Derivative-free mixed binary necklace optimization for cyclic-symmetry optimal design problems

Thi Thoi Tran, Delphine Sinoquet, Sébastien da Veiga, Marcel Mongeau

Noname manuscript No.
(will be inserted by the editor)

Derivative-free mixed binary necklace optimization for cyclic-symmetry optimal design problems

Thi Thoi TRAN · Delphine SINOQUET · Sébastien DA VEIGA · Marcel MONGEAU

Received: date / Accepted: date

Abstract This paper presents an adapted trust-region method for computationally expensive black-box optimization problems with mixed binary variables that involve a cyclic symmetry property. Mixed binary problems occur in several practical optimal design problems, *e.g.,* aircraft engine turbines, mooring lines of offshore wind turbines, electric engine stators and rotors. The motivating application for this study is the optimal design of helicopter bladed disk turbomachines. The *necklace* concept is introduced to deal with the cyclic symmetry property, and to avoid costly black-box objective-function evaluations at equivalent solutions. An adapted distance is proposed for the discrete-space exploration step of the optimization method. A convergence analysis is presented for the trust-region derivative-free algorithm, DFOb-d_H, extended to the mixed-binary case and based on the Hamming distance. The convergence proof is extended to the new algorithm, DFOb-d_{neck}, which is based on the necklace distance. Computational comparison with state-of-the-art black-box optimization methods is performed on a set of analytical problems and on a simplified industrial application.

Keywords Mixed integer nonlinear programming · black-box simulation · derivative-free optimization · trust-region method · cyclic symmetry · necklace distance

Thi Thoi TRAN
IFP Énergies Nouvelles, Université de Toulouse, France. E-mail: thi-thoi.tran@ifpen.fr

Delphine SINOQUET
IFP Énergies Nouvelles, France. E-mail: delphine.sinoquet@ifpen.fr

Sébastien DA VEIGA
Safran Tech, France. E-mail: sebastien.da-veiga@safran.fr

Marcel MONGEAU
ENAC, Université de Toulouse, France. E-mail: marcel.mongeau@enac.fr

Table 1 Abbreviations and nomenclature.

DFO	: Derivative Free Optimization
MINLP	: Mixed-Integer NonLinear Programming
NLP	: NonLinear Programming
QP	: Quadratic Programming
DFOb	: DFO trust-region method with mixed binary variables
DFOb-d_H	: DFOb with Hamming distance
DFOb-d_{neck}	: DFOb with necklace distance
RBF	: Radial Basis Function
EGO	: Efficient Global Optimization
NOMAD	: Nonlinear Optimization by Mesh-Adaptive Direct Search
x	: continuous-variable vector
$\bar{x}$	: upper bound for x
$\underline{x}$	: lower bound for x
y	: binary-variable vector
z	: mixed-variable vector $z = (x, y)$
m	: number of continuous variables
n	: number of binary variables
f	: objective function
Z	: interpolation set for mixed variables
p	: cardinality of Z
$\|.\|_F$	: Fröbenius norm
$\|.\|_\infty$	: l_∞ norm
$\|.\|_2$	: l_2 norm
Δ_x	: trust-region radius relative to the x search subspace
Δ_y	: trust-region radius relative to the y search subspace
$\Delta_{x,0}$	: initial value of Δ_x
$\Delta_{y,0}$	: initial value of Δ_y
d_H	: Hamming distance
d_{neck}	: necklace distance
$Rot^r(y)$	: rotation of y by r positions

1 Introduction and motivation

This paper addresses the general black-box mixed binary optimization problem:

$$\begin{cases} \min\limits_{x,y} f(x, y) \\ x \in [\underline{x}, \bar{x}] \subset \mathbb{R}^m, y \in \{0, 1\}^n, \end{cases} \tag{1}$$

where $x \in \mathbb{R}^m$ and $y \in \{0, 1\}^n$ are continuous and binary variables, respectively. The objective function $f : \mathbb{R}^m \times \{0, 1\}^n \to \mathbb{R}$ is the output of a "black-box" numerical simulator. We then assume that f is expensive to evaluate and its derivatives are not available. In the sequel we shall often use the optimization vector z to denote a couple $(x, y) \in \mathbb{R}^m \times \{0, 1\}^n$.

There is a large body of works in the operations research community that regards Mixed-Integer NonLinear Programming problems (MINLP), see for instance [8]. Most deterministic algorithms for solving MINLP are based on branch-and-bound methods. Briefly, the branch-and-bound algorithm is based on recursively sub-dividing the set of possible solutions during the branching

step, and estimating bounds on the optimal objective-function value in each branch (the "cut" or "bound" operation) to find a solution (see *e.g.*, [20]). In [41], the authors do not create a search tree but relax the integrality constraints via a sine function that penalizes the variables for not being integers. Remark that relaxing the binary variables is not possible in our application context. While convex MINLP's can be tackled by several available software, for instance BONMIN [9] or SKIP [3], nonconvex problems are more difficult and usually require convexification and reformulation strategies. Such strategies are either impossible (reformulation) or still need to be developed when dealing with black-box optimization. This is a real challenge, especially in our context of objective functions that are computationally expensive to evaluate (several hours or even days for a single evaluation). Therefore, we choose in this paper not to focus on (meta) heuristic methods (*e.g.*, evolutionary algorithms, [24] and simulated annealing) due to the large number of objective-function evaluations such approaches require.

A widely cited Derivative Free Optimization (DFO) algorithm, NOMAD [2, 29], implements the Mesh Adaptive Direct Search (MADS) algorithm [7] for black-box optimization under general nonlinear constraints. MADS is an extension of Torczon's generalized pattern search algorithms [6, 42]. MADS principally relies on two main steps. The *search* step is flexible enough to allow local and global explorations with generic strategies such as diverse Latin-Hypercube Sampling (LHS), or variable neighbourhood search [4]. The *poll* step is critical to the local convergence proof. It involves evaluating the objective function on a discrete grid that is dynamically updated. More recently, [15] introduced a search strategy that automatically constructs quadratic models to try and find promising trial points.

Other approaches to black-box optimization rely on building an approximate model of the objective function (referred to as response surface, surrogate model, or metamodel) which include the Radial Basis Function (RBF) based optimization methods and the Efficient Global Optimization (EGO) method. The surrogate models used in these methods are *global* models, *i.e.*, they use a single substitute of the objective function that aims to be sufficiently predictive in the whole search domain to detect areas of interest with good values of the objective function (*exploration*), and that can be refined in these areas (*exploitation*). Note that these exploitation and exploration objectives are similar to the goals of the poll and search steps of MADS. The RBF method for global optimization was introduced by Gutmann [25], and several variations followed [18,27,39,40]. EGO [28] is based on a Gaussian process surrogate and an adaptive strategy to propose new evaluation points based on the so-called expected improvement criterion, which balances between exploration and exploitation. In [36], Gaussian process kernels that are products of continuous and discrete kernels are integrated into an EGO method framework; the resulting mixed *categorical* (involving integer variables not related to effective quantities) optimization problem is then solved by NOMAD. The strengths and weaknesses of various types of kernels for Gaussian processes are discussed in [38].

Opposite to the above global-model methods, are the class of trust-region methods that build local models. For instance, [16] reviews DFO trust-region methods involving quadratic-model subproblems for continuous black-box problems. In [14], an extension to mixed binary variables is considered and a proof of convergence to locally-optimal solutions is given.

The present study concentrates on the class of trust-region methods, as described in the two major references of the field [5, 16], and their extensions to mixed binary variables such as in [14] with the extra difficulty of problems involving cyclic symmetry.

Several design applications have the form or can be transformed in the form of the mixed binary nonlinear problem (1). The motivating application of this work is the optimal design of the compressor blades of a helicopter engine [35]. There are n blades. The objective is to minimize the vibration of the compressor by changing the shapes of the compressor blades. Here, a single objective-function evaluation may require several hours of computation time. This optimization problem involves a vector, $x \in \mathbb{R}^m$, of m continuous variables, each of which describes one blade shape parameter, such as the thickness or the length of the blades. There are also integer variables that locate pre-defined possible blade geometries around the disk, as in [13]. In this study, we focus on the case involving only two different blade geometries; if we consider n blades, their relative positions are indicated with a binary vector $y \in \{0, 1\}^n$, where y_i indicates whether the ith blade is of a given type (a) or of the other type, (b), for $i = 1, 2, \ldots, n$. Figure 1 (left) illustrates the case with $n = 23$ blades with the two possible types of blade geometry. Figure 1 (right) displays, for the case of $n = 6$ blades, all the distinct arrangements and their equivalent configurations obtained by rotation.

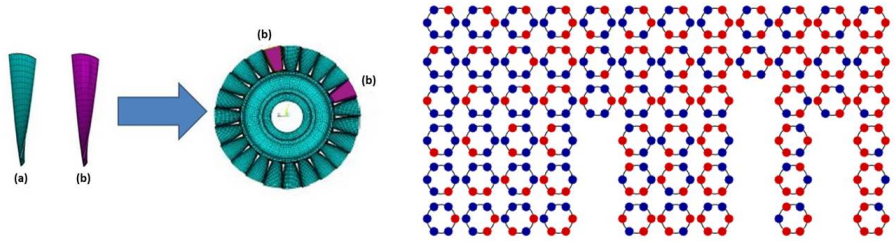

Fig. 1 Left: A 23-blade configuration (from [35]) with two different pre-defined shapes (a) and (b). Right: the first line displays the twelve possible distinct 6-blade configurations (all equivalent variants, obtained by rotation, are listed column-wise)

The cyclic-symmetry property of the problem yields a large number of equivalent arrangements: two blade disks that differ only by a rotation of the pattern around the disk not only lead to a same value of the objective function, but also correspond to identical compressors. The number of equivalent solutions also rapidly increases with n, as illustrated in Table 2. In this paper, we concentrate on avoiding recomputing during the optimization process,

the costly objective-function value at equivalent configurations. We propose a

Table 2 Number of distinct arrangements and number of total arrangements for n-blade disks

Number of blades on the disk (n)	Number of distinct arrangements ($\backsim 2^n/n$)	Total number of arrangements (2^n)
2	3	4
3	4	8
5	8	32
10	108	1 024
12	352	4 096
20	52 488	1 048 576

new DFO trust-region extended to the mixed binary case that can address the cyclic symmetry of our problem.

We wish to restrict the search to *distinct* arrangements, thereby avoiding costly black-box objective-function evaluations at equivalent solutions. To that aim, we introduce the *necklace distance*, noted d_{neck}, inspired from the concept of necklace in combinatorics [22,23], and we define a trust-region subproblem based on this new distance. Our main contribution is a new method, named DFOb-d_{neck}, which includes the necklace distance for derivative-free mixed binary optimization with cyclic-symmetry problems. We also provide a (local) convergence proof and propose a set of 25 analytical problems extended from well-known continuous optimization modified to our cyclic-symmetry and mixed-binary application context. Computational comparisons with NOMAD, RBFOpt and DFLBOX are performed on the analytical problems and on a simplified industrial simulator for the optimal design of the compressor blades.

This paper is structured as follows. Section 2 first describes a DFO algorithm, denoted by DFOb-d_H in the sequel, extended to the mixed-binary case with a mixed trust region based on the *Hamming* distance d_H for binary variables. Section 3 introduces the necklace distance, d_{neck}, and the adapted algorithm, DFOb-d_{neck}, to take into account the cyclic symmetry. In this section, some preliminary results of convergence are given. Section 4 introduces the numerical results obtained with DFOb-d_{neck}, compared with the solvers: NOMAD, RBFOpt, DFLBOX and DFOb-d_H. Then, conclusion and perspectives are given in Section 5.

2 DFOb-d_H: Trust-region derivative-free optimization method for mixed binary variables based on the Hamming distance

This section discusses the main ingredients of the DFOb-d_H algorithm, an extension of DFO trust-region methods to mixed continuous and binary variables proposed by [14]. It will serve as the key building block when proposing our new algorithm in Section 3. After summarizing the algorithm, we focus on

the theoretical proof of its local convergence due to the authors of [14] (but that was not explicitly provided in the original work [14]).

Algorithm DFOb-d_H aims at improving iteratively a starting feasible solution by solving quadratic optimization subproblems based on quadratic approximation models of the objective function. It starts with a set, Z, of *interpolation points* at each of which the objective function value is known. Each main iteration involves two main phases: exploitation and exploration. In the exploitation phase, a quadratic model, $\tilde{m}$, of the objective function is built with fixed y, then some numerical condition (*poisedness*) for the interpolation set, Z, is verified, and otherwise the interpolation set is updated. A better current solution is sought by solving trust-region quadratic optimization subproblems yielding updates of Z, and of the radii of the trust regions. The distance upon which is based the definition of the trust region for the discrete part, $\{0,1\}^n$, of the search space is the *Hamming distance*:

$$d_H(\bar{y},\tilde{y}) = \sum_{j:\tilde{y}_j=0} \bar{y}_j + \sum_{j:\tilde{y}_j=1} (1-\bar{y}_j), \tag{2}$$

for $\bar{y},\tilde{y} \in \{0,1\}^n$. Roughly speaking, this distance simply computes the minimal number of flips (from 0 to 1, or from 1 to 0) required to transform $\bar{y}$ into $\tilde{y}$. Then, an exploration phase is added to help the optimization explore wider the binary domain. The convergence result we are about to present in this section is in fact driven totally by the exploitation phase, which solves a continuous quadratic optimization subproblem by temporarily fixing the value of the discrete variables y, and by building *fully-linear* models (which will be defined in the next section) of the (continuous) objective function $f(\cdot,y)$.

2.1 The quadratic model

This subsection details how the trust-region quadratic subproblem model at iteration k, $\widetilde{m}_k$, is built.

Suppose that one is given a set of points $z^i = (x^i,y^i), x^i \in \mathbb{R}^m, y^i \in \{0,1\}^n, i=0,1,\ldots,p$, at which the objective function is evaluated with values $f^i := f(x^i,y^i), i=0,1,\ldots,p$, where $p > m+n$. This set of points is denoted by Z and is referred to as the *interpolation set*.

The derivative-free trust-region algorithm for mixed binary variables is based on the local quadratic model

$$\widetilde{m}_{\alpha,g,H}(z) = \alpha + g^T z + \frac{1}{2} z^T H z,$$

with $z=(x,y)$, $x \in \mathbb{R}^m$, $y \in \{0,1\}^n$, and where the coefficients $\alpha \in \mathbb{R}$, $g \in \mathbb{R}^{m+n}$, and H, a $(n+m)\times(n+m)$ real symmetric matrix, are solutions of the regularized fitting problem

$$\begin{cases} \min\limits_{\alpha,g,H=H^T} \dfrac{1}{2}\|H\|_F^2 \\ \widetilde{m}_{\alpha,g,H}(z^i) = f^i, i=0,1,\ldots,p, \end{cases} \tag{3}$$

where $\|.\|_F$ is the Frobenius norm. From a computational perspective, subproblem (3) can be addressed by NLP solvers such as IPOPT (see details in [48]). For the sake of notational simplicity, in the sequel, the model $\widetilde{m}_{\alpha,g,H}$ will simply be denoted $\widetilde{m}$.

The interpolation set needs to satisfy some conditions to ensure the uniqueness of the solution of the fitting problem (3). Let $d = 1$ or 2, and let $\{\phi_i\}_{i=0}^{h-1}$ be the natural basis of the space of polynomials of degree $\leq d$ in $\mathbb{R}^{m+n}$ (h is then simply the dimension of this space). In our context where $y \in \{0,1\}^n$, and when $d = 2$, the ϕ_i elements of this basis are the components of the vector:

$$\begin{aligned}\phi(z) = (1, x_1, \ldots, x_m, y_1, \ldots, y_n, \frac{1}{2}x_1^2, \ldots, \frac{1}{2}x_m^2, \ldots, x_i x_j, \ldots, x_i y_j, \ldots, \\ x_m y_n, y_1 y_2, \ldots, y_i y_j, \ldots, y_{n-1} y_n),\end{aligned} \tag{4}$$

since the purely quadratic terms in the y_i's are discarded (since $y_i^2 = y_i$). As a consequence, one has $h = (m+n+1)(m+n+2)/2 - n$.

In order to define the poisedness of the interpolation set Z, we need first to define the corresponding $(p+1) \times h$ *interpolation matrix*:

$$M := \begin{pmatrix} \phi_0(z^0) & \phi_1(z^0) & \ldots \phi_h(z^0) \\ \phi_0(z^1) & \phi_1(z^1) & \ldots \phi_h(z^1) \\ \vdots & \vdots & \vdots \\ \phi_0(z^p) & \phi_1(z^p) & \ldots \phi_h(z^p) \end{pmatrix}. \tag{5}$$

Let us consider the three possible cases for the dimensions of M (related to the number, $p+1$, of interpolation points and to the cardinality, h, of the basis – recall that $m+n+1 \leq p$):

- $h = p+1$ (*determined case*): Following [16], the interpolation set Z is said to be *poised* if the determinant of M is non-zero.
- $p+1 < h$ (*underdetermined case*): Again, as in [16], Z is *poised* if M is full column rank ($\text{rank}(M) = \min(p, h) = p$).
- $h < p+1$ (*overdetermined case*): In this case we propose to remove $p-h$ points from the interpolation set (we shall define precisely in the algorithm which points are to be eliminated), so that one falls into one of the two previous cases.

A so-called *ill-geometry* situation leading to a non-poised interpolation set occurs when for instance at some iteration, two or more interpolation points collapse or are affinely dependent. This results in non-uniqueness of solutions of the fitting problem (3). There is also an ill-geometry problem in the case of a near-singular interpolation matrix (when two interpolation points are too close to each other for example). To prevent this scenario, an improvement step based on LU factorization is set up in the mixed space $\mathbb{R}^m \times \{0,1\}^n$, inspired from the continuous version in [16], and detailed in Algorithm 1. It involves solving a MIQP (to be defined below).

Algorithm 1 provides a poised interpolation set such that when Gaussian elimination is applied to the interpolation matrix M, the absolute value of

all pivots are not smaller than the chosen threshold ξ. From a computational perspective, subproblem (6) can be addressed by an MIQP solver such as CPLEX or BONMIN.

Algorithm 1: Improving poisedness of Z in the trust region B [16]

0. Initialization

Choose an initial pivot polynomial basis with some basis $u_i(z), i = 0, 1, \ldots, h$, *e.g.*, the monomial basis $\phi(z)$ given by (4).

Select a pivot threshold $\xi > 0$.

For $i = 0, 1, \ldots, h$

1. **Point selection:**
   - If there exists an index $j \in \{i, i+1, \ldots, |Z|\}$ such that $|u_i(z^j)| \geq \xi$, swap z^j and z^i in set Z,
   - Otherwise, recompute z^i as

$$z^i \in \operatorname*{argmax}_{z \in B} |u_i(z)|, \tag{6}$$

   where B is the trust region we are considering.

   Stop if $|u_i(z^i)| < \xi$.
2. **Gaussian elimination:** For $j = i+1, i+2, \ldots, p$

$$u_j(z) \leftarrow u_j(z) - \frac{u_j(z^i)}{u_i(z^i)} u_i(z).$$

Lemma 6.7 of [16] is extended below (Lemma 1) to mixed binary variables. It guarantees the existence of the positive lower-bound value, ξ, involved in Algorithm 1 (pivot threshold).

Lemma 1 *Let $v^T\phi(z)$ be a quadratic polynomial where $\phi(z)$ is defined in (4) and $\|v\|_\infty = 1$. Then, there exists a constant $\sigma_\infty > 0$ independent of v such that*

$$\max_{x \in B(0,1), y \in \{0,1\}^n} |v^T\phi(z)| \geq \sigma_\infty, \tag{7}$$

where $B(0,1) = \{x \in \mathbb{R}^m, \|x\|_\infty \leq 1\}$.

For quadratic models, $\sigma_\infty \geq \dfrac{1}{4}$.

The proof is a straightforward extension of the proof of Lemma 6.7 in [16]. The complete proof can be found in Lemma 4.7, [46].

The introduction of binary variables requires an adapted trust-region definition. In [14], the authors introduce a l_∞-norm trust region for the continuous variables, and a Hamming-distance trust region for the binary variables.

Assuming in the sequel that the current iterate under consideration is (x_0, y_0), the mixed trust region is defined as

$$B(x_0, \Delta_x) \times \mathcal{B}(y_0, \Delta_y), \tag{8}$$

where

$$B(x_0, \Delta_x) = \{x \in \mathbb{R}^m : \|x - x_0\|_\infty \leq \Delta_x\}, \tag{9}$$

and

$$\mathcal{B}(y_0, \Delta_y) = \{y \in \{0,1\}^n : d_H(y, y_0) \leq \Delta_y\}, \tag{10}$$

for some given trust-region radii Δ_x and Δ_y.

In order to avoid ill-conditioning and ensure the local convergence of the algorithm, we rely on a class of so-called *fully-linear* models within the chosen trust region, that are defined as follows.

Definition 1 (Class of fully linear models, from [5]) Given a function $f : \mathbb{R}^m \to \mathbb{R}$, $\in \mathcal{C}^1$ and real value $\bar{\Delta} > 0$, a set of model functions $\tilde{m}(x)$, parameterized by $\Delta \in (0, \bar{\Delta}]$ is called a **fully linear** class of models of f if there exist positive constants κ_{ef}, κ_{eg} such that, given any $\Delta \in (0, \bar{\Delta}]$:

- the error between the gradient of the model and the gradient of the function satisfies, $\forall s \in B(0, \Delta)$,

$$\|\nabla f(x+s) - \nabla \tilde{m}(x+s)\| \leq \kappa_{eg}\Delta, \tag{11}$$

- the error between the model and the function satisfies

$$|f(x+s) - \tilde{m}(x+s)| \leq \kappa_{ef}\Delta^2. \tag{12}$$

As shown in [16] for continuous problems, if an interpolation set is poised, then the model obtained by solving the minimal Frobenius fitting problem is fully linear in the trust region of size $\Delta_x = \max_{i=1,2,\ldots,p}(\|x^i - x_0\|_\infty)$ defined by the interpolation points, which ensures a control of the model error by controlling the size of the trust region and the interpolation set poisedness with the model improvement step.

In our case, we ensure the local convergence of our algorithm by considering the subproblem with fixed binary variables, and by checking that the model for fixed $y = y_0$ is fully linear in the trust region:

$$B_{y_0}(x_0, \Delta_x) = \{(x, y_0) : x \in \mathbb{R}^m \text{ and } \|x - x_0\|_\infty \leq \Delta_x\}.$$

Note that, in our implementation, the model improvement step (Algorithm 1) is performed in $B(x_0, \Delta_x) \times \{0,1\}^n$, where $B(x_0, \Delta_x)$ is defined by (9). This allows a larger exploration with respect to binary variables than an improvement step in the mixed trust region $B(x_0, \Delta_x) \times \mathcal{B}(y_0, \Delta_y)$ while still fulfilling the required assumptions for Lemma 2 below. In the following, we give the proof of fully-linear models for fixed $y = y_0$.

Assumption 1

- *f is a continuously differentiable function with respect to the x variables that has a Lipschitz-continuous gradient in a closed subset, Ω, of the optimization domain, $\mathbb{R}^m \times \{0,1\}^n$;*

- *The interpolation set of $p+1$ points, Z, is poised in $B(x_0, \Delta_x) \times \{0,1\}^n$ where $p > m+n$ and $\Delta_x = \max_{i=1,2,\ldots,p}(\|x^i - x_0\|_\infty)$.*

Assumption 2

At every iteration k of the algorithm, the Frobenius norm of the model Hessian evaluated at iterate (x_k, y_k), H_k, is bounded.

Lemma 2 *Let (x_0, y_0) be the initial iterate. Under Assumption 1 and Assumption 2, the model $\widetilde{m}(\cdot, y_0)$ which is constructed from $\widetilde{m}(x, y)$ by fixing $y = y_0$ is **fully linear** in $B_{y_0}(x_0, \Delta_x)$. In other words, for all $x \in B_{y_0}(x_0, \Delta_x)$, there exist $\kappa_f^*, \kappa_g^* > 0$ such that:*

$$|f(x, y_0) - \widetilde{m}(x, y_0)| \leq \kappa_f^* \Delta_x^2, \tag{13}$$

and

$$\|\nabla_x f(x, y_0) - \nabla_x \widetilde{m}(x, y_0)\|_2 \leq \kappa_g^* \Delta_x. \tag{14}$$

The proof is given in Appendix A.

We can now state the local convergence of the algorithm.

Theorem 1 *Let Assumptions 1 and 2 hold. Then,*

$$\lim_{k \to \infty} \nabla_x f(x_k, y_0) = 0, \tag{15}$$

or all the limit points of the sequence of iterates are first-order critical points.

The proof is obtained by following the same process as in [16] (Theorem 10.13): from the results of Lemma 2, we can prove the local convergence (convergence for fixed y) of the algorithm with the additional assumption that f is bounded from below for all $(x, y) \in \Omega$, a closed subset of $\mathbb{R}^m \times \{0,1\}^n$.

Remark 1 As explained in [17], an interpolation point outside $B_{y_0}(x_0, \Delta_x)$ has to be replaced in order to ensure a fully linear model. However, in practice, in order to save expensive objective-function evaluations, we allow to go on with a model that is not certified to be fully linear when it yields effective progress in the minimization of the function.

In what follows, we detail the major stages of the DFOb-d_H algorithm.

2.2 Initial interpolation set

This subsection details the choice of the initial interpolation set (also often referred to as the *initial design*) in DFOb-d_H.

In order to construct a first quadratic model, one requires an interpolation set that contains a sufficient number of points together with the corresponding objective function values. As indicated in [16], for DFOb-d_H this number is often taken equal to $m+n+1$. Further, these points need to satisfy strict geometry conditions for the interpolation problem to be well posed. As remarked

in [47], a "good" design of experiments (DOE) not only needs to be *affinely independent*, but should additionally satisfy *space-filling, non-collapsing* properties.

There are several methods to choose a given number of sample points in a continuous space, such as factorial designs, Latin Hypercube Sample (LHS), and Optimal LHS designs (see *e.g.* [47]). However, here we deal with mixed continuous and binary variables problems: we need to provide a DOE in the mixed space $\mathbb{R}^m \times \{0,1\}^n$. When the dimension is small, one way to proceed is to sample among 2^{n+m} corner points of the boundary box: for example [25] proposes a strategy that chooses $m+n+1$ corner points plus the central point of the box. For larger dimensions, a popular strategy is the Latin Hypercube Sample (LHS) [32, 49], originally used for generating samples for continuous variables in a bounded subset. However, points sampled by this strategy will surely not satisfy our binary constraints.

In our implementation, we therefore proceed as in [18] for the RBFOpt algorithm: we first construct a Latin Hypercube Design with maximin distance criterion of $m + n + 1$ points in the considered bounded subset of $\mathbb{R}^{m+n}$, then we round the n components associated with binary variables to zero or one. Remark that rounding recovers the binary domain but may destroy the desirable properties of LHS or, even worse, it may generate identical points. A future track of our research will therefore be dedicated to improving the method for initial design of experiments.

Two main phases of DFOb-d_H remain to be specified: the exploitation and the exploration phases. The exploitation phase attempts at finding locally-optimal solutions of the optimization problem with fixed binary variables. The exploration phase focuses on escaping from local minima when we cannot improve the current local solution, by exploring the binary domain.

2.3 Exploitation phase

The next step - that will be denoted Step 1a - involves solving a continuous quadratic-programming (QP) subproblem temporary fixing the binary variables y to the associated current values of the trust region center, y_k:

$$\begin{cases} \min\limits_{x} \widetilde{m}_k(x, y_k) \\ \text{s.t.} \quad \|x - x_k\|_\infty \leq \Delta_{x,k}, \end{cases} \tag{16}$$

where $\widetilde{m}_k$ is the current model at the k^{th} iteration, (x_k, y_k) is the current iterate, and $\Delta_{x,k}$ is the trust-region radius with respect to the continuous variables x at iteration k. Note that the infinity norm l_∞ is used to define the trust region with respect to continuous variables for the sake of subproblem simplification (leading to bound constraints).

The following step, Step 1b, tests whether the solution, x^*, of (16) should be accepted based on the ratio, ρ, of the true improvement in f brought by

x^*, over the improvement predicted by the model:

$$\rho = \frac{f(x_k, y_k) - f(x^*, y_k)}{\widetilde{m}_k(x_k, y_k) - \widetilde{m}_k(x^*, y_k)}, \tag{17}$$

where one remarks that the denominator is always negative since x^* is solution of (16). We introduce η_{good}, η_{ok}, and η_{tol}, some pre-defined acceptance threshold values such that $\eta_{good} > \eta_{ok} > \eta_{tol} > 0$.

If $\rho > \eta_{tol}$, the new iterate is accepted (*successful iteration*). If $\rho < \eta_{tol}$, the solution is rejected (*unsuccessful iteration*).
This exploitation phase (referred to as Step 1 in the sequel) is summarized in Algorithm 2.

Algorithm 2: Exploitation phase of DFOb-d_H (Step 1)
at iteration k

Step 1a (TR QP)

- Solve (16) **for fixed** $\boldsymbol{y} = y_k$ in $B_{y_k}(x_k, \Delta_{x,k})$ to get x^*
- Evaluate $f(x^*, y_k)$; if $n_{simu} = \overline{n_{simu}} \rightarrow$ **STOP**
- Add $((x^*, y_k), f(x^*, y_k))$ to Z_k; $p \leftarrow p + 1$

Step 1b (Validation)

Compute the acceptance ratio ρ via (17)

- **If** $\rho \geq \eta_{tol}$ (successful Step 1): $x_k \leftarrow x^*$
  else (unsuccessful Step 1): x_k is rejected.

2.4 Exploration phase

After a successful Step 1 with fixed y_k, the following step (which will be referred to as Step 1.5a) attempts to improve the current-iterate solution, (x_k, y_k), in the mixed-variable search space by solving the mixed binary quadratic subproblem:

$$\begin{cases} \min\limits_{x,y} \widetilde{m}_k(x, y) \\ \text{s.t.} \quad \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ \qquad d_H(y, y_k) \leq \Delta_{y,k}. \end{cases} \tag{18}$$

In practice, this subproblem is addressed by MIQP solvers such as CPLEX or BONMIN.

Then, a validation step (referred to as Step 1.5b) checks if the solution of Step 1.5a provides a solution $y^* \neq y_k$, and whether the corresponding objective-function value, $f(x^*, y^*)$, associated to this solution is smaller than the current best objective-function value.

In case of an unsuccessful Step 1.5a (*i.e.*, no improvement in the minimization of f or failure in solving (18)), we continue with the same y_k, with a trust-region management with respect to the x component, and a new Step 1 to improve the current solution with y fixed to the value y_k:

- If $\rho > \eta_{good}$, then the solution (x^*, y_k) is accepted and the model is considered as a "good" predictor of f, the trust-region size is then increased;

- If $\rho \in [\eta_{ok}, \eta_{good}]$, then the solution is accepted and the model is considered as sufficiently predictive, the trust-region size remains unchanged;
- If $\rho < \eta_{ok}$, then (x^*, y_k) is rejected and the model is not considered sufficiently predictive. The trust-region radius is then reduced.

This trust-region management (which will constitute Step 2) can be summarized as:

$$\Delta_{x,k+1} = \begin{cases} 2\Delta_{x,k} & \text{if } \rho_k \geq \eta_{good}, \\ \Delta_{x,k} & \text{if } \eta_{ok} \leq \rho_k < \eta_{good}, \\ \frac{1}{2}\Delta_{x,k} & \text{if } \rho_k < \eta_{ok}. \end{cases}$$

If the solution of (18) does not yield improvement with respect to the current center (x^*, y^*), and the minimal value of the trust-region size, $\underline{\Delta_x}$, is reached, then (x^*, y^*) is considered to be a locally-optimal solution. In this case, the algorithm explores in a Step 3 the binary search space using *no-good cuts*, analogous to those introduced in [19] for general mixed optimization problems. This leads in our case to relaxing the trust-region constraint:

$$d_H(y, y^*) \leq K,$$

for some $K > 0$, and to force the algorithm to move away from the current locally-optimal solution by adding the extra (no-good cut) constraint:

$$\sum_{j:y_j^*=0} y_j + \sum_{j:y_j^*=1} (1 - y_j) \geq K^*, \tag{19}$$

where $K^* \in \mathbb{N}^*$ is some user-defined discrepancy value strictly greater than 1. Note that for a given x_k, several such no-good cut constraints are likely to cumulate, as there will be one constraint of the form (19) corresponding to each of the different y^* values obtained. The set of no-good cut constraints at iteration k is denoted by Ω_k^{NGC}.
The exploration step (Step 1.5) is outlined in Algorithm 3.

Algorithm 3: Exploration phase of DFOb-d_H (Step 1.5) **at iteration** k

Step 1.5a (MIQP subproblem)
- $is\_new\_NGC = 0$
- Solve MIQP (18) in $B(x_k, \Delta_{x,k}) \times (\mathcal{B}(y_k, \Delta_{y,k}) \cap \Omega_k^{NGC})$ to get (x^*, y^*)
- Evaluate $f(x^*, y^*)$ if $n_{simu} = \overline{n_{simu}} \rightarrow$ **STOP**
- Add $((x^*, y^*), f(x^*, y^*))$ to Z_k; $p \leftarrow p + 1$

Step 1.5b (Validation)
- **If** $y^* \neq y_k$ and $f(x^*, y^*) < \min_{(x,y) \in |Z_k| \cap (\mathbb{R}^m \times \Omega_k^{NGC})} f(x, y)$
  (successful step 1.5): $\Delta_{x,k} = \Delta_{x,0}$, (x_k, y_k)
  else (unsuccessful step 1.5)
    If $y^* \neq y_k$: $\Delta_{y,k} \leftarrow \Delta_{y,k} - 1$

The algorithm finally ends when the maximal budget of objective-function evaluations or the maximal number of no-good cuts is reached. The maximal

number of possible no-good cuts is theoretically equal to $2^n - 1$ (for n binary variables). We shall see later that in the context of cyclic symmetry, it is approximately $2^n/n$. But more importantly, with this type of property we will see that the definition of no-good cuts in (19) is not sufficient to discriminate equivalent configurations.

The DFOb-d_H algorithm is summarized in Algorithm 4.

Algorithm 4: DFOb-d_H algorithm

Initialization

- Given initial TR radii $0 < \underline{\Delta_x} < \Delta_{x,0} < \overline{\Delta_x}$, $0 < \underline{\Delta_y} < \Delta_{y,0} < \overline{\Delta_y}$, and tolerances $\eta_{good} > \eta_{ok} > \eta_{tol} > 0$, a maximal budget of evaluations $\overline{n_{simu}} > p$, maximal number of no-good cut constraints $\overline{n_{NGC}} > 0$, and a corresponding discrepancy value $K^* > 0$.
- Initial interpolation set $Z = \{(z^i, f^i)\}_{i=0,1,\ldots,p}$, $z^i = (x^i, y^i)$, $f^i = f(z^i)$
- Define initial iterate $(x_0, y_0) = \underset{i=0,1,\ldots,p}{\operatorname{argmin}} f(x^i, y^i)$
- Set $k = 0$, $\Omega_0^{NGC} = \{0,1\}^n$, $is\_new\_NGC = 0$

Iteration k:

Step 0 (Model update and improvement)
- Build quadratic model $\widetilde{m}_k(x, y)$ (cf. Subsection 2.1)
- Improve poisedness of Z_k by solving a TR MIQP in $B(x_k, \Delta_{x,k}) \times \mathcal{B}(y_k, \Delta_{y,k})$ (Algorithm 1)[a]
- **If** $n_{simu} = \overline{n_{simu}} \rightarrow$ **STOP**

if $is\_new\_NGC = 1$ go to **Step 1.5**

Step 1 (Exploitation phase): Algorithm 2 (cf. Subsection 2.3)
- **If** unsuccessful Step 1: **go to Step 2**

Step 1.5 (Exploration phase): Algorithm 3 (cf. Subsection 2.4)
- **If** successful Step 1.5: $\boldsymbol{k \leftarrow k+1}$ and **go to Step 0**

Step 2 (TR update and local convergence check)
- **If** $\rho \leq \eta_{ok}$: $\Delta_{x,k} \leftarrow \Delta_{x,k}/2$
- **If** $\rho \geq \eta_{good}$: $\Delta_{x,k} \leftarrow 2\Delta_{x,k}$
- **If** $\Delta_{x,k} > \underline{\Delta_x}$: $\boldsymbol{k \leftarrow k+1}$ and go to **Step 0**

Step 3 (Exploration after local convergence)
- Adding a new no-good cut : $\Omega_k^{NGC} \leftarrow \Omega_k^{NGC} \cap \{y \in \{0,1\}^n : d_H(y, y_k) \geq K^*\}$, $\boldsymbol{n_{NGC} \leftarrow n_{NGC} + 1}$; $\boldsymbol{is\_new\_NGC = 1}$
- Reinitialize TR radii: $\Delta_{x,k} = \Delta_{x,0}$, $\Delta_{y,k} = \overline{\Delta_y}$
- **If** $n_{NGC} \leq \overline{n_{NGC}}$: $\boldsymbol{k \leftarrow k+1}$ and go to **Step 0**
  else: $\rightarrow$ **STOP**

[a] Algorithm 1 adds possibly in Z_k new points to be simulated.

3 An adapted distance for cyclic-symmetry problems

To avoid useless costly evaluations of the numerous equivalent solutions for cyclic-symmetry problems (as illustrated in Table 2), engineers typically resort to simplifications or adapted strategies (such as the reduced-order model methodology [12,13]) to reduce the optimization problem dimension. However, such simplifications are likely to discard interesting or optimal configurations.

This section first defines a new distance to be used in DFOb so as to avoid to evaluate the costly black-box objective function at configurations that were previously evaluated (equivalent solutions), without arbitrary removal of (potentially good) candidate configurations. The new distance should lead to constraints easily manageable in efficient optimization methods, just like the Hamming distance which leads to linear constraints (of (2) and (19)). Then, in Subsection 3.2 we propose a reformulation of the algorithm optimization subproblems with this new distance for both the exploitation and the exploration phases. Subsection 3.3 summarizes the adapted algorithm, named DFOb-d_{neck}, and provides the local convergence statement.

3.1 The necklace distance

In order to avoid re-evaluating costly objective-function evaluations at equivalent blade arrangements, we propose to use the concept of *necklace* [22, 23]. In combinatorics, a *k-ary necklace of length n* is an equivalence class of n-character strings over an alphabet $\sum^k = \{a_1, a_2, \ldots, a_k\}$ of size k, considering all rotations as equivalent strings. It represents a structure with n circularly-connected characters, or *beads*, that have k available *colors* (elements of the alphabet).

Our blade design application can therefore be seen as a 2-color (or *binary*) necklace optimization problem involving a fixed number, n, of beads (the number of reference blade shapes). The number of distinct arrangements in our applicative context is therefore given by the number of n-bead necklaces: $\frac{1}{n}\sum_{d|n}\phi(d)2^{n/d}$, where ϕ is *Euler's totient function*[1] and the summation is taken over all divisors d of n.

Several applications are based on the necklace concept, with the use of various related distances: for example in music with the geometry distance and the swap distance [43–45], or in combinatorics with the Hamming distance with *shifts* [33], and the *necklace alignment distance* (NAD) based on various norms [10]. The new distance we shall use is inspired from the particular l_p necklace alignment distance (l_p NAD) with $p = 1$. Given two vectors of n real numbers, $v = (v_1, v_2, \ldots, v_n), v' = (v'_1, v'_2, \ldots, v'_n), v_i, v'_i \in [0, 1)$, the l_p NAD is defined as:

$$\min_{c,s} \sum_{i=1}^{n} (d^0((v_i + c) \bmod 1, v'_{(i+s) \bmod n}))^p, \tag{20}$$

[1] the number of positive integers between 1 and n that are relatively prime to n

where $c \in [0, 1)$ is a clockwise rotation angle of the first necklace relative to the second necklace, $s \in \{0, 1, \ldots, n-1\}$ is the *perfect matching* (best possible shift) between beads, and d^0 is the distance:

$$d^0(v_i, v'_j) = \min\{|v_i - v'_j|, (1 - |v_i - v'_j|)\}$$

(see [10] for more detail).

Taking into account the fact that our applications involve uniformly-distributed discrete locations, we set the rotation angle to the constant value $c = 0$, and we replace d^0 with the simple univariate Euler distance ($|v_i - v'_j|$). This yields the discrete necklace distance:

Definition 2 Given two k-ary necklaces of length n: $u = (u_1, u_2, \ldots, u_n)$ and $u' = (u'_1, u'_2, \ldots, u'_n)$, where $u_i, u'_i \in \{a_1, a_2, \ldots, a_k\}$, $i = 1, 2, \ldots, n$, the **discrete necklace distance** between u and u' is:

$$d^*_{neck}(u, u') = \min_{s=1,2,\ldots,n} \sum_{i=1}^{n} |u_i - u'_{i+s}|. \tag{21}$$

For the purpose of the present study which considers only *two* possible types of blade design, we focus on the case where $k = 2$ and the alphabet $\{a_1, \ldots, a_k\}$ reduces to $\{0, 1\}$. This leads to the *binary necklace distance*, denoted d_{neck}, on which our algorithm DFOb-d_{neck} will be based:

Definition 3 Given $y, y' \in \{0, 1\}^n$, the **binary necklace distance** between y and y' is:

$$d_{neck}(y, y') = \min_{i=1,2,\ldots,n} d_H(y, Rot^i(y')), \tag{22}$$

where d_H denotes the Hamming distance, and $Rot^i(y)$ is the rotation of y by i positions.

It is clear that d_{neck} is a distance since, for any $y, y', y'' \in \{0, 1\}^n$, it satisfies the following properties:

- non-negativity: $d_{neck}(y, y') \geq 0$,
- reflexivity: $d_{neck}(y, y) = 0$,
- commutativity: $d_{neck}(y, y') = d_{neck}(y', y)$,
- triangle inequality: $d_{neck}(y, y'') \leq d_{neck}(y, y') + d_{neck}(y', y'')$.

Besides, d_{neck} satisfies the key invariance property:

$$d_{neck}(y, y') = 0 \iff y \in Rot(y'), \tag{23}$$

where we define:

$$Rot(y) = \{y' \in \{0, 1\}^n : \exists i \in \{1, 2, \ldots, n\} \text{ such that } Rot^i(y') = y\}.$$

This invariance property will ensure that equivalent solutions are considered as identical solutions.

Unfortunately, contrary to the Hamming distance (see equations (2) or (19) for instance), a constraint involving the binary necklace distance cannot

be straightforwardly expressed as linear constraints (due to the "min" operator involved in the definition of d_{neck}). The next section proposes a way to address this critical issue for adapting Algorithm 4 to the new distance d_{neck} (which will replace the Hamming distance) so as to obtain an algorithm, named DFOb-d_{neck}, that deals only with linear constraints.

3.2 Reformulation of the QP subproblems involving the necklace distance

The incorporation of the new distance d_{neck} in the QP subproblems involves specific modifications in the formulation of the no-good cuts and of the trust-region constraints.

3.2.1 Necklace-distance based no-good cuts

In order to replace the Hamming distance by the necklace distance in the formulation of no-good cuts, first note that for any real numbers $a_1, a_2, \ldots, a_n$ and any positive integer K^*, one has

$$\min_{i=1,2,\ldots,n} \{a_i\} \geq K^* \Longleftrightarrow a_i \geq K^*, i = 1, 2, \ldots, n. \tag{24}$$

Now, letting $y, y_0 \in \{0,1\}^n$ and using the above equivalence with $a_i = d_H(y, Rot^i(y_0)),\ \ i = 1, 2, \ldots, n$, one straightforwardly obtains:

$$\min_{i=1,2,\ldots,n} \{d_H(y, Rot^i(y_0))\} \geq K^* \Longleftrightarrow d_H(y, Rot^i(y_0)) \geq K^*, i = 1, 2, \ldots, n, \tag{25}$$

or

$$d_{neck}(y, y_0) \geq K^* \Longleftrightarrow d_H(y, Rot^i(y_0)) \geq K^*, i = 1, 2, \ldots, n. \tag{26}$$

To summarize, one can formulate a no-good cut that avoids useless costly evaluations by using n linear constraints since (26) involves n Hamming-distance inequalities, each of which can be written under the form of a linear inequality following (2).

3.2.2 Necklace-distance based trust regions

The way we replace Hamming distances by binary necklace distances in the exploration phase (more precisely, in the trust-region mixed binary quadratic subproblem (18) of Step 1.5 of Algorithm 4) is less straightforward.

We consider the mixed binary optimization problem:

$$\begin{cases} \min\limits_{x,y} \widetilde{m}_k(x, y) \\ \text{s.t. } \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ \qquad d_{neck}(y, y_k) \leq \Delta_{y,k}, \\ \qquad y \in \{0,1\}^n, \end{cases} \tag{27}$$

where $\widetilde{m}_k : \mathbb{R}^m \times \{0,1\}^n \to \mathbb{R}$ is a quadratic function, and $x_k \in \mathbb{R}^m$, $y_k \in \{0,1\}^n$, $\Delta_{x,k}, \Delta_{y,k} \in \mathbb{R}$ are given.

We propose to replace (27) by the following perturbed problem which involves an auxiliary variable t:

$$\begin{cases} \min\limits_{x,y,t} \widetilde{m}_k(x,y) + \mu t \\ \text{s.t. } \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ \quad t = \min\limits_{i=1,2,\ldots,n} d_H(y, Rot^i(y_k)), \\ \quad t \leq \Delta_{y,k}, \\ \quad y \in \{0,1\}^n, \end{cases} \tag{28}$$

where $\mu > 0$ is a weighting parameter. We shall see in Subsection 3.3 that setting μ to a small-enough value conserves the fully-linear property of the perturbed model, thus ensuring the local convergence (Lemma 3 in next section) of the algorithm DFOb-d_{neck} to be presented.

Consider now the related mixed binary quadratic problem:

$$\begin{cases} \min\limits_{x,y,\tilde{y},t} \widetilde{m}_k(x,y) + \mu t \\ \text{s.t.} \quad \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ \quad t \geq d_H(y, Rot^i(y_k)) - M\tilde{y}_i, \quad i = 1,2,\ldots,n, \\ \quad t \leq \Delta_{y,k}, \\ \quad \sum\limits_{i=1}^{n} \tilde{y}_i = n-1, \\ \quad y, \tilde{y} \in \{0,1\}^n, \end{cases} \tag{29}$$

where M is some large-enough positive constant (one can easily verify that in fact it suffices to set M to the value $n+1$) and $\tilde{y}$ is a vector of n auxiliary binary variables.

In the sequel we shall write that two optimization problems are **equivalent** if an optimal solution of one problem straightforwardly provides an optimal solution of the other problem, and vice versa. Proposition 1 below is introduced in order to show that the new problem (29) (involving only linear constraints) is equivalent to problem (28). Since the essential difficulty resides in the "min" constraint of (28) and in the Hamming-distance constraints of (29), the proposition statement disregards the trust-region constraint on x and the constraint $t \leq \Delta_{y,k}$ (both of which are straightforwardly modeled in an MIQP). Corollary 1 below will establish the equivalence of problems (28) and (29), as a special case of Proposition 1.

Proposition 1 *(Mini-min reformulation) Let $\mu > 0$ be a given constant, N, n be positive integers, and let $f : \Omega \subseteq \mathbb{R}^N \to \mathbb{R}$ be a quadratic function, $g_i : \Omega \to \mathbb{R}, i = 1,2,\ldots,n$, be real-valued functions satisfying $0 \leq g_i(z) \leq M$, for all $z \in \Omega$, for some $M > 0$. Then, the two following optimization problems are equivalent:*

$$\begin{cases} \min\limits_{z,t} f(z) + \mu t \\ s.t. \quad t = \min\limits_{i=1,2,\ldots,n} \{g_i(z)\}. \end{cases} \tag{P_1}$$

$$\begin{cases} \min\limits_{z,\tilde{y},t} f(z) + \mu t \\ s.t. \ t \geq g_i(z) - M\tilde{y}_i, \quad i = 1, 2, \ldots, n \\ \quad \sum_{i=1}^{n} \tilde{y}_i = n - 1, \\ \quad \tilde{y}_i \in \{0, 1\}, i = 1, 2, \ldots, n. \end{cases} \tag{P_2}$$

The proof is given in Appendix B.

Corollary 1 *The two problems (28) and (29) are equivalent.*

Proof Consider the special case of Proposition 1 where $\Omega = \mathbb{R}^m \times \{0,1\}^n$, $z = (x, y)$, and $N = m + n$, and restrict both feasible sets of (P_1) and (P_2) by adding the two constraints: $\|x - x_k\|_\infty \leq \Delta_{x,k}$ and $t \leq \Delta_{y,k}$.

3.3 Algorithm DFOb-d_{neck}

The algorithm we are introducing in this paper to deal with derivative-free mixed binary optimization problems is a modified version of DFOb-d_H in which we replace the Hamming distance, d_H, with the necklace distance, d_{neck}. To do so we use the above formulation of the no-good cut constraints as linear constraints, and the reformulated MIQP subproblem. This subsection presents the new algorithm DFOb-d_{neck} and establishes its local convergence.

The new algorithm is named **Derivative-Free trust-region method for mixed binary necklace optimization**, and is noted **DFOb-d_{neck}**. It follows exactly the steps of DFOb-d_H (Algorithm 4, given at the end of Section 2), except for the following specific changes:

- In **Step 1.5a**:
  Solve MIQP subproblem (29), instead of MIQP subproblem (18).
- In **Step 3**:
  Replace, in the new no-good cut, the Hamming-distance inequality:
  $$d_H(y, y_k) \geq K^*,$$
  by the n inequalities:
  $$d_H(y, Rot^i(y_k)) \geq K^*, i = 1, 2, \ldots, n,$$
  which are linear constraints equivalent to $d_{neck}(y, y_k) \geq K^*$ by (26).

Let us now derive a result of local convergence for the new algorithm DFOb-d_{neck}, analogous to that established in Lemma 2 and Theorem 1 for DFOb-d_H. First, let us consider the perturbed model:

$$\widetilde{m}^\epsilon(x, y_0) = \widetilde{m}(x, y_0) + \epsilon, \tag{30}$$

with $\epsilon = \dfrac{\epsilon' \Delta_x^2}{n} \leq \epsilon' \Delta_x^2$, where $\epsilon' > 0$ is some small pre-defined value (in our computational results, we choose $\epsilon' = 10^{-8}$).

Lemma 3 *Under Assumption 1 and Assumption 2, the perturbed model* $\widetilde{m}^\epsilon(\cdot, y_0)$ *(defined with fixed* $y = y^0$*) is* ***fully linear*** *in* $B_{y_0}(x_0, \Delta_x)$*. In other words, for all* $x \in B_{y_0}(x_0, \Delta_x)$*, there exist positive constants* $\kappa_f^\epsilon, \kappa_g^\epsilon$ *such that:*

$$|f(x, y_0) - \widetilde{m}^\epsilon(x, y_0)| \leq \kappa_f^\epsilon \Delta_x^2, \tag{31}$$

and

$$\|\nabla_x f(x, y_0) - \nabla_x \widetilde{m}^\epsilon(x, y_0)\|_2 \leq \kappa_g^\epsilon \Delta_x, \tag{32}$$

with $\kappa_f^\epsilon = \kappa_f^* + \epsilon'$*, and* $\kappa_g^\epsilon = \kappa_g^*$*, and where* κ_f^* *and* κ_g^* *are the constants of Lemma 2.*

The proof is a straightforward adaptation of the proof of Lemma 2.

As for the convergence proof (detailed in Appendix A) for the DFOb-d_H algorithm, from Lemma 3 and with the additional assumption that f is bounded from below, we can prove that the algorithm DFOb-d_{neck} is locally convergent, following the lines of the proof of convergence of the (continuous) DFO algorithm in [16] (Theorem 10.13).

4 Numerical results

This section presents comparative numerical results. After briefly presenting the comparison methodology, we propose in Subsection 4.1 a set of benchmark mixed binary optimization problems that features cyclic symmetry. It consists of a set of 25 instances constructed by transforming existing analytical test problems from the literature, plus one completely original problem related to the design of compressor blades in a helicopter turbomachine. Subsection 4.2 reports numerical results on the 25-instance set, while Subsection 4.3 presents comparative results on the helicopter application problem.

We compare the two versions of our DFOb method (denoted DFOb-d_H for the version involving the Hamming distance, and DFOb-d_{neck} for the one with the necklace distance) with two state-of-the-art mixed integer derivative-free methods:

- The mesh adaptive direct search algorithm implemented in NOMAD software [2, 29],
- The surrogate-based optimization method implemented in RBFOpt [18] (based on radial basis functions) ,
- The derivative-free line-search bound constrained method, DFLBOX [30].

Following the methodology proposed in [34], we compare the solvers' performances in terms of number of evaluations of the objective function. This is a classical indicator for applications involving expensive objective-function evaluations where a solver is evaluated through its capacity to achieve a given function reduction within a limited budget of *simulations* (evaluations of the objective function).

In our comparisons we consider that a method solves a problem if it provides a solution $\bar{x}$ satisfying the following criterion on the objective-function value:

$$f(x_0) - f(\bar{x}) \geq (1 - \tau)(f(x_0) - f^*), \tag{33}$$

where, in the sequel, f^* denote the best function value found by any solver (or the global-minimum value, if known), x_0 is the starting point for each solver (or the best point of the initial interpolation points), and τ is the desired accuracy, a user-defined tolerance value (in our tests, $\tau = 10^{-3}$ or 10^{-5}). If a solver does not provide a solution that satisfies (33), we consider that it fails.

Performance and data profiles (see [34]) are complementary tools to compare solvers on a collection of problems.

For a given a collection of test problems, the performance profile of a solver displays the fraction of the problems that are solved by the solver with respect to some performance ratio. In our comparisons, we use the ratio between:

- the number of objective-function evaluations required to reach the chosen accuracy τ in (33) for a given solver, with:
- the number of objective-function evaluations required by the most efficient of the compared solvers (to reach the same accuracy).

The performance profile of a solver depends therefore on the other solvers tested. For instance, the value of the performance profile of a given solver for a performance ratio of 2 is the number of problems solved by this solver within less than twice the number of evaluations required by the most efficient solver for each problem. However, this does not give an accurate information on the number of evaluations required by a solver to solve a whole collection of problems. This is the reason why data profiles are widely used to compare DFO methods. Data profiles give the fraction of problems that can be solved within a given number of objective-function evaluations (this number of evaluations is often scaled by $n_v + 1$, where n_v is the number of variables of each problem). Data profiles therefore provide the performance of the solvers for any given simulation budget.

Table 3 summarizes the options and the main parameter values used in our comparison for the four solvers under study: DFOb-d_H, DFOb-d_{neck}, NOMAD , RBFOpt and DFLBOX.

4.1 New mixed binary optimization test problems featuring cyclic symmetry

This subsection details how we build 25 mixed binary cyclic-symmetry analytical benchmark problems from instances of the literature, and it also presents a simplified real-life application from Safran for designing the compressor blades of a helicopter turbomachine.

The 25 analytical benchmark problems. To our knowledge, no instance of cyclic-symmetry mixed binary optimization problems are proposed in the literature. The last two columns of Table 4 lists the name and the source of

Table 3 Solver parameters and options used for the benchmark.

All solvers	Maximal number of objective-function evaluations	300
	Initial design description	LHS design with rounding of discrete variables (default option of RBFOpt) For NOMAD and DFLBOX: the initial point is the best point of the initial design
	Initial design size	$m+n+1$ points
DFOb	Trust-region radii for continuous variables	$\Delta_{x,0}=1$ $\underline{\Delta_x}=10^{-3}, \overline{\Delta_x}=10^{2}$
	Trust-region radius for binary variables	$\Delta_{y,0}=2$
	Trust-region update parameters	$\eta_{ok}=0.01$, $\eta_{tol}=10^{-12}$, $\eta_{good}=0.9$
	Maximal number of no-good cuts	$\min(14, 2^n-1)$, if $n \leq 6$ 20, otherwise
	No-good-cut parameter	$K^*=1$

25 optimization problems that we selected to build our 25-instance benchmark problem set. Originally these benchmark problems involve continuous optimization variables only. Some of these optimization problems also include constraints. The present study considers mixed binary optimization problems featuring cyclic symmetry and involving only bound constraints. Thus, we describe now how we transform these instances from the literature into new mixed binary bound-constrained optimization problems featuring cyclic symmetry. A first step in this transformation process is to propose intermediate mixed *categorical* (involving integer variables not related to effective quantities) optimization problems.

The 10 instances from [31] are associated with continuous-optimization *minimax problems* of the form:

$$\min_{x\in[\underline{x},\bar{x}]} F(x) := \max_{w\in\{1,2,\ldots,l\}} f_w(x),$$

where $l \geq 2$, and f_1, f_2, …, f_l are given functions. We transform these instances into mixed categorical problems of the form:

$$\min_{x\in[\underline{x},\bar{x}],w\in\{1,2,\ldots,l\}} \tilde{F}(x,w) := \begin{cases} f_1(x), & \text{if } w=1, \\ f_2(x), & \text{if } w=2, \\ \quad\vdots & \\ f_l(x), & \text{if } w=l, \end{cases} \tag{34}$$

for which the integer w is the category variable.

Table 4 Analytical necklace-optimization benchmark problems

Test problem	m	n	**source instance**	**reference**
Branin-nl	1	3	Branin	[21]
Camel-nl	1	3	Camel	[21]
Goldstein-Price-nl	1	3	Goldstein-Price	[21]
Hartman3-nl	2	3	Hartman3	[21]
Hartman6-nl	5	3	Hartman6	[21]
Shekel7-nl	3	3	Shekel7	[21]
Shekel10-nl	3	3	Shekel10	[21]
HS2-nl	1	3	HS2	[26]
HS29log-nl	1	3	HS29log	[26]
HS3-nl	1	3	HS3	[26]
CB1-nl	2	2	CB1	[31]
CB2-nl	2	2	CB2	[31]
MAD1-nl	2	2	MAD1	[31]
MAD2-nl	2	2	MAD2	[31]
QL-nl	2	2	QL	[31]
Pentagon-nl	6	3	Pentagon	[31]
RosenSuzuki-nl	4	3	RosenSuzuki	[31]
WF-nl	2	2	WF	[31]
Wong2-nl	10	4	Wong2	[31]
Wong3-nl	20	6	Wong3	[31]
Perm6-nl	5	3	Perm6	[37]
Perm8-nl	7	3	Perm8	[37]
Ex8-1-1-nl	1	3	Ex8-1-1	MINLPLib [1]
Ex8-1-4-nl	1	3	Ex8-1-4	MINLPLib [1]
Sporttournament06-nl	14	3	Sporttournament06	MINLPLib [1]

For each of the 15 remaining benchmark problems, those from [21, 26, 37] and the problems from MINLPLib [1], the transformation into a mixed categorical problem goes as follows. We first restrict the last continuous variable, say x^{end}, to take only a finite number of values in the discretized-interval set:

$$X^{end} := \left\{ \underline{x}^{end} + (w-1)\frac{\bar{x}^{end} - \underline{x}^{end}}{l-1} : w = 1, 2, \ldots, l \right\},$$

where $\underline{x}^{end}$ and $\bar{x}^{end}$ are respectively the lower and upper bounds of the variable x^{end} in the original problem, and where the number, l, of categories is to be set by the user. One thereby obtains a mixed categorical optimization problem involving an objective function of the form $\tilde{F}(x, w)$, where w is a category variable ($w \in \{1, 2, \ldots, l\}$). In our numerical tests, we set the number

of categories to $l = 4$ for these 15 benchmark problems (*i.e.*, other than those that originate from minimax problems, for which l is given by the original instance).

After transforming the 25 problems into mixed categorical problems as above, we then introduce a cyclic symmetry by associating to each value of the categorical variable, w, a necklace (with all the solutions corresponding to its rotations). For instance, if a categorical variable w takes three values in $\{1, 2, 3\}$ (case where $l = 3$), the mixed categorical problem of minimizing $\tilde{F}(x, w)$ is transformed into a mixed *binary* problem with cyclic symmetry by considering the following new objective function:

$$\min_{x\in[\underline{x},\bar{x}],y\in\{0,1\}^2} f(x,y) := \begin{cases} \tilde{F}(x,1), & \text{if } y = (0,0), \\ \tilde{F}(x,2), & \text{if } y = (0,1), (1,0), \\ \tilde{F}(x,3), & \text{if } y = (1,1). \end{cases} \tag{35}$$

The resulting new mixed binary necklace-optimization instances are listed in Table 4 together with the number of continuous variables (m), the number of binary variables (n), the name of the original instance from which it was constructed together with the literature reference. The particular choice of problems is motivated by the sake of comparing methods on a diverse range of problem dimensions and difficulties, including the presence of multiple local minima.

Safran's helicopter application. As mentioned in Section 1, the present study is motivated by an application provided by Safran: proposing a design of the turbine blades of a helicopter engine that minimizes the vibrations of the compressor. This application involves $m = 1$ continuous optimization variable controlling the frequency amplitude, and a vector of $n = 12$ binary decision variables describing the layout of two reference types of blades on the turbine disk. Safran's engineers provide us a surrogate model built from costly real simulations (several hours of computer time are required for one real single simulation) to allow the computational comparison of the four optimization solvers under study.

4.2 Results obtained on the 25 analytical problems

The results obtained with DFOb-d_H, DFOb-d_{neck}, NOMAD , RBFOpt and DFLBOX over the 25 analytical problems are presented under the form of performance profiles (Figure 2) and data profiles (Figure 3). The required accuracy is set to $\tau = 10^{-3}$. The number of objective-function evaluations necessary to reach the best known solution for each problem is also displayed in Figure 4. If the solution is not reached by a solver, no point is displayed but the percentages of success are indicated for each solver. For all the numerical results, DFOb-d_H results are displayed in blue, DFOb-d_{neck} results in red, NOMAD results in black , RBFOpt in magenta, and DFLBOX in cyan.

These three figures show that for this benchmark of 25 analytical problems, DFOb-d_{neck} outperforms the four other optimization methods.

The new method DFOb-d_{neck} succeeds to solve 72% of the 25 problems, whereas RBFOpt succeeds to solve 68%, DFOb-d_H 64%, NOMAD and DFLBOX 60%.

The performance profiles of Figure 2 reveal that for a (number-of-simulation) performance ratio equal to 1 (to indicate the percentage of problems for which the solver does as well as the best solver), DFLBOX reaches the best value: 36%, while both DFOb-d_{neck} and RBFOpt reach the value: 32%.

Figure 2 also shows that the new necklace distance improves the efficiency of the DFOb method when addressing cyclic-symmetry optimization problems, as DFOb-d_H solves only 24% of the problems with the minimal number of objective-function evaluations.

In addition, the data profiles of Figure 3 illustrate that DFOb-d_{neck} solves most problems with a relatively small number of objective-function evaluations. For instance, 62% of the problems are solved within a number of simulations less than 15 times the number of variables.

Finally, Figure 4 shows that: DFLBOX requires a small number of simulations (less than 200) for the 60% of the problems it succeeds to solve, the two DFOb methods need less than 200 function evaluations for all successful problems but one (only one successful problem requires 270 function evaluations for each solver), whereas RBFOpt and NOMAD require more than 200 objective-function evaluations for 29.4% and 66.7% of their successful problems, respectively.

To summarize, the results obtained on this benchmark of 25 analytical functions demonstrate the efficiency of the DFOb methods within a limited budget of function evaluations. Moreover, 18 out of the 25 problems are solved by DFOb-d_{neck} with, generally, a smaller number of function evaluations compared with the two state-of-the-art solvers NOMAD and RBFOpt. DFLBOX method shows good performances in terms of number of simulations but is able to solve fewer problems than the DFO methods and RBFOpt.

In order to investigate the impact of problem dimension on the performance of the different methods, we analyze separately the subset of the 10 benchmark problems that involve more than 3 continuous variables (Figures 5, 6, and 7). RBFOpt and DFOb-d_{neck} succeed to solve 7 problems over 10, whereas DFOb-d_H and NOMAD only solve 5 problems, and DFLBOX only 4. These results show that, for higher-dimensional problems, RBFOpt and DFOb-d_{neck} reach the best performances, and DFLBOX the worst ones. This is not completely surprising as DFOb-d_H, DFLBOX and NOMAD are local optimization methods (with respect to the continuous search space, $\mathbb{R}^m$). DFOb-d_{neck} is also a local solver but with the help of the new distance, it succeeds to find better solution within the budget of simulations. RBFOpt is designed to address global optimization problems. Current research work is dedicated to this global-optimization issue via the proposition of a diversification search strategy for DFOb-d_{neck} in the continuous search space, $\mathbb{R}^m$.

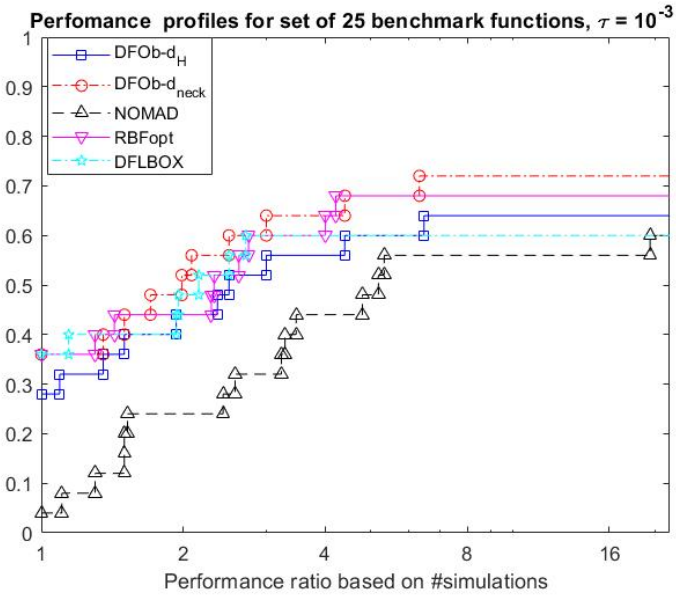

Fig. 2 Performance profiles of the for the 25 analytical problems

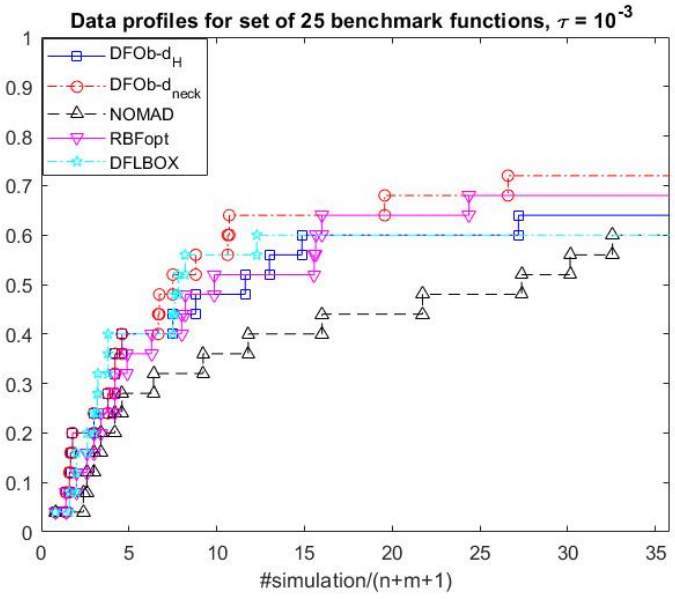

Fig. 3 Data profiles of the five solvers for the 25 analytical problems

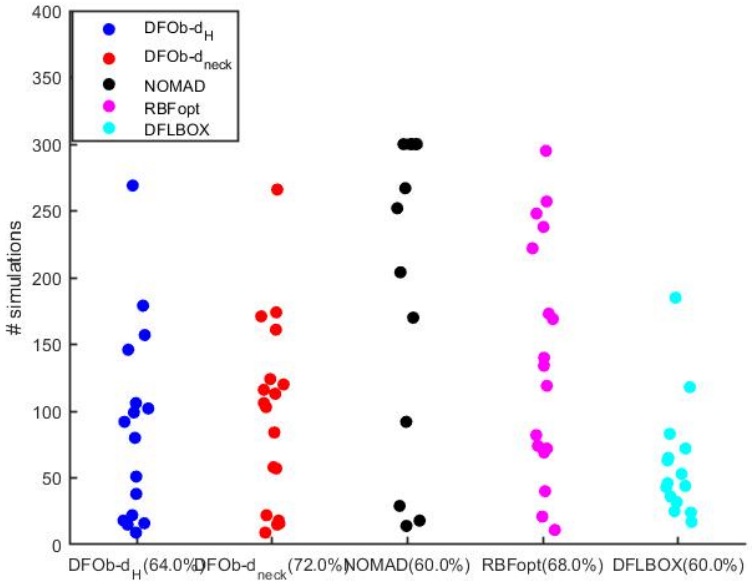

Fig. 4 Number of function evaluations to reach f^* up to $\tau = 10^{-3}$ for each of the 25 analytical problems for the five solvers (successful runs only) together with percentage of success (in parentheses)

4.3 Design of compressor blades in a helicopter turbomachine

In this subsection, we present results for a simplified optimal design application provided by Safran. Contrary to the benchmark of the previous subsection, this is a *real-life* cyclic-symmetry application.

For the sake of fair comparison, we repeat 50 runs of each of the four solvers. Each run starts with a different design of experiments (of cardinality $n+m+1 = 14$), following the construction process described in Subsection 2.2. Again, we report performance (Figure 8) and data (Figure 9) profiles, as well as the number of iterations (Figure 10) to reach, this time, a reduced accuracy of $\tau = 10^{-5}$ (due to the scale of the objective function) on the function reduction (33) associated with these 50 runs.

DFLBOX shows very good performances on this test case: it can solve all of the 50 repetitions within a very small number of simulations. As already noticed on the benchmark function results, DFLBOX is more efficient on small-dimensional problems than on large ones. For this application, which involves only one continuous variable, it clearly outperforms the four other methods. DFLBOX algorithm is based on line searches in each coordinate direction for both continuous and discrete variables. In this example, for all admissible configurations of variables, the minimum of the function corresponds to the same value of the continuous variable, so the relaxation used in DFLBOX works

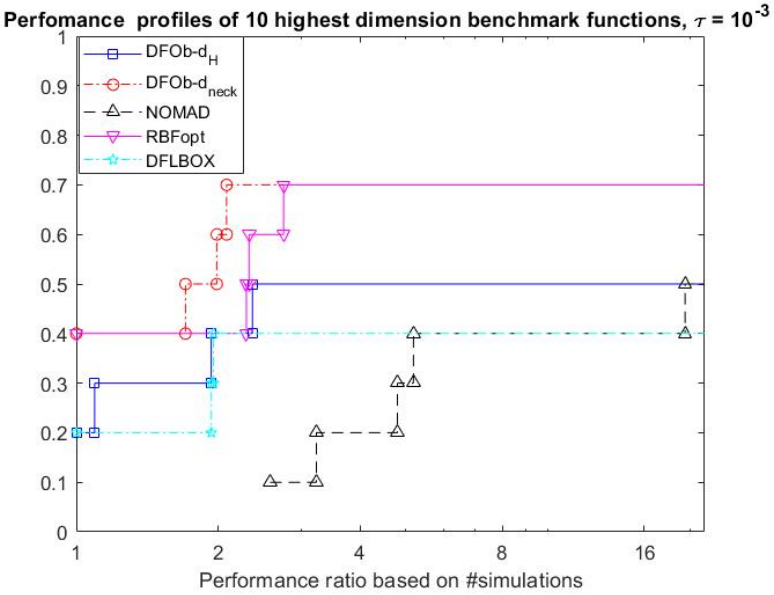

Fig. 5 Performance profiles of the five solvers for the 10 highest dimension analytical problems

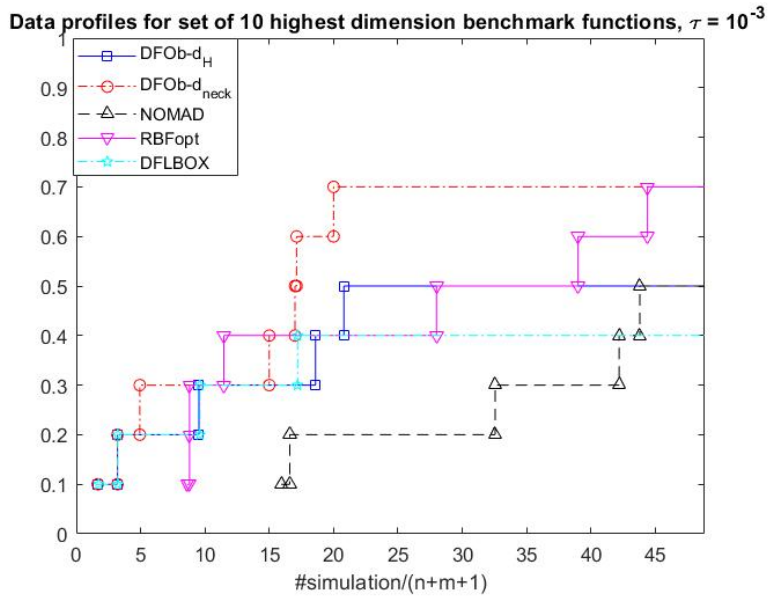

Fig. 6 Data profiles of the five solvers for the 10 highest dimension analytical problems

very well. In this particular problem, with one continuous variable and 12 bi-

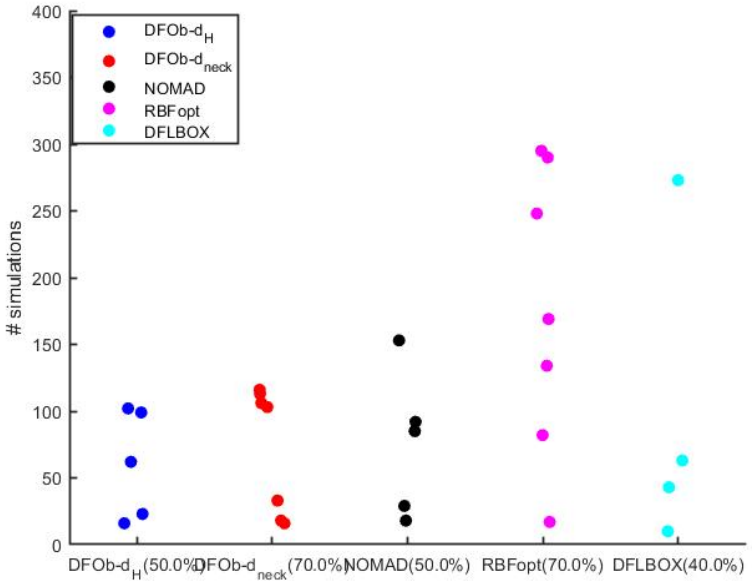

Fig. 7 Number of function evaluations to reach f^* up to $\tau = 10^{-3}$ for each of the 10 highest dimension analytical problems for the five solvers (successful runs only) together with percentage of success (in parentheses)

nary variables, this methodology is quite efficient as shown in the presented results.

Concerning the four other methods, the results show the very good performance of DFOb-d_{neck}, as it reaches the value f^* (up to τ accuracy) for 88% of the runs, compared with $84\%, 56\%$ and 78% for DFOb-d_H, NOMAD and RBFOpt, respectively. One therefore observes that, depending on the initial design of experiments, some runs do not achieve to reach the required reduction of the objective function. However, the two versions of DFOb are robust methods with regardto the initial design, with more than 80% of success within the budget of simulations.

Data profiles in Figure 9 show the robustness of the DFOb methods and their good performances in terms of number of simulations compared to RBFOpt and NOMAD methods. Figure 10 illustrates the efficiency of DFOb-d_{neck}, with only one successful run that requires more than 150 simulations.

Finally, Figure 11 displays the distribution of the 50 solutions found by DFOb-d_{neck} for each of the 50 runs. For 88% of the runs, DFOb-d_{neck} converged to a point, denoted (x^*, y_1^*), corresponding to the minimal objective-function value f^* (probably a global minimum); the remaining runs terminated with three other (locally-optimal) solutions, denoted (x^*, y_2^*), (x^*, y_3^*) and (x^*, y_4^*). All the runs converged to the same value of the continuous-variable component: $x^* = 0.03$ (corresponding to some geometry feature of

the blades). The discrete-variable components of these four different solutions found by DFOb-d_{neck} are displayed in Figure 12, where the 0 and 1 values correspond to different pre-defined types of blades.

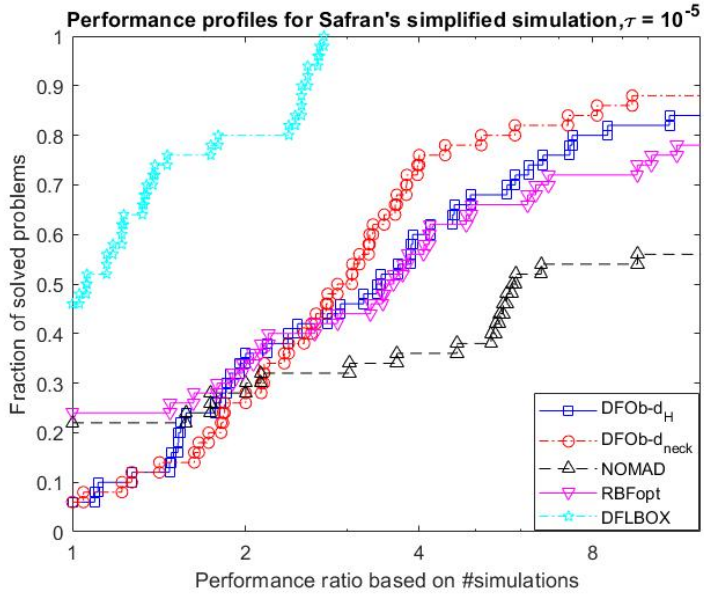

Fig. 8 Performance profiles of the four solvers for the blade design application with 50 repetitions

5 Conclusion and perspectives

In this paper we addressed derivative-free mixed binary optimization problems involving a cyclic symmetric property, by proposing an adapted distance, d_{neck}, for the binary search space. We presented theoretical results related to the linear formulation of constraints involving this *necklace* distance that allowed us to integrate d_{neck} in the trust-region derivative-free method DFOb-d_H proposed by [14] for mixed binary problems, in place of the Hamming distance. The convergence of both DFOb-d_H and that of the adapted algorithm, named DFOb-d_{neck}, to a locally-optimal solution was proved.

We proposed 25 analytical mixed binary cyclic-symmetry test problems built from a collection of continuous-optimization instances from the literature. The DFOb-d_{neck} method was evaluated on these analytical instances as well as on a surrogate approximation of a real optimal design application. Three state-of-the-art derivative-free mixed binary optimization solvers, NOMAD, DFLBOX and RBFOpt, were also applied for comparison. These preliminary results are very encouraging as our proposed method generally outperforms

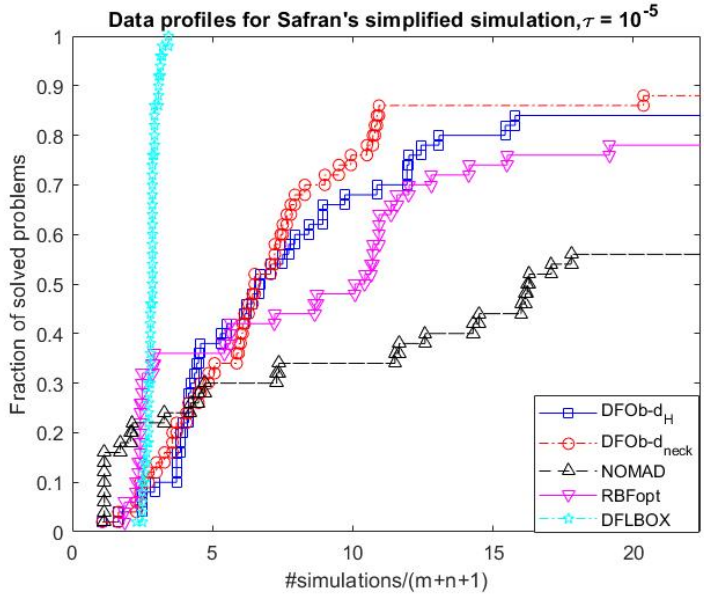

Fig. 9 Data profiles of the four solvers for the blade design application with 50 repetitions

the other methods in terms of both robustness and of number of objective-function evaluations on this benchmark.

There remains several opportunities of improvement as future work. First, since the computational results revealed the sensitivity of the results to the initial points for some of the analytical benchmark problems, we are currently working on a method to provide a design of experiments that is adapted to mixed discrete variables. Another important challenge is to improve the capacity of DFOb-d_{neck} to find solutions that are globally optimal, especially with respect to the continuous variables. A classical approach to overcome local minima is a multi-start technique [11, 49]. The high simulation costs and the presence of binary variables may however make this type of method inefficient. A coupling approach of our trust-region method with surrogate models may be a promising approach.

Acknowledgements We thank the anonymous referees for their valuable remarks and, Safran Tech and IFP Energies Nouvelles for funding the Ph.D. position of the first author.

References

1. MINLPLib URL `http://www.minlplib.org`
2. Abramson, M., Audet, C., Couture, G., J E Dennis, J., Le Digabel, S., Tribes, C.: The NOMAD project. Software available at `https://www.gerad.ca/nomad/`
3. Achterberg, T., Berthold, T., Koch, T., Wolter, K.: Constraint integer programming: A new approach to integrate CP and MIP. In: L. Perron, M. Trick (eds.) Integration

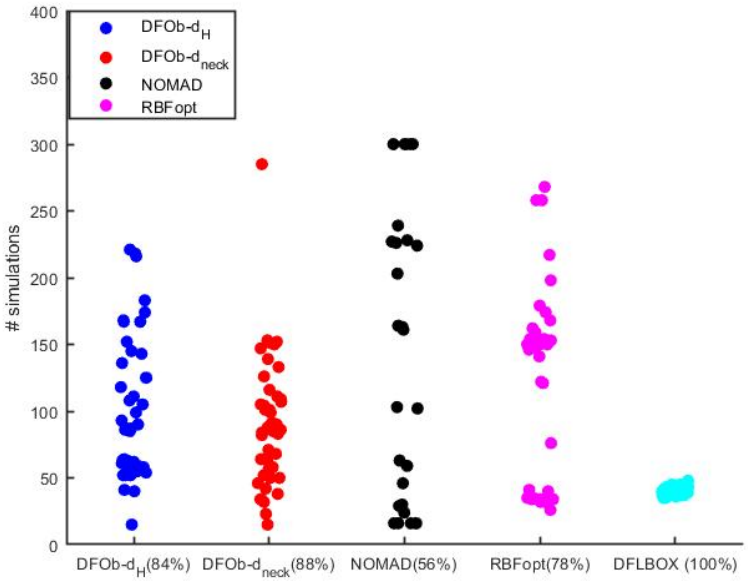

Fig. 10 Number of function evaluations to reach f^* up to $\tau = 10^{-5}$ for each of the 50 runs of the blade design application for the four solvers (successful runs only) together with percentage of success (in parentheses)

of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. LNCS 5015, CPAIOR 2008 Paris, France, pp. 6–20. Springer (2008). DOI 10.1007/978-3-540-68155-7_4

4. Audet, C., Béchard, V., Le Digabel, S.: Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search. Journal of Global Optimization **41**, 299–318 (2008). DOI 10.1007/s10898-007-9234-1
5. Audet, C., Hare, W.: Derivative-Free and Blackbox Optimization. Springer Series in Operations Research and Financial Engineering. Springer (2017). DOI 10.1007/978-3-319-68913-5
6. Audet, C., J E Dennis, J.: Analysis of generalized pattern searches. SIAM Journal on Optimization **13** (2000). DOI 10.1137/S1052623400378742
7. Audet, C., J E Dennis, J.: Mesh adaptive direct search algorithms for constrained optimization. SIAM Journal on Optimization **17**, 188–217 (2006). DOI 10.1137/060671267
8. Belotti, P., Kirches, C., Leyffer, S., Linderoth, J., Luedtke, J., Mahajan, A.: Mixed-integer nonlinear optimization. Acta Numerica **22**, 1–131 (2013). DOI 10.1017/S0962492913000032
9. Bonami, P., Biegler, L.T., Conn, A.R., Cornuéjols, G., Grossmann, I.E., Laird, C.D., Lee, J., Lodi, A., Margot, F., Sawaya, N., Wächter, A.: An algorithmic framework for convex mixed integer nonlinear programs. Discrete Optimization **5**(2), 186 – 204 (2008). URL `https://doi.org/10.1016/j.disopt.2006.10.011`. In Memory of George B. Dantzig
10. Bremner, D., Chan, T.M., Demaine, E.D., Erickson, J., Hurtado, F., Iacono, J., Langerman, S., Pătraşcu, M., Taslakian, P.: Necklaces, convolutions, and X+Y. Algorithmica **69** (2014). URL `https://doi.org/10.1007/s00453-012-9734-3`
11. Cartis, C., Roberts, L., Sheridan-Methven, O.: Escaping local minima with derivative-free methods: A numerical investigation. Optimization and Control (math.OC) (2018). DOI arXiv:1812.11343

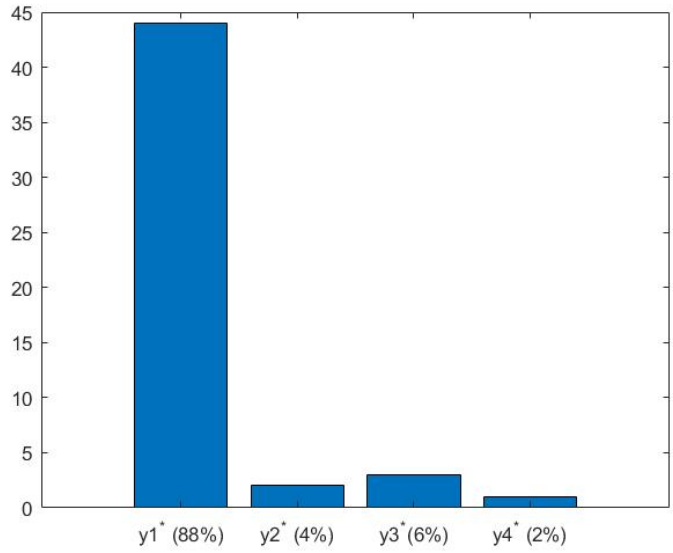

Fig. 11 Distribution of the binary component of the 50 solutions found by DFOb-d_{neck} for the blade design application.

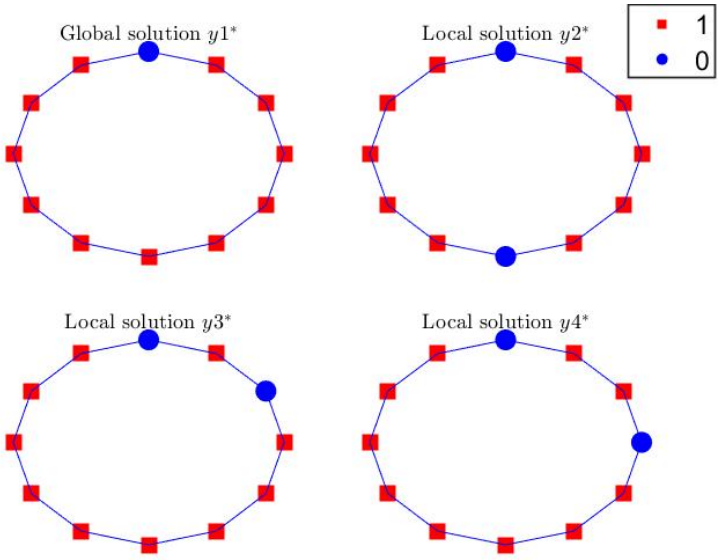

Fig. 12 The four solutions found by DFOb-d_{neck} for the blade design application (the continuous component has the same common value: $x^* = 0.03$)

12. Choi, B.: Pattern optimization of intentional blade mistuning for the reduction of the forced response using genetic algorithm. KSME International Journal **17**(7), 966–977 (2003). DOI 10.1007/BF02982981. URL https://doi.org/10.1007/BF02982981
13. Choi, B.K., Lentz, J., Rivas-Guerra, A.J., Mignolet, M.P.: Optimization of intentional mistuning patterns for the reduction of the forced response effects of unintentional mistuning: Formulation and assessment. Journal of Engineering for Gas Turbines and Power **125**(1), 131–140 (2003). DOI 10.1115/1.1498270
14. Conn, A.R., D'Ambrosio, C., Liberti, L., Sinoquet, D.: A trust region method for solving grey-box mixed integer nonlinear problems with industrial applications. SMAI-MODE 2016, Toulouse, France URL https://mode2016.sciencesconf.org/file/223761
15. Conn, A.R., Digabel, S.L.: Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. Optimization Methods & Software **28**(1), 139–158 (2013). DOI 10.1080/10556788.2011.623162. URL https://doi.org/10.1080/10556788.2011.623162
16. Conn, A.R., Scheinberg, K., Vicente, L.: Introduction to Derivative-Free Optimization. Society for Industrial and Applied Mathematics (2009). URL https://epubs.siam.org/doi/abs/10.1137/1.9780898718768
17. Conn, A.R., Scheinberg, K., Vicente, L.N.: Global convergence of general derivative-free trust-region algorithms to first- and second-order critical points. SIAM Journal on Optimization **20**(1), 387–415 (2009). DOI 10.1137/060673424. URL https://doi.org/10.1137/060673424
18. Costa, A., Nannicini, G.: RBFOpt: An open-source library for black-box optimization with costly function evaluations. Mathematical Programming Computation **10**(4), 597–629 (2018). URL https://doi.org/10.1007/s12532-018-0144-7
19. D'Ambrosio, C., Frangioni, A., Liberti, L., Lodi, A.: On interval-subgradient and no-good cuts. Operations Research Letters **38**(5), 341 – 345 (2010). DOI https://doi.org/10.1016/j.orl.2010.05.010. URL http://www.sciencedirect.com/science/article/pii/S0167637710000738
20. D'Ambrosio, C., Lodi, A.: Mixed integer nonlinear programming tools: An updated practical overview. Annals of Operations Research **204**, 1572–9338 (2013). URL https://doi.org/10.1007/s10479-012-1272-5
21. Dixon, L.C.W., Szegö, G.P.: The global optimization problem: An introduction. In: Dixon, L.C.W, Szegö, G.P (eds.) Towards Global Optimization, North Holland pp. 1–15 (1975)
22. Fredricksen, H., Kessler, I.J.: An algorithm for generating necklaces of beads in two colors. Discrete Mathematics **61**(2), 181 – 188 (1986). URL http://www.sciencedirect.com/science/article/pii/0012365X86900890
23. Gabric, D., Sawada, J.: Constructing de Bruijn sequences by concatenating smaller universal cycles. Theoretical Computer Science **743**, 12 – 22 (2018). URL https://doi.org/10.1016/j.tcs.2018.06.039
24. Gendreau, M., Potvin, J.Y.: Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol. 272. Springer (2019). 3rd edition
25. Gutmann, H.M.: A radial basis function method for global optimization. Journal of Global Optimization **19**(3), 201–227 (2001). URL https://doi.org/10.1023/A:1011255519438
26. Hock, W., Schittkowski, K.: Test examples for nonlinear programming codes. Lecture Notes in Economics and Mathematical Systems, Springer **87** (1981)
27. Holmström, K., Quttineh, N.H., Edvall, M.M.: An adaptive radial basis algorithm (ARBF) for expensive black-box mixed-integer constrained global optimization. Optimization and Engineering **9**(4), 311–339 (2008). URL https://doi.org/10.1007/s11081-008-9037-3
28. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. Journal of Global Optimization **13**(4), 455–492 (1998). URL https://doi.org/10.1023/A:1008306431147
29. Le Digabel, S.: Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm. ACM Transactions on Mathematical Software **37**(4), 1–15 (2011)
30. Liuzzi, G., Lucidi, S., Rinaldi, F.: Derivative-free methods for bound constrained mixed-integer optimization. Computational Optimization and Application **53**, 505–526 (2012). DOI 10.1007/s10589-011-9405-3

31. Lukšan, L., Vlček, J.: Test problems for nonsmooth unconstrained and linearly constrained optimization. Institute of Computer Science, Academy of Sciences of the Czech Republic. Technical report VT798-00 (2000)
32. Mckay, M.D., Beckman, R.J., Conover, W.J.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. American Statistical Association and American Society for Quality **42**, 55–61 (2010). URL :http://www.jstor.org/stable/1271432
33. Minghui, J.: On the sum of distances along a circle. Discrete Mathematics **308**(10), 2038 – 2045 (2008). DOI https://doi.org/10.1016/j.disc.2007.04.025
34. Moré, J., Wild, S.M.: Benchmarking derivative-free optimization algorithms. SIAM Journal on Optimization **20**, 172–191 (2009). DOI https://doi.org/10.1137/080724083
35. Moustapha, M.: Conception robuste en vibration et aéroélasticité des roues aubagées de turbomachines. Ph.D. thesis, Université Paris-Est Marne la Vallée, France (2009). URL https://tel.archives-ouvertes.fr/tel-00529002v2/document
36. Munoz, Z.M., Sinoquet, D.: Global optimization for mixed categorical-continuous variables based on Gaussian process models with a randomized categorical space exploration step. INFOR: Information Systems and Operational Research pp. 310–341 (2020)
37. Neumaier, A.: Neumaier's collection of test problems for global optimization pp. 1–15 (Retrieved in May 2014). URL http://www.mat.univie.ac.at/~neum/glopt/my_problems.html
38. Pelamatti, J., Brévault, L., Balesdent, M., Talbi, E.G., Guerin, Y.: Efficient global optimization of constrained mixed variable problems. Journal of Global Optimization **73**(3), 583–613 (2019). DOI 10.1007/s10898-018-0715-1
39. Regis, R., Shoemaker, C.: Improved strategies for radial basis function methods for global optimization. Journal of Global Optimization **37**, 113–135 (2007). DOI 10.1007/s10898-006-9040-1
40. Regis, R., Shoemaker, C.: A stochastic radial basis function method for the global optimization of expensive functions. INFORMS Journal on Computing **19**, 497–509 (2007). DOI 10.1287/ijoc.1060.0182
41. Shin, D.K., Gurdal, Z., O. H. Griffin, J.: A penalty approach for nonlinear optimization with discrete design variables. Engineering Optimization **16**(1), 29–42 (1990). URL https://doi.org/10.1080/03052159008941163
42. Torczon, V.: On the convergence of pattern search algorithms. SIAM Journal on Optimization **7**, 1–25 (1997)
43. Toussaint, G.: A mathematical analysis of African, Brazilian, and Cuban clave rhythms. pp. 157–168. School of Computer Science, McGill University, Montreal, Canada (2002)
44. Toussaint, G.: The geometry of musical rhythm. In: J. Akiyama, M. Kano, X. Tan (eds.) Discrete and Computational Geometry, pp. 198–212. Springer (2005)
45. Toussaint, G.: Computational geometric aspects of rhythm, melody, and voice-leading. Computational Geometry **43**(1), 2 – 22 (2010). URL http://www.sciencedirect.com/science/article/pii/S092577210900042X. Special Issue on the 14th Annual Fall Workshop
46. Tran, T.T.: Nonlinear optimization of mixed continuous and discrete variables for black-box simulators. Research report, IFP Energies Nouvelles (2020). URL https://hal-ifp.archives-ouvertes.fr/hal-02511841
47. Vu, K.K., d'Ambrosio, C., Hamadi, Y., Liberti, L.: Surrogate-based methods for black-box optimization. International Transactions in Operational Research (2017). DOI https://onlinelibrary.wiley.com/doi/pdf/10.1111/itor.12292
48. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming **106**, 25–57 (2006). DOI https://doi.org/10.1007/s10107-004-0559-y
49. Wild, S.M.: Derivative-free optimization algorithms for computationally expensive functions. Ph.D. thesis, Cornell University, Ithaca, NY, USA (2009)

Appendix A Proof of Lemma 2

Lemma 2 Let (x_0, y_0) be the initial iterate. Under Assumption 1 and Assumption 2, the model $\widetilde{m}(\cdot, y_0)$ which is constructed from $\widetilde{m}(x, y)$ by fixing

$y = y_0$ is **fully linear** in $B_{y_0}(x_0, \Delta_x)$. In other words, for all $x \in B_{y_0}(x_0, \Delta_x)$, there exist $\kappa_f^*, \kappa_g^* > 0$ such that:

$$|f(x, y_0) - \widetilde{m}(x, y_0)| \leq \kappa_f^* \Delta_x^2, \tag{36}$$

and

$$\|\nabla_x f(x, y_0) - \nabla_x \widetilde{m}(x, y_0)\|_2 \leq \kappa_g^* \Delta_x. \tag{37}$$

Proof The model constructed in mixed space is given as:

$$\widetilde{m}(z) = c + g^T z + \frac{1}{2} z^T H z,$$

where $z = (x, y)$, $g = (g_x, g_y)$, $H = \begin{pmatrix} H_{xx} & H_{xy} \\ H_{yx} & H_{yy} \end{pmatrix}$, $H_{xy} = H_{yx}$, and where H_{xx}, H_{yy} are symmetric matrices.
Thus, the model with y fixed to y_0 is defined as follows:

$$\widetilde{m}(x, y_0) = \bar{c}_x + \bar{g}_x x + \frac{1}{2} x^T \bar{H}_x x, \tag{38}$$

with $\bar{c}_x = \left(c + g_y^T y_0 + \frac{1}{2} y_0^T H_{yy} y_0\right)$, $\bar{g}_x x = \left(g_x^T + H_{xy} y_0\right)$ and $\bar{H}_x = H_{xx}$.

The gradient of $\widetilde{m}(x, y_0)$ with respect to x is therefore:

$$\nabla_x \widetilde{m}(x, y_0) = \bar{g}_x + \bar{H}_x x.$$

To be convenient, let us introduce the following notations: $f_0(x) = f(x, y_0)$, $\widetilde{m}_0(x) = \widetilde{m}(x, y_0)$, $\nabla f_0(x) = \nabla_x f(x, y_0)$, $\nabla \widetilde{m}_0(x) = \nabla_x \widetilde{m}(x, y_0)$ and $B_0(\Delta_x) = B_{y_0}(x_0, \Delta_x)$.

We define

$$\begin{aligned} err_0^f(x) &= f_0(x) - \widetilde{m}_0(x), \\ err_0^g(x) &= \nabla f_0(x) - \nabla \widetilde{m}_0(x). \end{aligned}$$

For all $x^i \in B_0(\Delta_x)$, we develop

$$\begin{aligned} (x^i - x)^T err_0^g(x) &= (x^i - x)^T (\bar{H}_x x + \bar{g}_x - \nabla f_0(x)) \\ &= (x^i - x)^T \bar{H}_x x + (x^i - x)^T \bar{g}_x - f_0(x^i) + f_0(x) \\ &+ [f_0(x^i) - f_0(x) - (x^i - x)^T \nabla f_0(x)] \\ &= m_0(x^i) - m_0(x) - \frac{1}{2}(x^i - x)^T \bar{H}_x (x^i - x) - f_0(x^i) + f_0(x) \\ &+ [f_0(x^i) - f_0(x) - (x^i - x)^T \nabla f_0(x)] \\ &= err_0^f(x^i) - err_0^f(x) - \frac{1}{2}(x^i - x)^T \bar{H}_x (x^i - x) \\ &+ [f_0(x^i) - f_0(x) - (x^i - x)^T \nabla f_0(x)]. \end{aligned} \tag{39}$$

Since f_0 is continuously differentiable, we have:

$$[f_0(x^i)-f_0(x)-(x^i-x)^T\nabla f_0(x)] = \int_0^1 (x^i-x)^T(\nabla f_0(x+t(x^i-x))-\nabla f_0(x))dt,$$

which implies that

$$\begin{aligned}(x^i-x)^T err_0^g(x) = &\int_0^1 (x^i-x)^T(\nabla f_0(x+t(x^i-x))-\nabla f_0(x))dt \\ &+ err_0^f(x^i) - err_0^f(x) - \frac{1}{2}(x^i-x)^T\bar{H}_x(x^i-x).\end{aligned} \tag{40}$$

Using (40) with $x = x_0$, we obtain:

$$\begin{aligned}(x^i-x_0)^T err_0^g(x) &= (x^i-x)^T err_0^g(x) - (x_0-x)^T err_0^g(x) \\ &= \int_0^1 (x^i-x)^T(\nabla f_0(x+t(x^i-x))-\nabla f_0(x))dt + err_0^f(x^i) \\ &- \frac{1}{2}(x^i-x)^T\bar{H}_x(x^i-x) \\ &- \int_0^1 (x_0-x)^T(\nabla f_0(x+t(x_0-x))-\nabla f(x))dt - err_0^f(x_0) \\ &+ \frac{1}{2}(x_0-x)^T\bar{H}_x(x_0-x).\end{aligned} \tag{41}$$

First, note that $err_0^f(x_0) = 0$. Then, for each terms of (41), we obtain the following upper bounds:

- From the Lipschitz property of $f_0(x)$ (Assumption 1), one has:

$$\begin{aligned}\Big|\int_0^1 (x^i-x)^T(\nabla f(x+t(x^i-x))-\nabla f(x))dt\Big| &\le \frac{1}{2}\nu\|x^i-x\|^2 \\ &\le \frac{1}{2}\nu(2\Delta_x)^2 \\ &\le 2\nu\Delta_x^2.\end{aligned} \tag{42}$$

- In the same way, we have:

$$\Big|\int_0^1 (x_0-x)^T(\nabla f(x+t(x_0-x))-\nabla f(x))dt\Big| \le \frac{1}{2}\nu\|x_0-x\|^2 \le \frac{1}{2}\nu\Delta_x^2. \tag{43}$$

- In the following two inequalities, note that $\|\bar{H}_x\|_F$ is bounded from Assumption 2:

$$|\frac{1}{2}(x^i-x)^T\bar{H}_x(x^i-x)| \le \frac{1}{2}\|\bar{H}_x\|_F\|x^i-x\|^2 \le \frac{1}{2}\|\bar{H}_x\|_F(2\Delta_x)^2 \le 2\|\bar{H}_x\|_F\Delta_x^2. \tag{44}$$

$$|\frac{1}{2}(x_0-x)^T\bar{H}_x(x_0-x)| \le \frac{1}{2}\|\bar{H}_x\|_F\|x_0-x\|^2 \le \frac{1}{2}\|\bar{H}_x\|_F\Delta_x^2. \tag{45}$$

- There exists $\epsilon' > 0$ such that

$$|err_0^f(x^i)| \leq \epsilon' \Delta_x^2, \tag{46}$$

which can be shown by contradiction. Indeed, suppose that we have

$$|err_0^f(x^i)| > \epsilon' \Delta_x^2 \quad \forall \epsilon' > 0. \tag{47}$$

By definition of err_0^f and from the continuity assumption on f_0 and $\widetilde{m}$ on $B_0(\Delta_x)$, there exist $\epsilon_1, \epsilon_2 > 0$ such that:

$$\begin{aligned} |err_0^f(x^i)| &= |f_0(x^i) - \widetilde{m}_0(y_0)| \\ &= |f_0(x^i) - f_0(x_0) + \widetilde{m}_0(x_0) - \widetilde{m}_0(x^i)| \\ &\leq |f_0(x^i) - f_0(x_0)| + |\widetilde{m}_0(x_0) - \widetilde{m}_0(x^i)| \\ &\leq (\epsilon_1 + \epsilon_2)\Delta_x. \end{aligned} \tag{48}$$

Thus, setting $\epsilon' = \dfrac{\epsilon_1 + \epsilon_2}{\Delta_{x,min}} \geq \dfrac{\epsilon_1 + \epsilon_2}{\Delta_x}$ in (47) contradicts (48).

Thus, we find from (41) and the inequalities (42-46):

$$|(x^i - x_0)^T err_0^g(x)| \leq \frac{5}{2}\Delta_x^2\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big), \tag{49}$$

with $\epsilon = \dfrac{2}{5}\epsilon'$.
Using now Cauchy-Schwarz inequality, we obtain:

$$\|err_0^g(x)\|_2 \leq \frac{5}{2}\Delta_x\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big). \tag{50}$$

Consider now the matrix $X = \dfrac{1}{\Delta_x}\big[x^1 - x_0, x^2 - x_0, \ldots, x^p - x_0\big]$.

We recall that the interpolation set Z is defined as

$$Z = \begin{pmatrix} x_0 & y_0 \\ x^1 & y^1 \\ \vdots & \vdots \\ x^p & y^p \end{pmatrix}. \tag{51}$$

Since Z is poised, Z is full rank, *i.e.*, $\text{rank}(S) = \min(p, m+n) = m+n$ based on the fact that $p > m+n$ (see Subsection 2.1), and the m column vectors $x_0, x^1, \ldots, x^p$ are linearly independent. Therefore, X^T is a non-singular matrix.

We have

$$X^T err_0^g(x) = \frac{1}{\Delta_x}\begin{bmatrix} (x^1 - x_0)^T \\ \vdots \\ (x^p - x_0)^T \end{bmatrix} err_0^g(x). \tag{52}$$

Then, we obtain from inequality (49):

$$\begin{aligned}\|X^T err_0^g(x)\|_\infty &= \frac{1}{\Delta_x}\max_{i=1,\ldots,p}|(x^i - x_0)^T err_0^g(x)| \\ &\leq \frac{1}{\Delta_x}\left[\frac{5}{2}\Delta_x^2\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big)\right] \\ &= \Delta_x\left[\frac{5}{2}\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big)\right].\end{aligned} \tag{53}$$

Moreover, we have:

$$\begin{aligned}\|err_0^g(x)\|_2 &= \|X^{-T}X^T err_0^g(x)\|_2 \\ &\leq \|X^{-T}\|_2\ \|X^T err_0^g(x)\|_2.\end{aligned} \tag{54}$$

Thus, we obtain:

$$\|err_0^g(x)\|_2 \leq \sqrt{m}\|X^{-T}\|_2\ \|X^T err_0^g(x)\|_\infty \leq \sqrt{m}\|X^{-T}\|_2\ \Delta_x\left[\frac{5}{2}\Big(\nu+\|\bar{H}_x\|_F+\epsilon\Big)\right]. \tag{55}$$

Recovering $err_0^f(x)$ by equation (40), we have:

$$\begin{aligned}|err_0^f(x)| &\leq \|err_0^g(x)\|\Delta_x + 2\nu\Delta_x^2 + 2\|\bar{H}_x\|_F\Delta_x^2 + |err_0^f(x^i)| \\ &\leq \left[\sqrt{m}\|X^{-T}\|_2\frac{5}{2}\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big) + 2(\nu + \|\bar{H}_x\|_F) + \epsilon'\right]\Delta_x^2.\end{aligned} \tag{56}$$

We complete the proof by the definition of the two required constants:

$$\kappa_g^* = \frac{5}{2}\sqrt{m}\|X^{-T}\|_2\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big), \tag{57}$$

and

$$\kappa_f^* = \Big(\nu + \|\bar{H}_x\|_F)\Big)\Big(\frac{5}{2}\sqrt{m}\|X^{-T}\|_2 + 2\Big) + \frac{5}{2}\epsilon\Big(\sqrt{m}\|X^{-T}\|_2 + 1\Big). \tag{58}$$

□

Appendix B Proof of Proposition 1

Proposition 1 Let $\mu > 0$ be a given constant, N, n be positive integers, and let $f : \Omega \subseteq \mathbb{R}^N \to \mathbb{R}$ be a quadratic function, $g_i : \Omega \to \mathbb{R}$, $i = 1, 2, \ldots, n$, be real-valued functions satisfying $0 \leq g_i(z) \leq M$, for all $z \in \Omega$, for some $M > 0$. Then, the two following optimization problems are equivalent:

$$\begin{cases} \min\limits_{z,t} f(z) + \mu t \\ \text{s.t.} \quad t = \min\limits_{i=1,2,\ldots,n} \{g_i(z)\}. \end{cases} \tag{P_1}$$

$$\begin{cases} \min\limits_{z,\tilde{y},t} f(z) + \mu t \\ \text{s.t.} \;\; t \geq g_i(z) - M\tilde{y}_i, \quad i = 1, 2, \ldots, n \\ \quad \sum_{i=1}^{n} \tilde{y}_i = n - 1, \\ \quad \tilde{y}_i \in \{0, 1\}, i = 1, 2, \ldots, n. \end{cases} \tag{P_2}$$

Proof We prove the proposition in two steps:

- firstly, we show that, (P_2) is a relaxation of (P_1) in the sense that if $(\bar{z}, \bar{t})$ is a feasible solution of (P_1), then $(\bar{z}, \bar{y}, \bar{t})$ is a feasible solution of P_2;
- secondly we prove that any optimal solution (z^*, y^*, t^*) of (P_2) is feasible for(P_1).

Let us consider the first assertion: (P_2) is a relaxation of (P_1).
Let $(\bar{z}, \bar{t})$ be a feasible solution of (P_1).
Consider now the point $(\bar{z}, \bar{y}, \bar{t})$ where, for $i = 1, 2, \ldots, n$:

$$\bar{y}_i := \begin{cases} 0, \text{ if } i \text{ is the smallest index such that } \bar{t} = \min\limits_{i=1,2,\ldots,n} \{g_i(\bar{z})\}, \\ 1, \text{otherwise.} \end{cases} \tag{59}$$

Let I be the unique index i such that $\bar{y}_i = 0$.
From the definition of $\bar{y}$, we note that:

- $\bar{t} = g_I(\bar{z})$,
- $\bar{y}_I = 0$,
- $\bar{y}_i = 1$ for all $i \neq I$.

Then, for $i \neq I$, the constraint $\bar{t} \geq g_i(\bar{z}) - M$ holds since

$$\bar{t} = g_I(\bar{z}) = \min_{i=1,2,\ldots,n} \{g_i(\bar{z})\} \geq 0 \geq g_i(\bar{z}) - M.$$

And for $i = I$, $\bar{t} \geq g_I(\bar{z}) - M$ holds also since

$$\bar{t} = g_I(\bar{z}) \geq g_I(z) - M\bar{y}_I = g_I(\bar{z}).$$

Then, $(\bar{z}, \bar{y}, \bar{t})$ is feasible for (P_2).

For the second step, let us now show that: if (z^*, y^*, t^*) is an optimal solution of (P_2) then (z^*, t^*) is feasible for (P_1), *i.e.*, we want to prove that $t^* = \min\limits_{i=1,2,\ldots,n} \{g_i(z^*)\}$.
By contradiction, we shall suppose that this optimal solution of (P_2) is such that $t^* \neq \min\limits_{i=1,2,\ldots,n} \{g_i(z^*)\}$.
Let us consider two cases:

- either $t^* < \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}$,
- or $t^* > \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}$.

Let I_{y^*} denote the unique index i such that $y_i^* = 0$.
Then, $y_{I_{y^*}} = 0$ and $y_i^* = 1$, for all $i \neq I_{y^*}$.
Using the fact that (z^*, y^*, t^*) is a feasible solution for (P_2), we have

$$t^* \geq g_{I_{y^*}}(z^*). \tag{60}$$

In the first case, with $t^* < \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}$, we have

$$g_{I_{y^*}}(z^*) \geq \min_{i=1,2,\ldots,n}\{g_i(z^*)\} > t^*,$$

which contradicts (60).

Therefore, the second case necessarily holds, *i.e.*, $t^* > \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}$. Consider now a solution $(\bar{z}, \bar{y}, \bar{t})$ defined as follows:

$$\begin{aligned} \bar{z} &= z^*, \\ \bar{t} &= \min_{i=1,2,\ldots,n}\{g_i(z^*)\}, \end{aligned} \tag{61}$$

and

$$\bar{y}_i := \begin{cases} 0, & \text{if } i \text{ is the smallest index satisfying } g_i(z^*) = \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}, \\ 1, & \text{otherwise,} \end{cases} \tag{62}$$

where $I^* = \{i : g_i(z^*) = \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}\}$. We have:

- This new solution $(\bar{z}, \bar{y}, \bar{t})$ is feasible for (P_2). Indeed, for $i \neq I^*$, the i^{th} constraint, $t \geq g_i(z) - My_i$, is satisfied for $(\bar{z}, \bar{y}, \bar{t})$, since M is an upper bound for the function $g_i(z)$ and $\bar{y}_i = 1$.
  If $i = I^*$, then on the one hand $\min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\} = \bar{t}$, and on the second hand $g_{I^*}(\bar{z}) = g_{I^*}(z^*) = \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}$ by definition of I^*. Therefore, the I^*th constraint of (P_2) is satisfied for $(\bar{z}, \bar{y}, \bar{t})$.
- In terms of objective-function values, it is clear that

  $$f(z^*) + \mu t^* > f(\bar{z}) + \mu \bar{t},$$

  since by hypothesis $t^* > \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}$, while $\bar{t} = \min\limits_{i=1,2,\ldots,n}\{g_i(z^*)\}$. This contradicts the optimality of (z^*, y^*, t^*).

□