 + 0.02054x_3x_{10} - 0.0198x_4x_{10} + 0.028x_6x_{10} - 9.9 \leq 0 \\
g_{10}(x) &= 16.45 - 0.489x_3x_7 - 0.843x_5x_6 + 0.0432x_9x_{10} - 0.0556x_9x_{11} - 0.000786x_{11}^2 - 15.7 \leq 0
\end{aligned}
$$

Bound constraints and variables types for

Default $0.5 \leq x_1, x_3, x_4 \leq 1.5,\ 0.45 \leq x_2 \leq 1.35,\ 0.875 \leq x_5 \leq 2.625,\ 0.4 \leq x_6, x_7 \leq 1.2,$
$x_8, x_9 \in \{0.192, 0.345\}$ (discrete), $0.5 \leq x_{10}, x_{11} \leq 1.5$

DC $0.5 \leq x_1, x_3, x_4 \leq 1.5,\ 0.45 \leq x_2 \leq 1.35,\ 0.875 \leq x_5 \leq 2.625,\ 0.4 \leq x_6, x_7 \leq 1.2,$
$x_8, x_9 \in \{0.192, 0.345\}$ (categorical), $0.5 \leq x_{10}, x_{11} \leq 1.5$

## A.3 G07

Minimize

$$
\begin{aligned}
f(x) &= x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 \\
&\quad +5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45
\end{aligned}
$$

Subject to

$$
\begin{aligned}
g_1(x) &= -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0 \\
g_2(x) &= 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0 \\
g_3(x) &= -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0 \\
g_4(x) &= 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0 \\
g_5(x) &= 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0 \\
g_6(x) &= x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leq 0 \\
g_7(x) &= 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leq 0 \\
g_8(x) &= -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0
\end{aligned}
$$

Bound constraints and variables types for

Case3 $x_i \in \{-10, -5, 0, 1.3, 2.2, 5, 8.2, 8.7, 9.5, 10\}\ \forall i = 1:6,$
$-10.0 \leq x_7$ and $x_8 \leq 10.0,\ -10 \leq x_9$ and $x_{10} \leq 10,$
$x_1$ to $x_6$ discrete, $x_7, x_8 \in \mathbb{R},\ x_9, x_{10} \in \mathbb{N}$

Case4 $x_i \in \{-10, -5, 0, 1.3, 2.2, 5, 8.2, 8.7, 9.5, 10\}\ \forall i = 1:6,$
$-10.0 \leq x_7$ and $x_8 \leq 10.0,\ -10 \leq x_9$ and $x_{10} \leq 10,$
$x_1$ to $x_6$ categorical, $x_7, x_8 \in \mathbb{R},\ x_9, x_{10} \in \mathbb{N}$

## A.4 G09

Minimize

$$
f(x) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7
$$

Subject to

$$
g_1(x) = 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 - 127 \leq 0
$$

$$\begin{aligned} g_2(x) &= 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 - 282 \leq 0 \\ g_3(x) &= 23x_1 + x_2^2 + 6x_6^2 - 8x_7 - 196 \leq 0 \\ g_4(x) &= 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0 \end{aligned}$$

Bound constraints and variables type for

Default $-10 \leq x_i \leq 10\ \forall i = 1:7$ and $x_i \in \mathbb{N}\ \forall i = 1:3$ and $x_i \in \mathbb{R}\ \forall i = 4:7$.
IC $-10 \leq x_i \leq 10\ \forall i = 1:7$,
$x_i \in \mathbb{N}$ treated as categorical variable $\forall i = 1:3$, $x_i \in \mathbb{R}, \forall i = 4:7$.

## A.5 Mystery

Minimize

$$f(x) = 2 + 0.1(x_2 - x_1^2)^2 + (1 - x_1)^2 + 2(2 - x_2)^2 + 7\sin(0.5x_1)\sin(0.7x_1x_2)$$

Bound constraints and variables type for

Case1 $x_1 \in \{-0.5, 0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5\},\ -0.5 \leq x_2 \leq 5.0,\ x_1$ discrete, $x_2 \in \mathbb{R}$
Case2 $0 \leq x_1 \leq 5,\ -0.5 \leq x_2 \leq 5.0,\ x_1 \in \mathbb{N},\ x_2 \in \mathbb{R}$
Case3 $x_1 \in \{1, 2, 3\},\ -0.5 \leq x_2 \leq 5.0,\ x_1$ categorical, $x_2 \in \mathbb{R}$
Case6 $x_1 \in \{1, 2, 3\},\ -0.5 \leq x_2 \leq 5.0,\ x_1$ discrete, $x_2 \in \mathbb{R}$

## A.6 PressureVessel

Minimize

$$f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$$

Subject to

$$\begin{aligned} g_1(x) &= -x_1 + 0.0193x_3 \leq 0 \\ g_2(x) &= -x_2 + 0.00954x_3 \leq 0 \\ g_3(x) &= -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0 \end{aligned}$$

Bound constraints and variables types for

Default $x_1 = 0.0625n_1,\ x_2 = 0.0625n_2$ where $n_1$ and $n_2 \in \mathbb{N},\ x_3$ and $x_4 \in \mathbb{R}$
and where $1 \leq n_1 \leq 99, 1 \leq n_2 \leq 99, 10.0 \leq x_3 \leq 200.0,\ 10.00 \leq x_4 \leq 200.0$
IC $x_1 = 0.0625n_1,\ x_2 = 0.0625n_2$ where $n_1$ and $n_2 \in \mathbb{N}$ treated as categorical variables, $x_3$
and $x_4 \in \mathbb{R}$ and where $1 \leq n_1 \leq 99, 1 \leq n_2 \leq 99, 10.0 \leq x_3 \leq 200.0,\ 10.00 \leq x_4 \leq 200.0$

## A.7 Rastrigin

Minimize

$$f(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$$

Bound constraints and variables types for

Case1 $x_1 \in \{-5, -3, -1, 0, 1, 3, 5\},\ -5.0 \leq x_2 \leq 5.0,\ x_1$ discrete, $x_2 \in \mathbb{R}$
Case2 $-5 \leq x_1 \leq 5,\ -5.0 \leq x_2 \leq 5.0,\ x_1 \in \mathbb{N},\ x_2 \in \mathbb{R}$
Case3 $x_1 \in \{-5, -3, -1, 0, 1, 3, 5\},\ -5.0 \leq x_2 \leq 5.0,\ x_1$ categorical, $x_2 \in \mathbb{R}$
Case12 $x_1, x_2 \in \{-5, -3, -1, 0, 1, 3, 5\},\ x_3 \in \{-5, 0, 2, 5\},\ x_4, x_5 \in \{-5, -3, -1, 0, 1, 3, 5\}$,
$x_6 \in \{0, 1, 2, 3\},\ -5.0 \leq x_7, x_8 \leq 5.0,\ -5 \leq x_9, x_{10} \leq 5$,
$x_1$ to $x_3$ discrete, $x_4$ to $x_6$ categorical, $x_7, x_8 \in \mathbb{R},\ x_9, x_{10} \in \mathbb{N}$
Case22 $x_i \in \{-5, -3, -1, 0, 1, 3, 5\}\ \forall i = 1:8,\ x_9,\ x_{10} \in \{0, 1, 2, 3\},\ -5 \leq x_i \leq 5\ \forall i = 11:20$,
$x_1$ to $x_4$ discrete, $x_5$ to $x_{10}$ categorical, $x_{11}, x_{12} \in \mathbb{N},\ x_{13}$ to $x_{20} \in \mathbb{R}$

**Table 6: Possible values for variable $A$ in ReinforcedConcreteBeam problem.**

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0.2 | 0.8 | 1.4 | 2 | 2.64 | 3.41 | 4.03 | 4.84 | 6.16 | 8 | 11.06 |
| 0.31 | 0.88 | 1.55 | 2.17 | 2.79 | 3.52 | 4.2 | 5 | 6.32 | 8.4 | 11.85 |
| 0.4 | 0.93 | 1.58 | 2.2 | 2.8 | 3.6 | 4.34 | 5.28 | 6.6 | 8.69 | 12 |
| 0.44 | 1 | 1.6 | 2.37 | 3 | 3.72 | 4.4 | 5.4 | 7.11 | 9 | 13 |
| 0.6 | 1.2 | 1.76 | 2.4 | 2.08 | 3.95 | 4.65 | 5.53 | 7.2 | 9.48 | 14 |
| 0.62 | 1.24 | 1.8 | 2.48 | 2.1 | 3.96 | 4.74 | 5.72 | 7.8 | 10.27 | 15 |
| 0.79 | 1.32 | 1.86 | 2.6 | 3.16 | 4 | 4.8 | 6 | 7.9 | 11 | |

## A.8 ReinforcedConcreteBeam

Minimize

$$f(A,b,h) = 29.4A + 0.6bh$$

Subject to

$$\begin{aligned} g_1(A,b,h) &= h - 4b \leq 0 \\ g_2(A,b,h) &= 180b + 7.375A^2 - Abh \leq 0 \end{aligned}$$

Bound constraints and variables types for

| | |
|---|---|
| Default | $A$ chosen among discrete values from Table 6,<br>$28 \leq b \leq 40,\ 5.0 \leq h \leq 10.0$, $b \in \mathbb{N}$ and $h \in \mathbb{R}$ |
| DC | $A$ chosen among categorical values from Table 6,<br>$28 \leq b \leq 40,\ 5.0 \leq h \leq 10.0$, $b \in \mathbb{N}$ and $h \in \mathbb{R}$ |
| IC | $A$ chosen among discrete values from Table 6,<br>$28 \leq b \leq 40,\ 5.0 \leq h \leq 10.0$, $b \in \mathbb{N}$ treated as a categorical variable and $h \in \mathbb{R}$ |
| IDC | $A$ chosen among categorical values from Table 6,<br>$28 \leq b \leq 40,\ 5.0 \leq h \leq 10.0$, $b \in \mathbb{N}$ treated as a categorical variable and $h \in \mathbb{R}$ |

## A.9 Rosenbrock

Minimize

$$f(x) = \sum_{i=1}^{n-1} (1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2$$

Bound constraints and variables types for

| | |
|---|---|
| Case1 | $x_1 \in \{-2,-1.5,-1,-0.5,0,0.5,1,1.5,2\},\ -2.0 \leq x_2 \leq 2.0,\ x_1$ discrete, $x_2 \in \mathbb{R}$ |
| Case2 | $-2 \leq x_1 \leq 2,\ -2.0 \leq x_2 \leq 2.0,\ x_1 \in \mathbb{N},\ x_2 \in \mathbb{R}$ |
| Case3 | $x_1 \in \{0,1,2\},\ -2.0 \leq x_2 \leq 2.0,\ x_1$ categorical, $x_2 \in \mathbb{R}$ |
| Case12 | $x_1, x_2 \in \{-2,-1.5,-1,-0.5,0,0.5,1,1.5,2\},\ x_3 \in \{-1.8,0,1,0.6,1.6\},$<br>$-2.0 \leq x_4, x_5 \leq 2.0,\ x_6 \in \{-2,-1.5,-1,-0.5,0,0.5,1,1.5,2\},\ x_7 \in \{2,0,1\},$<br>$x_8 \in \{0.5,1,-1,0.5,-2,2,-0.5,0\},\ -2 \leq x_9, x_{10} \leq 2,$<br>$x_1$ to $x_3$ discrete, $x_4, x_5 \in \mathbb{R},\ x_6$ to $x_8$ categorical, $x_9, x_{10} \in \mathbb{N}$ |
| Case22 | $x_i \in \{-2,-1.5,-1,-0.5,0,0.5,1,1.5,2\},\ \forall i = 1:8,\ x_9, x_{10} \in \{2,0,1\},$<br>$-2 \leq x_i \leq 2,\ \forall i = 11:20,$<br>$x_1$ to $x_4$ discrete, $x_5$ to $x_{10}$ categorical, $x_{11}, x_{12} \in \mathbb{N},\ x_{13}$ to $x_{20} \in \mathbb{R}$ |

## A.10 SpeedReducer

Minimize

$$\begin{aligned} f(x) &= 0.7854 x_1 x_2^2 (3.3333 x_3^2 + 14.9334 x_3 - 43.0934) - 1.508 x_1 (x_6^2 + x_7^2) \\ &\quad + 7.477 (x_6^3 + x_7^3) + 0.7854 (x_4 x_6^2 + x_5 x_7^2) \end{aligned}$$

Subject to

$$\begin{aligned}
g_1(x) &= 27 - x_1 x_2^2 x_3 \leq 0 \\
g_2(x) &= 397.5 - x_1 x_2^2 x_3^2 \leq 0 \\
g_3(x) &= 1.93 x_4^3 - x_2 x_3 x_6^4 \leq 0 \\
g_4(x) &= 1.93 x_5^3 - x_2 x_3 x_7^4 \leq 0 \\
g_5(x) &= \sqrt{(745.0 x_4)^2 + 16.9 \times 10^6 x_2^2 x_3^2} - 110 x_2 x_3 x_6^3 \leq 0 \\
g_6(x) &= \sqrt{(745.0 x_5)^2 + 157.5 \times 10^6 x_2^2 x_3^2} - 85 x_2 x_3 x_7^3 \leq 0 \\
g_7(x) &= x_2 x_3 - 40 \leq 0 \\
g_8(x) &= 5 x_2 - x_1 \leq 0 \\
g_9(x) &= x_1 - 12 x_2 \leq 0 \\
g_{10}(x) &= 1.9 + 1.5 x_6 - x_4 \leq 0 \\
g_{11}(x) &= 1.9 + 1.1 x_7 - x_5 \leq 0
\end{aligned}$$

Bound constraints and variables types for

| | |
|---|---|
| Default | $2.6 \leq x_1 \leq 3.6,\ 0.7 \leq x_2 \leq 0.8,\ 17 \leq x_3 \leq 28,\ 7.3 \leq x_4$ and $x_5 \leq 8.3$, $2.6 \leq x_6 \leq 3.9,\ 5 \leq x_7 \leq 5.5$, $x_3 \in \mathbb{N}$, $x_1$, $x_2$ and $x_4$ to $x_7 \in \mathbb{R}$ |
| IC | $2.6 \leq x_1 \leq 3.6,\ 0.7 \leq x_2 \leq 0.8,\ 17 \leq x_3 \leq 28,\ 7.3 \leq x_4$ and $x_5 \leq 8.3$, $2.6 \leq x_6 \leq 3.9,\ 5 \leq x_7 \leq 5.5$, $x_3 \in \mathbb{N}$ treated as categorical, $x_1$, $x_2$ and $x_4$ to $x_7 \in \mathbb{R}$ |

## A.11 Spring

Minimize

$$f(x) = (x_3 + 2) x_1 x_2^2$$

Subject to

$$\begin{aligned}
g_1(x) &= 71785 x_2^4 - x_1^3 x_3 \leq 0 \\
g_2(x) &= 5108 x_2^2 (4 x_1^2 - x_1 x_2) + 12566 (x_1 x_2^3 - x_2^4) - 64187128 x_2^5 (x_1 - x_2) \leq 0 \\
g_3(x) &= x_1^2 x_3 - 140.45 x_2 \leq 0 \\
g_4(x) &= x_1 + x_2 - 1.5 \leq 0
\end{aligned}$$

Bound constraints and variables types for

| | |
|---|---|
| Default | $0.25 \leq x_1 \leq 1.3,\ 0.05 \leq x_2 \leq 2.0,\ 2 \leq x_3 \leq 15,\ x_1,\ x_2 \in \mathbb{R}$ and $x_3 \in \mathbb{N}$ |
| IC | $0.25 \leq x_1 \leq 1.3,\ 0.05 \leq x_2 \leq 2.0,\ 2 \leq x_3 \leq 15,\ x_1,\ x_2 \in \mathbb{R}$ and $x_3 \in \mathbb{N}$ treated as categorical |

## A.12 SteppedCantileverBeam

Minimize

$$f(x) = l(x_1 x_2 + x_3 x_4 + x_5 x_6 + x_7 x_8 + x_9 x_{10})$$

Subject to

$$\begin{aligned}
g_1(x) &= 6Pl - \sigma_{max} x_9 x_{10}^2 \leq 0 \\
g_2(x) &= 6P(2l) - \sigma_{max} x_7 x_8^2 \leq 0 \\
g_3(x) &= 6P(3l) - \sigma_{max} x_5 x_6^2 \leq 0 \\
g_4(x) &= 6P(4l) - \sigma_{max} x_3 x_4^2 \leq 0 \\
g_5(x) &= 6P(5l) - \sigma_{max} x_1 x_2^2 \leq 0
\end{aligned}$$

$$
\begin{aligned}
g_6(x) &= \frac{Pl^3}{E}\left(244x_3x_4^3x_5x_6^3x_7x_8^3x_9x_{10}^3 + 148x_1x_2^3x_5x_6^3x_7x_8^3x_9x_{10}^3 + 76x_1x_2^3x_3x_4^3x_7x_8^3x_9x_{10}^3\right. \\
&\quad \left. + 28x_1x_2^3x_3x_4^3x_5x_6^3x_9x_{10}^3 + 4x_1x_2^3x_3x_4^3x_5x_6^3x_7x_8^3\right) - \delta_{max}x_1x_2^3x_3x_4^3x_5x_6^3x_7x_8^3x_9x_{10}^3 \leq 0 \\
g_7(x) &= x_2 - 20x_1 \leq 0 \\
g_8(x) &= x_4 - 20x_3 \leq 0 \\
g_9(x) &= x_6 - 20x_5 \leq 0 \\
g_{10}(x) &= x_8 - 20x_7 \leq 0 \\
g_{11}(x) &= x_{10} - 20x_9 \leq 0
\end{aligned}
$$

Bound constraints and variables types for

| | |
|---|---|
| Default | $x_1 \in \{1,2,3,4,5\}$, $x_2$ and $x_4 \in \{45.0, 50.0, 55.0, 60.0\}$, $x_3$ and $x_5 \in \{2.4, 2.6, 2.8, 3.1\}$, $x_6 \in \{30, 31, ..., 65\}$, $1 \leq x_7 \leq 5$, $30 \leq x_8 \leq 65$, $1 \leq x_9 \leq 5$, $30 \leq x_{10} \leq 65$ $x_1$ and $x_6 \in \mathbb{N}$, $x_2$ to $x_5$ discrete and $x_7$ to $x_{10} \in \mathbb{R}$ |
| DC | $x_1 \in \{1,2,3,4,5\}$, $x_2$ and $x_4 \in \{45.0, 50.0, 55.0, 60.0\}$, $x_3$ and $x_5 \in \{2.4, 2.6, 2.8, 3.1\}$, $x_6 \in \{30, 31, ..., 65\}$, $1 \leq x_7 \leq 5$, $30 \leq x_8 \leq 65$, $1 \leq x_9 \leq 5$, $30 \leq x_{10} \leq 65$ $x_1$ and $x_6 \in \mathbb{N}$, $x_2$ to $x_5$ categorical and $x_7$ to $x_{10} \in \mathbb{R}$ |
| IC | $x_1 \in \{1,2,3,4,5\}$, $x_2$ and $x_4 \in \{45.0, 50.0, 55.0, 60.0\}$, $x_3$ and $x_5 \in \{2.4, 2.6, 2.8, 3.1\}$, $x_6 \in \{30, 31, ..., 65\}$, $1 \leq x_7 \leq 5$, $30 \leq x_8 \leq 65$, $1 \leq x_9 \leq 5$, $30 \leq x_{10} \leq 65$ $x_1$ and $x_6 \in \mathbb{N}$ treated as categorical variables , $x_2$ to $x_5$ discrete and $x_7$ to $x_{10} \in \mathbb{R}$ |
| IDC | $x_1 \in \{1,2,3,4,5\}$, $x_2$ and $x_4 \in \{45.0, 50.0, 55.0, 60.0\}$, $x_3$ and $x_5 \in \{2.4, 2.6, 2.8, 3.1\}$, $x_6 \in \{30, 31, ..., 65\}$, $1 \leq x_7 \leq 5$, $30 \leq x_8 \leq 65$, $1 \leq x_9 \leq 5$, $30 \leq x_{10} \leq 65$ $x_1$ and $x_6 \in \mathbb{N}$ treated as categorical variables, $x_2$ to $x_5$ categorical and $x_7$ to $x_{10} \in \mathbb{R}$ |

Parameters

$$P = 50000N,\ L = 500cm,\ l = 100cm,\ \delta_{max} = 2.7cm,\ \sigma_{max} = 14000N/cm^2,\ E = 2 \times 10^7 N/cm^2$$

## References


Abramson, M. A. 2004. Mixed Variable Optimization of a Load-Bearing Thermal Insulation System Using a Filter Pattern Search Algorithm. Optimization and Engineering 5:157–177.

Abramson, M. A., C. Audet, J. W. Chrissis, and J. G. Walston. 2009a. Mesh adaptive direct search algorithms for mixed variable optimization. Optimization Letters 3:35–47.

Abramson, M. A., C. Audet, G. Couture, J. E. Dennis Jr., S. Le Digabel, and C. Tribes. Consulted on May 23rd, 2017. The NOMAD project. Software available at https://www.gerad.ca/nomad/.

Abramson, M. A., C. Audet, J. E. Dennis Jr., and S. Le Digabel. 2009b. ORTHOMADS: a deterministic MADS instance with orthogonal directions. SIAM Journal on Optimization 20:948–966.

Affenzeller, M., S. Wagner, S. Winkler, and A. Beham. 2009. Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications. Numerical Insights. CRC Press.

Audet, C., S. Le Digabel, and C. Tribes. 2009. NOMAD user guide. Cahier du GERAD, Montreal, QC, Canada .

Audet, C., and J. E. Dennis Jr. 2006. Mesh Adaptive Direct Search Algorithms for Constrained Optimization. SIAM Journal on Optimization 17:188–217.

Baert, L., C. Beauthier, M. Leborgne, and I. Lepot. 2015. Surrogate-based optimisation for a Mixed-Variable Design Space: Proof of Concept and Opportunities for Turbomachinery Applications. Montreal, Canada: ASME 2015–GT2015-43254.

Bajer, L., and M. Holeňa. 2010. Intelligent Data Engineering and Automated Learning – IDEAL 2010: 11th International Conference, Paisley, UK, September 1-3, 2010. Proceedings, chap. Surrogate Model for Continuous and Discrete Genetic Optimization Based on RBF Networks, pp. 251–258. Springer Berlin Heidelberg.

Bonami, P., L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wachter. 2008. An algorithmic framework for convex mixed integer nonlinear programs. Discrete Optimization 5:186–204.

Bonami, P., and J. Lee. 2013. BONMIN Users' Manual. Version 1.7.

Booker, A. J., J. E. Dennis Jr, P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. 1998. A Rigourous Framework for Optimization of Expensive Functions by Surrogates. ICASE Report No. 98-47 .

Cagnina, L. C., S. C. Esquivel, and C. A. C. Coello. 2008. Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer. Informatica 32:319–326.

Clarke, F. H. 1990. Optimization and Nonsmooth Analysis. SIAM.

Clarke, F. H., Y. S. Ledyaev, R. J. Stern, and P. R. Wolenski. 1998. Nonsmooth Analysis and Control Theory. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Conn, A., N. Gould, and P. Toint. 2000. Trust-Region Methods. Number 01 in MPS-SIAM Series on Optimization. Philadelphia, USA: SIAM.

Conn, A. R., K. Scheinberg, and L. N. Vicente. 2009. Introduction to Derivative-Free Optimization. Number 08 in MPS-SIAM Series on Optimization. Philadelphia, USA: SIAM.

Costa, A., and G. Nannicini. 2014. RBFOpt: an open-source library for black-box optimization with costly function evaluations. Optimization Online 2014-09-4538 .

Deb, K. 2000. An Efficient Constraint Handling Method for Genetic Algorithms. Computer Methods in Applied Mechanics and Engineering 186:311–338.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Transaction on Evolutionary Computation 6(2):181–197.

Forrester, A., A. Sobester, and A. Keane. 2008. Engineering Design via Surrogate Modelling. WILEY.

Forrester, A. I. J., and A. J. Keane. 2009. Recent advances in surrogate-based optimization. Progress in Aerospace Sciences 50–79.

Gandomi, A. H., X.-S. Yang, and A. H. Alavi. 2011. Mixed variable structural optimization using Firefly Algorithm. Computers and Structures 89:2325–2336.

Goldberg, D. E. 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley Longman.

Herrera, M., A. Guglielmetti, M. Xiao, and R. F. Coelho. 2014. Metamodel-assisted optimization based on multiple kernel regression for mixed variables. Structural and Multidisciplinary Optimization 49:979–991.

Hilal, M. A., and R. W. Boom. 1977. Optimization of mechanocal supports for large superconductive magnets. Advances in Cryogenic Engineering 22:224–232.

Holmström, K. 1997. TOMLAB - An Environment for Solving Optimization Problems in MATLAB. Stockholm, Sweden: Nordic MATLAB Conference '97, October 28-29.

Huang, M.-W., and J. S. Arora. 1997. Optimal design with discrete variables: some numerical experiments. International journal for numerical methods in engineering 40:165–188.

Kokkolaras, M., C. Audet, and J. E. Dennis Jr. 2001. Mixed Variable Optimization of the Number and Composition of Heat Intercepts in a Thermal Insulation System. Optimization and Engineering 2:5–29.

Kolda, T. G., R. M. Lewis, and V. Torczon. 2003. Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods. SIAM Review 45:385–482.

Le Digabel, S. 2011. Algorithm 909: NOMAD: Nonlinear Optimization with the MADS algorithm. ACM Transactions on Mathematical Software 37:1–15.

Liao, T., K. Socha, M. M. de Oca, T. Stutzle, and M. Dorigo. 2014. Ant Colony Optimization for Mixed-Variable Optimization Problems. Evolutionary Computation, IEEE Transactions on 18:503–518.

Madhavan, V., and P. Martiny. 2013. Accounting for manufacturability constraints in the optimisation of composites structures. Montreal, Canada: The 19th international conference on composite materials.

Mahajan, A., V. Madhavan, C. Beauthier, T. Lonfils, I. Tapeinos, and S. Koussios. 2015. High-fidelity multi-sphere hypersonic vehicle cryogenic tank design by mixed-variable surrogate-based optimization methods. Krakow, Poland: 6th European Conference for AeroSpace Sciences, EUCASS.

McCane, B., and M. H. Albert. 2008. Distance functions for categorical and mixed variables. Pattern Recognition Letters 27:986–993.

Moré, J. J., and S. M. Wild. 2009. Benchmarking Derivative-Free Optimization Algorithms. SIAM Journal on Optimization 20:172–191.

Müller, J. 2015. MISO: mixed-integer surrogate optimization framework. Optimization and Engineering 1–27.

Müller, J., C. A. Shoemaker, and R. Pichè. 2013a. SO-I: a surrogate model algorithm for expensive nonlinear integer programming problems including global optimization applications. Journal of Global Optimization 59:865–889.

Müller, J., C. A. Shoemaker, and R. Pichè. 2013b. SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems. Computers and Operations Research 40:1383–1400.

Nelder, J. A., and R. Mead. 1965. A Simplex Method for Function Minimization. The Computer Journal 7:308–313.

Newby, E., and M. M. Ali. 2015. A trust-region-based derivative free algorithm for mixed integer programming. Computational Optimization and Applications 60:199–229.

Platenga, T. D. 2009. HOPSPACK 2.0 User Manual. Tech. Rep. SAND2009-6265, Sandia National Laboratories, Albuquerque.

Powell, M. J. D. 2009. The BOBYQA algorithm for bound constrained optimization without derivatives. Tech. Rep. DAMTP 2009/NA06, Cambridge University.

Qian, P. Z. G., H. Wu, and C. F. J. Wu. 2008. Gaussian Process Models for Computer Experiments with Qualitative and Quantitative Factors. Technometrics 50:383–396.

Regis, R. G. 2014. Constrained Optimization by Radial Basis Function Interpolation for High-Dimensional Expensive Black-Box Problems with Infeasible Initial Points. Engineering Optimization 46:218–243. Online supplement.

Rios, L. M., and N. V. Sahinidis. 2013. Derivative-free optimization: a review of algorithms and comparison of software implementations. Journal of Global Optimization 56:1247–1293.

Rippa, S. 1999. An algorithm for selecting a good value for the parameter $c$ in radial basis function interpolation. Advances in Computational Mathematics 11(2-3):193–210.

Robinson, T., K. Willcox, M. Eldred, and R. Haimes. 2006. Multifidelity Optimization for Variable-Complexity Design. Portsmouth, Virginia: 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.

Sainvitu, C., V. Iliopoulou, and I. Lepot. 2009. Global Optimization with Expensive Functions - Sample Turbomachinery Design Application. Leuven, Belgium: 14th Belgian-French-German Conference on Optimization.

Sasena, M. J. 2002. Flexibility and Afficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations. Ph.D. thesis, University of Michigan.

Wilson, D., and T. Martinez. 1997. Improved heterogeneous distance functions. Journal of Artificial Intelligence Research 6:1–34.

Yang, X.-S. 2010. Engineering Optimization: An introduction with Metaheuristic Applications, chap. Test problems in optimization. John Wiley and Sons.