A solution method for mixed-variable constrained blackbox optimization problems

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

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

A solution method for mixed-variable constrained blackbox optimization problems

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

Received: date / Accepted: date

Abstract Many real-world application problems encountered in industry have no analytical formulation, that is they are blackbox optimization problems, and often make use of expensive numerical simulations. We propose a new blackbox optimization algorithm named BOA to solve mixed-variable constrained blackbox optimization problems where the evaluations of the blackbox functions are computationally expensive. The algorithm is two-phased: in the first phase it looks for a feasible solution and in the second phase it tries to find other feasible solutions with better objective values. Our implementation of the algorithm constructs surrogates approximating the blackbox functions and defines subproblems based on these models. The open-source blackbox optimization solver NOMAD is used for the resolution of the subproblems. Experiments performed on instances stemming from the literature and two automotive applications encountered at Stellantis show promising results of BOA in particular with cubic RBF models. The latter generally outperforms two surrogate-assisted NOMAD variants on the considered problems.

Keywords Derivative-free optimization · blackbox optimization · mixed-variable blackbox · constrained optimization

1 Introduction

Many real-world applications necessitate solving challenging optimization problems whose hardness may stem from different factors. In particular, such prob-

Marie-Ange Dahito · Alessandro Maddaloni · José Neto
SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France
E-mail: marie-ange.dahito@polymtl.ca, alessandro.maddaloni@telecom-sudparis.eu, jose.neto@telecom-sudparis.eu

Laurent Genest
Stellantis, CEMR A, Boulevard de l'Europe, 78300 Poissy, France
E-mail: laurent.genest@stellantis.com

lems may involve different types of variables and functions whose derivatives are not available and whose evaluations result from expensive simulation runs. The present study is namely motivated by structural design optimization concerns raised by the multinational automotive manufacturing corporation Stellantis. They can be translated into a general form of constrained blackbox optimization problems for which we propose a method.

1.1 **Considered blackbox optimization problems**

In this work, we consider mixed-variable constrained blackbox optimization problems involving both an objective and constraint functions that are computationally costly to evaluate. Such problems may be formulated as follows:

$$\begin{array}{lll} \min\limits_{x\in\mathcal{X}} & f(x) & \\ s.t. & g_j(x) \leq 0, & \forall j \in J, \\ & x_i \in \mathbb{R}, & \forall i \in C, \\ & x_i \in \mathbb{Z}, & \forall i \in I, \\ & x_i \in \mathcal{D}_i, & \forall i \in D, \end{array} \tag{P}$$

where

- (C, I, D) is a partition of the indices of the variables into the subsets of indices corresponding to the continuous, integer and discrete variables, respectively, such that $C \cup I \cup D := \{1, 2, \ldots, n\}$ and n is the dimension of the problem,
- $\mathcal{D}_i$ is a finite set of ordered real values, for all $i \in D$,
- $\mathcal{X} := \{x \in \mathbb{R}^n, l_i \leq x_i \leq u_i, \forall i \in C \cup I \cup D\}$ is the admissible subset, where l_i, u_i are the lower and upper bounds of each component x_i of x,
- $f : \mathcal{X} \to \mathbb{R}$ is the objective function to minimize,
- $g_j : \mathcal{X} \to \mathbb{R}$, $\forall j \in J$ are the constraint functions, with J a finite index set.

It is also assumed that all or part of the functions f and g_j are not known explicitly, and that each evaluation of these so-called *blackbox* functions at any given point demands important computational resources in terms of time and/or memory requirements. In particular, no analytical or derivative information about them is available. Besides, the constraints $(g_j)_{j\in J}$ are considered quantifiable and relaxable and the presence of hidden constraints is envisaged [42].

Note that, unlike categorical variables, discrete variables have an intrinsic ordering. Indeed, although categorical values may be represented by numerical values, they are qualitative variables and their ordering may have no physical meaning. Discrete variables include granular values that are regularly spaced. Integer values are then a special case of granular values. The objective is to design a derivative-free optimization method providing an esteemed "*good*" solution of (P) with a limited number of blackbox function evaluations. In what

follows, by *one evaluation* at some given point x we mean the computation of the objective and all the constraints at this point. In this work we assume that all the functions (objective and constraints) are deterministic and without noise.

1.2 **Practical relevance of blackbox optimization problems**

Problems of the form (P) are relevant to many applications such as reservoir engineering [34,81], reliability design [12,14] or structural optimization [50, 71]. Solving this kind of problems is more and more requested as it can have decisive impacts, for instance in reducing greenhouse gases emissions or improving system performance. In our case, the interest for this type of problems is related to applications in the automotive manufacturing group Stellantis. A typical optimization problem is the minimization of the weight of a body-in-white structure subject to physical constraints, where all functions come from expensive computer simulations so that their differentiabilities are unknown. The variable space is usually mixed with, for instance, continuous shape parameters and discrete sheet metal thicknesses.

An additional relevant feature to be pointed out is that the problems we are interested in are presently considered as medium- to high-dimensional problems in this context since they involve more than 10 variables.

1.3 **Solution approaches**

Problems such as (P) are tackled making use of derivative-free optimization (DFO) and blackbox optimization (BBO) methods. A survey of DFO methods and comparisons on 502 problems are presented in [67]. The majority of the compared solvers in this study can only handle continuous variables. Three main approaches seem to emerge: evolutionary algorithms, direct search methods, and surrogate-based methods. They are not exclusive: there have been developments on so called *hybrid methods* combining some of these approaches, in particular the use of surrogate models in metaheuristics or direct search methods.

1.3.1 Evolutionary and swarm intelligence algorithms

An evolutionary algorithm (EA) is a derivative-free iterative method inspired from nature. From a set of points called the parent population, an offspring is generated by applying genetic operators, often crossovers and mutations. The best children according to some fitness estimate are chosen to generate the next parent population and the cycle is repeated until a stopping criterion is satisfied. EAs generally provide no convergence guarantee to a local optimum, nor indication about the quality of the obtained solution. Besides, for high-dimensional problems, the number of function evaluations to get an acceptable solution may be high.

The use of surrogates to accelerate the convergence of EAs is not rare, in particular for continuous expensive optimization problems. An example is the Constrained Evolutionary Programming assisted by Radial Basis Functions surrogates (CEP-RBF) [62], where RBF models guide the choice of the most promising children. The developed CEP-RBF is compared to other heuristic methods, including the RBF-based constrLMSRBF, on 18 benchmark problems and the 124-dimensional MOPTA08 automotive problem. The results show appreciable improvements when an evolutionary algorithm is assisted by RBF models, and substantial to important advantages compared with other methods in the context of very limited budget. Another example of RBF-assisted EA is the CONstrained Optimization by Particle swarm Using Surrogates (CONOPUS) [63], which shows advantages compared with alternative methods on 12 problems stemming from the literature and the MOPTA08 automotive application problem.

Other examples are the surrogate-assisted stochastic ranking evolution strategy of [68], the kriging-assisted scatter search SSKm [18] or the use of neural networks in [31].

Ensemble surrogates to improve the performance and robustness of EAs are employed in [44,45] for surrogate-assisted memetic algorithms or [78] on a surrogate-assisted particle swarm optimization relying on committee-based active learning.

Developments to deal with mixed-variable optimization problems include the mixed-integer evolution strategy of [85] using RBF and the Kendall rank correlation coefficient, [43] also using RBF networks and the approach of [32] for algorithm configuration problems.

1.3.2 Direct search methods

Direct search methods, that are used in DFO and BBO, proceed iteratively with the evaluation of sample points according to a certain strategy and include pattern search methods. One of the most known direct local search methods is the Mesh Adaptive Direct Search (MADS) [2,5,6], which is an extension of the Generalized Pattern Search (GPS) [74]. Under appropriate assumptions, many direct search methods guarantee convergence to a stationary point. It is the case for MADS which exhibits global convergence properties.

As its name indicates, MADS proceeds on a mesh and first performs a global search, from an initial guess. If the incumbent cannot be replaced by a better point after this phase, a set of points is evaluated in its neighbourhood: it is called the poll. The two phases are performed until a stopping criterion is met. The poll points are defined from a set of positive spanning directions that are updated iteratively and become asymptotically dense in $\mathbb{R}^n$. Two grid parameters called the poll size and the mesh size are updated at every iteration such that they increase when a better candidate is found and decrease otherwise. The algorithm is implemented in the Nonlinear Optimization with the MADS algorithm (NOMAD) open-source software [7] which handles blackbox constraints as well as integer, granular and categorical variables. The global

convergence of MADS relies on the poll step of the algorithm and is proven, under some assumptions such as bounded level sets of the objective function, for several cases including continuous [5] but also mixed-variable [1] optimization problems.

The fact of using surrogate models inside direct search methods enables to guide the search. A surrogate management framework (SMF) is presented in [10] for optimizing expensive bound-constrained blackbox optimization problems on which the use of traditional quasi-Newton methods is not appropriate. It is a pattern search method using surrogate approximations and featuring a search and a poll phase. In [49], kriging models and biharmonic splines models are used in the SMF for the resolution of trailing-edge shape optimization problems of maximum 5 variables. The experiments exhibit robustness and effectiveness of the SMF. Another example of use of the SMF with MADS is [48] for the resolution of cardiovascular geometries problems.

This framework is adapted to general constraints in [23] where treed Gaussian processes are used in the search step of MADS. The proposed extension is tested on 5 problems arising from the literature and real-world applications with dimensions between 2 and 8 and show improving performance on the MADS algorithm.

Adaptations to mixed variables also exist. The resolution of constrained expensive blackbox optimization problems with mixed variables is tackled in [15]. To do that, a testbed of 37 literature and physical problems with 2 to 20 variables is used as well as a 32-dimensional application modelling a thermal insulation system. Experiments show better efficiency and robustness of NOMAD when combined with RBF.

1.3.3 Surrogate model-based methods

The basic idea of surrogate-based methods for blackbox optimization is to make use of so called *surrogate functions*, also referred to as metamodels or response surfaces, instead of the blackbox ones to guide the search to an optimal solution, while avoiding costly evaluations. The whole process of surrogate-based methods may then be summarized in a very generic way by the following steps:

- Step 1 [Initialization]: Generate some set of points, called the Design of Experiment (DOE), and evaluate the blackbox functions at each of those points. Use these evaluations to determine initial surrogates $\widehat{f}, \widehat{g}_j, j \in J$, approximating the objective and constraints, respectively.
- Step 2 [Sample point(s) generation]: Determine a restricted set of points Γ on which to evaluate the blackbox functions with a selection procedure involving the current surrogates.
- Step 3 [Evaluation of sample point(s)]: Evaluate the blackbox functions at each point of Γ.
- Step 4 [Surrogate update]: Update the surrogates using all the available evaluations, and return to Step 2 until some stopping criterion is satisfied.

There are many possible ways of implementing each of these four steps for which the reader is referred to, e.g. [52, 76] and the references therein.

Several papers review the state-of-the-art surrogate models. As examples, [20] and [77] review the main types of metamodels for surrogate-based optimization and the authors of [3] study surrogate techniques with applications to groundwater modelling.

Some kriging and RBF-based approaches are investigated in [65] to solve constrained blackbox global optimization problems where at least one function among the objective and constraints is expensive to evaluate. The RBF-based optimization methods include COBRA [61], ConstrLMSRBF [60] and CONORBIT [66].

COBRA first finds a feasible sample by iteratively minimizing the sum of the squared constraint violations subject to the constraints added with slacks, and minimum distance constraint to already evaluated points. Then, it looks for a better sample by iteratively solving other subproblems with the same constraints but minimizing the objective function. The MATLAB function `fmincon`, using a gradient-based sequential quadratic programming (SQP) method, is used for the subproblems. Variants called COBRA-R [38] and SACOBRA [8] aim at dealing with respectively the difficulty of finding a feasible point of the subproblems and the sensitivity to the parameterization.

ConstrLMSRBF is a heuristic that also uses RBF models for the objective and constraints and where the sample points are selected among random generations of points, usually from a Gaussian distribution around the incumbent. Among the generated points with the minimum number of predicted constraint violations, a sample point is chosen according to two criteria that are the predicted objective function value and the minimum distance from previous samples. An extended ConstrLMSRBF presented in [61] deals with finding an initial feasible point by using the two-phase approach of COBRA.

Finally, CONORBIT [66] is a trust-region algorithm that is an extension of the ORBIT algorithm [80]. It builds RBF models of the objective and constraints by selecting points only in a trust region of the current iterate. The next sample point is chosen by minimizing a local subproblem defined by the surrogates and a small margin for the RBF constraints. Another example using RBF in a trust-region algorithm is TARBF of [46].

Kriging-based approaches include the Efficient Global Optimization (EGO) method introduced in [36] and its extensions to constrained optimization such as SuperEGO proposed in [70] that uses a penalized expected improvement in case of inequality constraints. Extensions of SuperEGO to constrained high-dimensional problems, called SEGOKPLS(+K) [11], use kriging with partial least squares.

Some methods handle discrete problems as is the case for SO-I, presented in [55], that solves integer constrained expensive blackbox optimization problems using RBF functions. When no feasible point is known, it first iteratively minimizes the sum of constraint violations. Once a feasible point is found, the second phase consists in minimizing a penalty augmented objective function. Experiments show that the RBF-based methods, including SO-I, have a

generally better performance than the other compared methods. CONDOR is introduced in [64]. It is an RBF-based method for high-dimensional discrete blackbox problems that performs various perturbations of the incumbent to find a better point. It exhibits promising results on a 222-dimensional automotive problem with 54 constraints.

Few surrogate-based methods handle mixed variables. MISO [52] is a framework using RBF for the resolution of mixed-integer unconstrained blackbox optimization problems. It implements different sampling strategies and shows efficiency in solution improvement with cubic RBF models, compared with NOMAD version 3.6.2 and MATLAB's genetic algorithm. SO-MI [54] also deals with mixed-integer problems using RBF but also handle constraints. The DOE is assumed to contain at least one feasible point and the method models a penalty augmented objective function to evaluate 4 points at each iteration. The experiments on 21 problems arising from the literature or applications show it is generally better than the compared methods: a branch-and-bound algorithm, a genetic algorithm and NOMAD version 3.5. Multivariate Adaptive Regression Spline (MARS) is used in MARSOPT [50], a mixed-integer linear program to optimize non-convex piecewise linear MARS models with constraints involving linear regression and piecewise linear MARS models. The method is tested in the context of a vehicle crash on a safety system design application with mixed continuous and binary variables and 50 constraints. On this problem, different MARSOPT models were compared with a customized genetic algorithm using penalties, showing a better efficiency of MARSOPT to find good solutions.

Other methods for dealing with mixed variables include GOSAC of [56] using RBF for mixed-integer problems with expensive constraints and a cheap objective.

1.4 **Our contributions**

We introduce a generic solution method which integrates ideas emanating from different recent research papers and some original features are proposed with the aim of dealing with instances that are presently considered as "*medium-*" and "*high-dimensional*" in the area of BBO. The proposed method, described in Sect. 2, does not require an initial feasible solution and is intended to provide competitive solutions for mixed-variable BBO problems with expensive objective and constraint functions. Our implementation takes into account the possibility of failures of the blackbox that can occur in real-world optimization problems. A parallel version of the algorithm is also designed to make use of the possibility of simultaneous blackbox calls. Sect. 3 presents benchmarks from the literature and some practical problems including structural design optimization problems from the automotive industry. These optimization instances are used for computational experiments in Sect. 4, which provide some hints on the relative contributions of different key parts of the algorithm for its performance. Finally, the results of the numerical tests and some perspectives

are discussed in Sect. 5.

Notation. Let S be a real subspace, S_+ denotes its nonnegative values, S^* means that 0 is removed and S_+^* stands for the strictly positive values of S. The cardinality of a set I is indicated as $|I|$. Furthermore, we use $\|\cdot\|$ for the Euclidean norm. Finally, $\widehat{f}$ is used to denote a surrogate model of a real-valued function f.

2 Description of the proposed algorithm

This section presents a *Blackbox Optimization Algorithm*, called BOA, through its structure and main features.

2.1 The overall layout of BOA

2.1.1 General description

As is usually the case in constrained BBO algorithms not requiring an initial feasible solution in the inputs, the proposed method BOA is made up of two successive phases that we call "Phase I" and "Phase II", respectively. Phase I is intended to identify a feasible solution of (P), while Phase II aims at iteratively improving the best feasible solution found so far, both phases taking into account a limited budget w.r.t. the number of function evaluations.

The general layout of BOA is described in Algorithm 1 and starts with an initialization of the algorithm parameters, that we introduce later in Sect. 2.1.3. Then, an initial DOE is evaluated, among which a best point is identified and initialized as $x_{\texttt{best}}$. This best solution, which will be updated after each iteration of BOA, is chosen among the evaluated points as a point that minimizes the sum of the squared constraint violations if all evaluated points are infeasible with respect to (P), otherwise as a feasible point minimizing the objective value. If $x_{\texttt{best}}$ violates at least one constraint of (P), BOA begins Phase I and then switches to Phase II as soon as a feasible solution is found and if some fixed maximum number $N_{\max}$ of function evaluations is not reached yet. Otherwise, if $x_{\texttt{best}}$ is feasible, Phase I is skipped and BOA directly switches to Phase II. The algorithm stops when the maximum evaluation budget is reached.

In Phase I, surrogate models of the objective and constraint functions of (P) are built iteratively, taking into account all the evaluations already done. These surrogates are used in optimization problem formulations – solved by an external solver – to identify a promising feasible point with respect to (P), namely a feasible point with low objective value. These formulations, explicitly given in Sect. 2.1.2, globally aim at minimizing the sum of squared surrogate constraint violations. Phase I together with the proposed management of the involved parameters are described in Algorithm 2. It takes namely (but not

only) account of the set of violated surrogate constraints to adjust parameters. Important key points, that rely on several parameters, are that i) the minimization of the surrogate objective value is already considered in Phase I but with a secondary importance compared to the surrogate constraints satisfaction, ii) each surrogate constraint is more or less relaxed, thanks to its own adaptive slack parameter, to take into account modelling errors, and iii) exploration of the variable space of (P) is managed by a minimum distance requirement to points that were already evaluated with the original objective and constraint functions.

Surrogates are also iteratively updated in Phase II to be used in optimization problem formulations. Phase II still involves parameters aiming at favouring feasible and diverse solutions, namely slack parameters for each surrogate constraint and a minimum distance requirement to already evaluated points. However, the optimization problems solved focus more on the decrease of the original objective value, compared to Phase I. As Phase II is entered only after a feasible solution of (P) is found, the best point $x_{\texttt{best}}$ is updated only if a new feasible point with lower objective value w.r.t. (P) is evaluated. The proposed management of Phase II is described in Algorithm 3.

Apart from the initial settings and parameters updates which are specific to each phase, both Phase I and Phase II make use of the following strategy which is detailed in Algorithm 4. Considering the predicted constraint values in Phase I and the real constraint values in Phase II, if the solution to some formulation violates a constraint, the slack corresponding to the surrogate constraint is increased to favour its satisfaction. On the contrary, if a constraint is satisfied by the solution of the formulation, after a certain amount $k_{\texttt{feas}} \in \mathbb{N}_+^*$ of successive satisfactions of this constraint, the corresponding slack is decreased to lay less stress on that constraint. We describe each phase in more details in Sect.2.1.2 and the different parameter updates in Sect.2.1.3.

2.1.2 The surrogate-based subproblems considered

In Phase I, a feasible solution is sought but, differently from what is usually done, we minimize the sum of squared surrogate constraint violations added with a fraction of the surrogate objective function. Similarly to the method COBRA, in order to raise the chance of producing truly feasible points, the constraints $\widehat{g}_j(x) + \epsilon_j \leq 0$ (with $\epsilon_j \geq 0$) are added and a minimum distance to evaluated points is enforced to favour exploration. Thus, the first phase aims at solving

$$\begin{array}{ll} \min\limits_{x \in \mathcal{X}} \sum_{j \in J} \max(0, \widehat{g}_j(x))^2 + \lambda \widehat{f}(x) & \\ s.t.\ \ \widehat{g}_j(x) + \epsilon_j \leq 0, & \forall j \in J, \\ \quad d_{\min} - \min\limits_{y \in \mathcal{P}} \|x - y\| \leq 0, & \end{array} \qquad (OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$$

where λ and $(\epsilon_j)_{j \in J}$ stand for some nonnegative scalar values, $d_{\min}$ is a positive value and $\mathcal{P}$ is the set of points that have been evaluated with the real blackbox functions f and $(g_j)_{j \in J}$. In our solution procedure, λ and $(\epsilon_j)_{j \in J}$ are iteratively

updated inside each iteration of Phase I so that the "leading" term of the objective in $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$ is the first one, that is the sum of squared surrogate constraint violations. By adding a fraction of the surrogate objective we aim at favouring the search of better feasible solutions, in terms of the original objective.

As long as the solution $\widehat{x}$ of $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$ calculated by Algorithm 2 at line 6 or line 12 is such that $\widehat{g}_j(\widehat{x}) > 0$, for some $j \in J$ and there is a considered reasonable decrease of the surrogate constraint violations, this subproblem is iteratively solved with updated λ and $(\epsilon_j)_{j\in J}$ (while loop at line 9 of Algorithm 2).

The surrogate-based subproblem is solved with an external algorithm and its computed solution $\widehat{x}$ is not guaranteed to be feasible for $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$, whether its feasible set is empty or not. If $\widehat{x}$ is not feasible for $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$, it is not evaluated and the problem is relaxed to find another solution to evaluate. The first relaxation (line 19 of Algorithm 2) consists in temporarily multiplying all $(\epsilon_j)_{j\in J}$ by -1. Indeed, the harder a constraint is and the bigger is the corresponding slack so, by taking the opposite, the corresponding surrogate constraint in $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$ becomes easier to satisfy. Thus, the first relaxation corresponds to

$$\begin{array}{lll} \min\limits_{x\in\mathcal{X}} \sum_{j\in J} \max(0,\widehat{g}_j(x))^2 + \lambda\widehat{f}(x) & & \\ s.t.\ \ \widehat{g}_j(x) - \epsilon_j \leq 0, & \forall j \in J, & (OBJ^{\lambda,-\epsilon}_{\texttt{PhaseI}}) \\ \quad d_{\min} - \min\limits_{y\in\mathcal{P}} \|x-y\| \leq 0. & & \end{array}$$

In case the newly calculated solution $\widehat{x}$ is not feasible for $(OBJ^{\lambda,-\epsilon}_{\texttt{PhaseI}})$, we focus on the distance criterion by keeping it as the only constraint (line 21 of Algorithm 2). However, the constraints satisfaction is still considered through the objective of the subproblem. Hence, the second relaxation can be written as follows

$$\begin{array}{l} \min\limits_{x\in\mathcal{X}} \sum_{j\in J} \max(0,\widehat{g}_j(x))^2 + \lambda\widehat{f}(x) \\ s.t.\ \ d_{\min} - \min\limits_{y\in\mathcal{P}} \|x-y\| \leq 0. \end{array} \qquad (OBJ^{\lambda})$$

If this last relaxation is considered, $\widehat{x}$ is updated as the calculated solution of (OBJ^{λ}). Therefore, at the end of an iteration of Phase I, the point $\widehat{x}$ is a feasible solution of $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$ if such a point is found. Otherwise, at least one relaxation problem is solved and $\widehat{x}$ is thus either a feasible solution of $(OBJ^{\lambda,-\epsilon}_{\texttt{PhaseI}})$ if one is found, or is a candidate solution of (OBJ^{λ}). The computed $\widehat{x}$ is evaluated with the blackbox functions, even if it is not feasible for $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$, $(OBJ^{\lambda,-\epsilon}_{\texttt{PhaseI}})$ or (OBJ^{λ}). This new evaluation is used to update the best point $x_{\texttt{best}}$ and the surrogates for the next iteration.

The minimum distance $d_{\min}$ is updated at the end of each iteration depending on the number of function evaluations already performed and according to the quality of the new evaluated point. Indeed, if at least 90% of the maximum number $N_{\max}$ of function evaluations has already been used, $d_{\min}$ is updated only if $x_{\texttt{best}}$ shows an improvement that is lower than 5% in terms of the sum

of squared constraint violations (using the original functions), compared to the previous best solution, in which case it is decreased to favour local search. Otherwise, that is if the remaining allowed number of function evaluations is greater than 10% of $N_{\max}$, then $d_{\min}$ is increased to favour exploration if the new best solution is considered as a good improvement compared to the previous one, and is decreased otherwise. The corresponding implementation of our strategy to update $d_{\min}$ (lines 26 to 34 of Algorithm 2) involves a discrete set of positive values $\Delta = \{d_1, \ldots, d_{|\Delta|}\}$ ordered in increasing order and is further described in Sect. 2.1.3. In any case, the value of $d_{\min}$ is limited by a lower bound γ which depends on the types of variables involved in (P).

The whole process is repeated until a feasible solution of the original problem (P) is found or the maximum evaluation budget is reached (lines 3 to 36 of Algorithm 2). In case no feasible solution of the original problem (P) is found within the maximum number of function evaluations $N_{\max}$, then a point in $\mathcal{P}$ minimizing the sum of squared constraint violations is returned. Otherwise, as soon as a feasible solution of (P) is found (i.e. the first time the condition on line 10 of Algorithm 1 given hereafter is verified), Phase II starts.

The goal of Phase II, described by Algorithm 3, is to improve the objective value of the best feasible solution. We aim at solving

$$\begin{array}{ll} \min\limits_{x\in\mathcal{X}} & \widehat{f}(x) \\ s.t. & \widehat{g}_j(x) + \epsilon_j \leq 0, \qquad \forall j \in J, \\ & d_{\min} - \min\limits_{y\in\mathcal{P}} \|x - y\| \leq 0. \end{array} \qquad (OBJ^{\epsilon}_{\texttt{PhaseII}})$$

Observe that the feasible region of this problem may be empty. Similarly to the first phase, $(OBJ^{\epsilon}_{\texttt{PhaseII}})$ is solved iteratively and relaxed in case no feasible solution is found. The first relaxation uses the opposite values of $(\epsilon_j)_{j\in J}$ and, thus, minimizes

$$\begin{array}{ll} \min\limits_{x\in\mathcal{X}} & \widehat{f}(x) \\ s.t. & \widehat{g}_j(x) - \epsilon_j \leq 0, \qquad \forall j \in J, \\ & d_{\min} - \min\limits_{y\in\mathcal{P}} \|x - y\| \leq 0. \end{array} \qquad (OBJ^{-\epsilon}_{\texttt{PhaseII}})$$

In case no feasible solution of $(OBJ^{-\epsilon}_{\texttt{PhaseII}})$ is found, only the distance constraint is kept. However, in order to take into account the constraints of the original problem, the sum of squared constraint violations is added to the objective. This second relaxation of Phase II solves (OBJ^{λ}) with λ equals 1: we denote this relaxation problem (OBJ^1). Similarly to what is done in Phase I, the calculated solution $\widehat{x}$ of the last subproblem solved is evaluated with the blackbox functions and is used to update the best point $x_{\texttt{best}}$, as well as the surrogate functions for the next iteration.

Unlike the first phase, $(\epsilon_j)_{j\in J}$ are updated only once per iteration and this is done with respect to the true constraint values (and not the surrogate predictions). The procedure to update $(\epsilon_j)_{j\in J}$ is presented in Algorithm 4 and detailed in Sect. 2.1.3.

Like in Phase I, the minimum distance $d_{\min}$ is updated at the end of each iteration of Phase II (lines 14 to 22 of Algorithm 3), but here using the improvement in the original objective function value as criterion instead of the sum of squared constraint violations. Its value is still limited by the lower bound γ.

The whole process of Phase II is repeated until the evaluation budget $N_{\max}$ is reached, in which case the algorithm returns $x_{\texttt{best}}$ which is, among the evaluated points, a feasible solution of (P) minimizing the objective function value.

Algorithm 1: `BOA`

1 Initialize phase number: $phase \leftarrow 1$
2 Initialize slacks: $\epsilon_{\max} > 0, \epsilon_j \leftarrow 0, \forall j \in J$.
3 Initialize slack factors: $\sigma_{\texttt{inc}} > 1$, $\sigma_{\texttt{dec}} \leftarrow \frac{1}{\sigma_{\texttt{inc}}}$, $\rho \in (0,1)$, $k_{\texttt{feas}} \in \mathbb{N}^*$.
4 Initialize distance parameters: $\gamma \geq 0$, $\Delta = \{d_1, d_2, \ldots, d_{|\Delta|}\} \subset \mathbb{R}_+^*$, $\nu \in \{1, 2, \ldots, |\Delta|\}$
5 $d_{\min} \leftarrow \max\{\gamma, d_\nu \cdot \min(u_i - l_i, \forall i \in C \cup I \cup D)\}$
6 Determine a set of points $\mathcal{P}_0 \subset \mathcal{X}$ ($|\mathcal{P}_0| \leq N_{\max}$), $\mathcal{P} \leftarrow \mathcal{P}_0$
7 Evaluate the functions f, $(g_j)_{j \in J}$ at each point in $\mathcal{P}_0$.
8 Initialize $x_{\texttt{best}}$.
9 Initialize $h_{\texttt{best}} \leftarrow \sum_{j \in J} \max(0, g_j(x_{\texttt{best}}))^2$.
10 **if** $\max_{j \in J} g_j(x_{\texttt{best}}) \leq 0$ **then**
11 $phase \leftarrow 2$ // $x_{\texttt{best}}$ is a feasible point in $\mathcal{P}$.
12 Update $x_{\texttt{best}}$ with Algorithm 3 // BOA-Phase II
13 **else**
14 Update $x_{\texttt{best}}$, $phase$ with Algorithm 2 // BOA-Phase I
15 **if** $phase = 2$ **then**
16 Update $x_{\texttt{best}}$ with Algorithm 3 // BOA-Phase II

Algorithm 2: `BOA-Phase I`

Given *phase*, $x_{\texttt{best}}$, $h_{\texttt{best}}$, $\mathcal{P}$, $N_{\max}$, Δ, ν, γ, $d_{\min}$, $\epsilon_{\max}$, $\sigma_{\texttt{inc}}$, $\sigma_{\texttt{dec}}$, ρ, $k_{\texttt{feas}}$, $\epsilon_j, \forall j \in J$
Initialize $\kappa_j \leftarrow 0, \forall j \in J$, *threshold* $\in (0,1)$, $\lambda_0 \in (0,1)$
while *phase* $= 1$ *and* $|\mathcal{P}| < N_{\max}$ **do**
  From $\mathcal{P}$, build surrogate functions $\widehat{f}, (\widehat{g}_j)_{j\in J}$.
  Initialize *dec_lambda* $\leftarrow$ `true`, $\lambda \leftarrow \lambda_0$.
  $\widehat{x} \leftarrow$ Solve $\left(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}}\right)$ // Compute a solution $\widehat{x}$ of $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$.
  Evaluate $\widehat{f}(\widehat{x})$, $(\widehat{g}_j(\widehat{x}))_{j\in J}$
  $\widehat{h} \leftarrow \sum_{j\in J} \max\left(0, \widehat{g}_j(\widehat{x})\right)^2$
  /* If $\widehat{x}$ not feasible w.r.t. the surrogate constraints $\widehat{g}_j(\widehat{x}) \leq 0$ for all $j \in J$, try to improve constraint satisfaction reducing λ and adjusting slacks $(\epsilon_j)_{j\in J}$. */
  while $\max_{j\in J} \widehat{g}_j(\widehat{x}) > 0$ *and dec_lambda is* `true` **do**
    $\lambda \leftarrow \frac{1}{2}\min\left(\lambda; \max_{j\in J} \widehat{g}_j(\widehat{x})\right)$
    Update $\epsilon_j, \kappa_j, \forall j \in J$ with Algorithm 4 and the predicted constraint values $(\widehat{g}_j(\widehat{x}))_{j\in J}$
    $\widehat{x} \leftarrow$ Solve $\left(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}}\right)$
    Evaluate $\widehat{f}(\widehat{x})$, $(\widehat{g}_j(\widehat{x}))_{j\in J}$
    $\widehat{h}_{\texttt{old}} \leftarrow \widehat{h}$
    $\widehat{h} \leftarrow \sum_{j\in J} \max\left(0, \widehat{g}_j(\widehat{x})\right)^2$
    if $\widehat{h}_{\texttt{old}} - \widehat{h} <$ *threshold* **then**
      dec_lambda $\leftarrow$ `false`.
  if $d_{\min} - \min_{y\in\mathcal{P}}(\|\widehat{x} - y\|) > 0$ *or* $\max_{j\in J}(\widehat{g}_j(x) + \epsilon_j) > 0$ **then**
    $\widehat{x} \leftarrow$ Solve $\left(OBJ^{\lambda,-\epsilon}_{\texttt{PhaseI}}\right)$ // Relax all ϵ_j in $-\epsilon_j$.
    if $d_{\min} - \min_{y\in\mathcal{P}}(\|\widehat{x} - y\|) > 0$ *or* $\max_{j\in J}(\widehat{g}_j(x) - \epsilon_j) > 0$ **then**
      $\widehat{x} \leftarrow$ Solve (OBJ^{λ})
  Evaluate $f(\widehat{x})$, $(g_j(\widehat{x}))_{j\in J}$
  $h_{\texttt{old}} \leftarrow h_{\texttt{best}}$
  Update $x_{\texttt{best}}$
  $h_{\texttt{best}} \leftarrow \sum_{j\in J} \max\left(0, g_j(x_{\texttt{best}})\right)^2$, $\mathcal{P} \leftarrow \mathcal{P} \cup \{\widehat{x}\}$
  if $|\mathcal{P}| \geq 0.9 \cdot N_{\max}$ **then**
    if $h_{\texttt{best}} > 0.95 \cdot h_{\texttt{old}}$ **then**
      $d_{\min} \leftarrow \max\left(\gamma, \frac{1}{2}\cdot\min(d_1, d_{\min})\right)$
  else
    if $h_{\texttt{best}} > 0.95 \cdot h_{\texttt{old}}$ **then**
      $\nu \leftarrow \max(1, \nu - 1)$
    else
      $\nu \leftarrow \min(|\Delta|, \nu + 1)$
    $d_{\min} \leftarrow \max\left(\gamma, d_\nu \cdot \min(u_i - l_i, \forall i \in C \cup I \cup D)\right)$
  if $\max_{j\in J} g_j(\widehat{x}) \leq 0$ **then**
    phase $\leftarrow 2$ // $\widehat{x}$ is feasible.

Algorithm 3: `BOA-Phase II`

Given $x_{\texttt{best}}$, $h_{\texttt{best}}$, $\mathcal{P}$, $N_{\max}$, Δ, ν, γ, $d_{\min}$, $\epsilon_{\max}$, $\sigma_{\texttt{inc}}$, $\sigma_{\texttt{dec}}$, ρ, $k_{\texttt{feas}}$, $\epsilon_j, \forall j \in J$
Initialize $\kappa_j \leftarrow 0, \forall j \in J$
while $|\mathcal{P}| < N_{\max}$ **do**
  $f_{\texttt{old}} \leftarrow f(x_{\texttt{best}})$
  From $\mathcal{P}$, build surrogate functions $\widehat{f}, (\widehat{g}_j)_{j \in J}$
  $\widehat{x} \leftarrow$ Solve $(OBJ^{\epsilon}_{\texttt{PhaseII}})$
  if $d_{\min} - \min_{y \in \mathcal{P}} (\|\widehat{x} - y\|) > 0$ *or* $\max_{j \in J} (\widehat{g}_j(x) + \epsilon_j) > 0$ **then**
    $\widehat{x} \leftarrow$ Solve $\left(OBJ^{-\epsilon}_{\texttt{PhaseII}}\right)$ `// Relax all` ϵ_j `in -`ϵ_j.
    if $d_{\min} - \min_{y \in \mathcal{P}} (\|\widehat{x} - y\|) > 0$ *or* $\max_{j \in J} (\widehat{g}_j(x) - \epsilon_j) > 0$ **then**
      $\widehat{x} \leftarrow$ Solve (OBJ^1) `// Solve` (OBJ^{λ}) `with` $\lambda = 1$.
  Evaluate $f(\widehat{x})$, $(g_j(\widehat{x}))_{j \in J}$
  Update $x_{\texttt{best}}$, $\mathcal{P} \leftarrow \mathcal{P} \cup \{\widehat{x}\}$
  Update $\epsilon_j, \kappa_j, \forall j \in J$ with Algorithm 4 and the real constraint values $(g_j(\widehat{x}))_{j \in J}$
  if $|\mathcal{P}| \geq 0.9 \cdot N_{\max}$ **then**
    if $f(x_{\texttt{best}}) > 0.95 \cdot f_{\texttt{old}}$ **then**
      $d_{\min} \leftarrow \max\left(\gamma, \frac{1}{2} \cdot \min(d_1, d_{\min})\right)$
  else
    if $f(x_{\texttt{best}}) > 0.95 \cdot f_{\texttt{old}}$ **then**
      $\nu \leftarrow \max(1, \nu - 1)$
    else
      $\nu \leftarrow \min(|\Delta|, \nu + 1)$
    $d_{\min} \leftarrow \max\left(\gamma, d_{\nu} \cdot \min(u_i - l_i, \forall i \in C \cup I \cup D)\right)$

Algorithm 4: `BOA-Update` ϵ, κ

Given $\sigma_{\texttt{inc}}$, $\sigma_{\texttt{dec}}$, ρ, $k_{\texttt{feas}}$, $\epsilon_{\max}$, $\epsilon_j, \kappa_j, \forall j \in J$
Given $\tilde{g}_j(\widehat{x}), \forall j \in J$ `// Predicted or real constraint values at` $\widehat{x}$.
for *each* $j \in J$ **do**
  if $\tilde{g}_j(\widehat{x}) > 0$ **then**
    $\kappa_j \leftarrow 0$ `// Reset feasibility counter.`
    $\tilde{g}_{\texttt{trunc}} \leftarrow \min(1, \tilde{g}_j(\widehat{x}))$
    $\epsilon_j \leftarrow \min\left(\max\left(\sigma_{\texttt{inc}} \cdot \epsilon_j; \rho \cdot \tilde{g}_{\texttt{trunc}}\right), \epsilon_{\max}\right)$ `// Increase` ϵ_j.
  else
    $\kappa_j \leftarrow \kappa_j + 1$
    if $\kappa_j \geq k_{\texttt{feas}}$ **then**
      $\epsilon_j \leftarrow \sigma_{\texttt{dec}} \cdot \epsilon_j$ `// Decrease` ϵ_j.
Return $\epsilon_j, \kappa_j, \forall j \in J$

2.1.3 Parameters update

The parameter λ involved in Phase I is reinitialized after each update of the surrogate models of the objective and constraint functions (line 5 of Algo-

rithm 2). It is then iteratively decreased (line 10 of Algorithm 2) as long as the solution $\widehat{x}$ of $(OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}})$ is not feasible w.r.t. the surrogate constraints (i.e. $\widehat{g}_j(\widehat{x}) > 0$ for some $j \in J$) and the sum of squared constraint violations is sufficiently reduced w.r.t a fixed *threshold* (Algorithm 2, lines 9-17). The idea is to give more importance to the objective of finding a feasible solution, provided that decreasing λ effectively contributes to favour constraints satisfaction.

The slacks $(\epsilon_j)_{j\in J}$ are computed according to the predicted (in Phase I) or real (in Phase II) constraint values. After $k_{\texttt{feas}} \in \mathbb{N}^*_+$ successive satisfactions of the constraint $j \in J$, the corresponding slack is decreased whereas it is increased as soon as the constraint is not satisfied, with respect to the real or predicted values depending on the phase. By waiting before decreasing a slack, we want to make sure that the satisfaction of the corresponding constraint is "robust/reliable". The increase of ϵ_j is done taking namely account of the amount of violation of constraint j and an upper bound $\epsilon_{\max}$ (line 7 of Algorithm 4). In addition to $k_{\texttt{feas}}$, three other fixed input parameters ($\sigma_{\texttt{inc}}, \sigma_{\texttt{dec}}$ and ρ) are used to modulate the changes of the slack values. The procedure is detailed in Algorithm 4. The data used to build the surrogate models is scaled, so that all the input and output values are upper bounded by one in absolute value. This however does not preclude the constructed surrogate functions from returning values out of this range. This motivates the truncation made on line 6 of Algorithm 4 to avoid sudden important increases of slacks in such situations.

The minimum distance parameter $d_{\min}$ is updated after every iteration in both phases according to the quality of the new evaluated point. It is set according to the minimum edge length of $\mathcal{X}$, an ordered finite set of positive values $\Delta = \{d_1, d_2, \ldots, d_{|\Delta|}\}$ and a lower bound γ. The latter depends on the nature of the variables: it is equal to 0 when there is at least one continuous variable and, otherwise, to the positive minimum gap between distinct admissible discrete or integer values. Let $\nu \in \{1, 2, \ldots, |\Delta|\}$ be the index of the chosen value from Δ, $d_{\min}$ is initialized as $\max\{\gamma, d_\nu \cdot \min(u_i - l_i, \forall i \in C \cup I \cup D)\}$. The minimum distance is updated at each iteration: it can be increased to enforce exploration after a considered good improvement in feasibility, and decreased otherwise to enable local exploitation. This is simply done by increasing or decreasing ν. In order to refine the solution of (P), lower values of $d_{\min}$ are allowed after a ratio of the evaluation budget (see lines 26 to 34 of Algorithm 2 and lines 14 to 22 of Algorithm 3).

The best point $x_{\texttt{best}}$ (line 8 of Algorithm 1, line 24 of Algorithm 2 and line 12 of Algorithm 3) is defined in $\mathcal{P}$ as the one (or one of those) minimizing the sum of the squared constraint violations if all points are infeasible, otherwise among the feasible points in $\mathcal{P}$, it is the one (or one of those) minimizing the objective value.

2.2 **Distinctive features of BOA**

BOA shares similarities with several surrogate-based BBO solvers like COBRA. The latter also performs a two-phase optimization where the first part aims at finding a feasible candidate. However both methods differ in many aspects. We point out some of them hereafter.

COBRA is based on RBF interpolations and reported results in [61] only make use of this surrogate, unlike the proposed method for which we report experiments with different types of surrogates. Besides, the distance parameter is adapted in BOA according to the quality of the solution found at each iteration. The resolutions of the subproblems are also different. COBRA handles continuous variables only and uses the MATLAB function `fmincon` that employs a SQP method. BOA is designed for problems involving discrete variables and our implementation uses the direct search solver NOMAD. Furthermore, the best iterate $x_{\texttt{best}}$ of the first phase of BOA is chosen directly based on the sum of squared constraint violations instead of the number of violated constraints or the maximum amount of one constraint violation. As other distinctive features, BOA updates its slacks already in Phase I, they may not be equal for all constraints and they are decreased as soon as the corresponding (predicted or original) constraints are not satisfied.

Differently from other methods such as SO-MI or CONDOR, the proposed method does not require an initial feasible solution. We solve an auxiliary problem to determine a candidate point whereas CONDOR proceeds to different types of perturbations of the currently best solution, taking into account integrality constraints. At each iteration, SO-MI evaluates 4 candidates that are chosen from 4 groups. Each group is generated by random perturbations of the variables and uniform random points generations.

2.3 **Adaptation of BOA to parallel evaluations**

The BOA algorithm described above conducts sequential evaluations of the blackbox. However, when the expensive optimization problem and the calculation resources enable parallel evaluations, it may be interesting to adapt the method by taking advantage of the parallelization and, hopefully, considerably reduce the total computational time. As an example, typical size optimization problems encountered at Stellantis are solved by proceeding to an order of 25 parallel evaluations of the finite element models. With this in mind, an extension of BOA to deal with parallel evaluations was designed.

Let $B \in \mathbb{N}^*$ stand for the number of allowed parallel calls to the blackbox. Apart from the initial DOE, the points evaluated in BOA come from the resolution of a surrogate subproblem. The idea now is to solve a batch of B subproblems where originally only one was solved in BOA, which leads to B points $\{\hat{x}^{(1)}, \ldots, \hat{x}^{(B)}\}$ to evaluate with the expensive functions at each iteration. The subproblems can be solved in parallel or sequentially as they only involve surrogate calls which are assumed computationally negligible com-

pared to the real blackbox evaluations. Thus, lines 6 to 21 of Algorithm 2 and lines 6 to 10 of Algorithm 3 are executed B times per iteration of the respective algorithm.

To do this, for each resolution b of a batch, a slack ϵ_{bj} is declined for each constraint, with an associated decrease counter κ_{bj}, for all $(b, j) \in \{1, 2, \ldots, B\} \times \{1, 2, \ldots, |J|\}$, as well as there is a parameter λ_b for each subproblem of a batch in Phase I. The second phase of BOA starts as soon as one feasible point with respect to the real blackbox functions is found.

All candidates examined by the subproblem solver during an iteration are stored and sorted according to a lexicographic order of:

$$\left(\sum_{j \in J} \max\left(0, \widehat{g}_j(x)\right)^2, \widehat{f}(x) \right).$$

With this sorting, feasible points are preferred to infeasible ones, and lower objective values and constraint violation of the feasible and infeasible points, respectively, are favoured. Let $\hat{\mathcal{P}}_b$ denote the points considered during the b^{th} resolution of a batch of subproblems, with $b \in \{1, 2, \ldots, B\}$, and let $V = \cup_{b=1}^{B} \hat{\mathcal{P}}_b$ be the union of the B sets of candidate points. Let us assume that V is sorted according to the above lexicographic order. The points $\{\hat{x}^{(1)}, \ldots, \hat{x}^{(B)}\}$ to evaluate with the expensive functions are iteratively selected from V such that the distance to $\mathcal{P}$ and to the already selected points is greater than $d_{\min}$. If needed, this distance is halved until we actually get B points for the parallel blackbox evaluations. This enables to not consider duplicate or very close points, and to favour exploration.

When a starting point is needed by an algorithm to solve a subproblem, as it is the case for MADS for instance, B points are chosen from the set of already evaluated points $\mathcal{P}$, considering two measures. The first one is defined by the following application:

$$\varphi : x \mapsto \begin{cases} f(x) & \text{if } g_j(x) \leq 0, \forall j \in J \\ f_{\max} + \sum_{j \in J} \max\left(0, g_j(x)\right)^2 & \text{if } \exists\ j \in \{1, 2, \ldots, |J|\}, g_j(x) > 0, \end{cases}$$

where $f_{\max}$ is the highest feasible objective value so far. The second criterion is the distance to the other evaluated points, that is to be maximized. Hence, the points of $\mathcal{P}$ are ranked in non-dominated sets according to the measures $(\varphi(x), -\min_{y \in \mathcal{P} \setminus \{x\}}(\|x - y\|))$, similarly to the non-dominated sorting of NSGA-II [17]: the sorting is iteratively done, removing the already chosen points, until B points are selected.

The respect of the evaluation budget is controlled before the parallel evaluations: if the remaining allowed number of blackbox calls is lower than the batch size B, then the latter is updated to $(N_{\max} - |\mathcal{P}|)$ and only this number of points is chosen for the expensive evaluations.

2.4 **Description of the components of BOA**

2.4.1 Initial DOE

As is very often the case in surrogate based BBO methods present in the literature, the DOE method we use for our experiments is a symmetric Latin Hypercube Design (SLHD) [82]. It is a variant of the space filling Latin hypercube sampling developed by [51].

By default, the algorithm starts with a DOE consisting of $(n + 1)$ SLHD vectors. If the vectors are not linearly independent, random points are added to the DOE until it contains $(n + 1)$ independent vectors. The independence requirement is also used in MISO for the starting points generation as it guarantees a unique setting for the RBF parameters.

2.4.2 Considered surrogate models

In this study, we consider radial basis functions, kriging models and MARS models that have shown good performance in the literature.

RBF, polynomial, kriging and MARS models are investigated in [35] on 13 nonlinear mathematical problems where RBF exhibit an overall best performance regarding, among others, accuracy, robustness and efficiency. On a high-dimensional automotive benchmark problem, the authors of [60] conclude that the methods using RBF are performing better than other methods including a kriging-based NOMAD and a sequential quadratic programming algorithm. The work of [53] investigates the influence of the surrogate type and the sampling strategy used in the resolution of expensive blackbox optimization problems subject to box constraints. The study notably includes cubic RBF, Gaussian kriging models and MARS models. The results on 15 continuous problems show that the ensembles including the cubic RBF often outperform those that do not use this type of surrogate.

Gaussian processes are also commonly used. In [37], RBF, kriging and polynomial surrogates are used to model 18 literature functions. The results exhibit a globally higher accuracy, robustness and efficiency of the kriging models, except on low-order nonlinear functions when the size of the DOE is small.

Linear, splines, kriging, neural networks, a support vector machine regression and random forests models are compared in [75] on a corn cultivation application study. The quality of the metamodels is evaluated for N_2O prediction and for N leaching prediction according to different measures. Splines and kriging based methods have the best results with small and medium training datasets.

Radial basis functions [28] are interpolating radially symmetric functions depending only on the distance between the input and some fixed point. They are generally used with the Euclidean norm for function approximation in an optimization framework. The RBF approximation $s(x)$ of an objective value

$f(x)$ is built as a weighted sum of basis functions:

$$s(x) = \sum_{i=1}^{N} \omega_i \phi(\|x - c^{(i)}\|),$$

where $x \in \mathbb{R}^n$, $\phi(\|x - c^{(i)}\|)$ are the basis functions evaluations, $c^{(i)} \in \mathbb{R}^n$ is the i^{th} out of $N \in \mathbb{N}^*$ basis function centres and the weights $w_i \in \mathbb{R}$ can easily be estimated by interpolation or least squares. Below are some examples of fixed and parametric basis functions:

- linear $\phi : r \mapsto r$
- cubic $\phi : r \mapsto r^3$
- thin plate spline $\phi : r \mapsto r^2 ln(r)$
- Gaussian $\phi : r \mapsto e^{-\frac{r^2}{2\sigma^2}}$
- multi-quadric $\phi : r \mapsto (r^2 + \sigma^2)^{\frac{1}{2}}$
- inverse multi-quadric $\phi : r \mapsto (r^2 + \sigma^2)^{-\frac{1}{2}}$

where σ represents a real parameter value. Additional terms such as polynomials can be added to the RBF formulation to increase its flexibility. In this study, we focus on cubic basis functions added with a polynomial tail.

Kriging models are well-known interpolation surrogates developed by [39] and that reduce the mean squared error of the approximation of a function. The variant of [69] lies on Gaussian processes based on prior covariances for the approximations.

Let $x \in \mathbb{R}^n$ be a point where the expensive function f is not known, the value of $f(x)$ is considered as the realization of a normally distributed random variable $Y(x) = \mu(x) + Z(x)$, where $\mu(x)$ is the deterministic mean and Z is a Gaussian process of mean 0. In case the mean is a constant value, if it is known the kriging is qualified as simple, otherwise it is ordinary if the constant value is unknown. In the latter case, $\mu(x)$ is often assigned the mean value of the approximations. Finally, if $\mu(x)$ is a linear combination of basis regression functions of x, the kriging is said to be universal. The covariance of Z at two points x and $\tilde{x}$ can be simply expressed as:

$$Cov(Z(x), Z(\tilde{x})) = \sigma^2 R(\theta, \|x - \tilde{x}\|),$$

where σ^2 is the process variance and R is a correlation function with parameter θ.

There exists different families of correlation functions and below are those used in this paper:

- Gaussian $R : \tau \mapsto e^{-\frac{\tau^2}{2\theta^2}}$
- exponential $R : \tau \mapsto e^{-\frac{|\tau|}{\theta}}$
- Matérn 3/2 $R : \tau \mapsto (1 + \frac{\sqrt{3}|\tau|}{\theta})e^{-\frac{\sqrt{3}|\tau|}{\theta}}$

– Matérn 5/2 $R : \tau \mapsto (1 + \frac{\sqrt{5}|\tau|}{\theta} + \frac{\sqrt{5}\tau^2}{3\theta^2})e^{-\frac{\sqrt{5}|\tau|}{\theta}}$,

where θ is the parameter of the correlation models.

The parameters μ, σ^2 and θ are calculated in order to maximize the likelihood function of the observed data, and often the log of the likelihood function. Kriging can be used for an efficient global optimization with use of a merit function such as the expected improvement (EI) and an enrichment strategy. Indeed, once the model is built from the DOE, the maximization of the EI gives promising points to evaluate and these points can be used to enrich the kriging model. Maximizing the EI actually simultaneously reduces the standard deviation (representing the uncertainty of the model) and the objective function.

Multivariate Adaptive Regression Spline is a surrogate model introduced in [21] that makes use of piecewise linear regression models to capture the non-linearities of a function. The search space is split into disjoint subregions, defined by points that are called knots, where regression models are built. A MARS model has the following form:

$$s(x) = \alpha_0 + \sum_{m=1}^{M} \alpha_m \mathcal{B}_m(x),$$

where the variable x is in $\mathbb{R}^n$ with n the dimension of the variable space, $\{\mathcal{B}_m\}_{m=1}^{M}$ are $M \in \mathbb{N}^*$ maximum linearly independent interaction basis functions, α_0 is the intercept coefficient that is the mean of the responses and α_m is the coefficient associated to the m^{th} basis function. The basis functions are the product of at least two truncated linear functions, namely functions that are linear on a bounded domain, defined between two knots, and equal to zero elsewhere. These truncated functions or hinge functions are univariate basis functions.

Let x_i denote the unidimensional variable corresponding to a hinge basis function $\mathcal{H}$, where $i \in \{1, \ldots, n\}$, $\mathcal{H}$ has the form:

$$\mathcal{H}(x_i) = \max(0, x_i - k) \qquad \text{or} \qquad \mathcal{H}(x_i) = \max(0, k - x_i),$$

with k its respective knot. The m^{th} interaction basis function is defined as the product of $L_m \in \mathbb{N}^*$ truncated linear functions:

$$\mathcal{B}_m(x) = \prod_{l=1}^{L_m} \max(0, s_{ml}(x_{i(m,l)} - k_{ml})),$$

where $s_{ml} = \pm 1$, $x_{i(m,l)}$ is the i^{th} variable on which the l^{th} hinge function of the m^{th} basis function depends and k_{ml} is the knot corresponding to $x_{i(m,l)}$.

2.4.3 Constraint handling

Slack factors are used in BOA to manage constraint satisfaction. The motivation behind this comes from [61] where conducted preliminary experiments exhibited solutions of the subproblems at the boundaries, where the surrogates are not accurate. As a result, in many cases these points were infeasible with respect to the true blackbox constraints. Although the subproblem formulation of the second phase is common, BOA considers the objective function already in the first phase through its parameter λ and introduces the slacks $(\epsilon_j)_{j\in J}$ for each constraint also in this phase. The update of the slacks also differs as they are updated independently for each constraint in BOA. In this way, a distinction is made between the constraints that are often or easily satisfied and the ones that are often violated. Hence, better solutions are expected by avoiding the pitfall of handling an important slack due to few constraints that are often violated and of staying far away from the border of easily satisfied constraints. Besides, the increase of an ϵ_j occurs as soon as the corresponding surrogate constraint is violated.

2.4.4 Blackbox crash handling

In real-world application problems, a blackbox may not give outputs for all inputs. As an example, the simulation of a finite element model may crash due to divergence in solving the underlying differential equations. This often results in a "NaN" output. In order to deal with this kind of hidden constraints, the non-real outputs are set to infinity. This way, the corresponding points are added to the set of already evaluated points to avoid duplicate evaluations and points in their close vicinity, i.e. in the ball of radius $d_{\min}$, are not considered. However, these candidates are not used in the construction of the models in order to not affect the model by making assumptions on their neighbourhoods.

2.4.5 Discrete variables handling

Inside NOMAD, discrete variables are treated as integers. Indeed, although integer and granular variables have a specific handling in NOMAD, there is no direct way to treat general discrete variables in the solver. However, the original variables are considered for each call of a surrogate or a blackbox. Hence, the true distances are considered to build the models.

3 Considered optimization problems

For the experiments, we consider instances derived from the literature and two automotive applications from Stellantis. All of them have inequality constraints (between 1 and 91) and can be considered medium to high dimensional in DFO as they have more than 10 variables. Note that with our terminology,

we distinguish mixed-integer problems that have continuous and integer variables from mixed-variable problems that have discrete variables other than integers.

3.1 **Instances from the literature**

The first set of instances consists of 19 constrained problems stemming from the literature, among which some are classical analytical problems and others are derived from applications. It comprises continuous, integer, mixed-integer and mixed-variable problems, whose names start with 'C', 'I', 'MI' and 'MV', respectively. Table 1 gives a description of these problems in terms of numbers and types of variables, and the source papers.

We use three instances from the well-known G-problems benchmark collection and derivations of two of them. Indeed, among the three integer problems taken from [55], the problems I1 and I3 are derived from the problem G01. The problem I2 is the hmittelman problem from the MINLPLib[1] library. A derivation of G07 called MV2 with mixed variables, including discrete ones, is used and its formulation comes from [15]. Applications about car side impact and stepped cantilever beam, respectively named MV3 and MV4 in our experiments, are also used.

Four problems that have mixed continuous and integer variables are taken from [54] and three of them derive from applications.

Among problems arisen from applications, six of them (C4, C5, C6, MI5, MI6 and MV1) come from a real-world benchmark suite introduced in [40].

3.2 **Applications from Stellantis**

In addition to the 19 problems from the literature, we use two mixed-variable problem instances encountered at Stellantis for some of the experiments. Table 2 gives descriptions of these problems.

3.2.1 Vehicle pole lateral crash

The first instance, called RSMLateralCrash, is a model of a pole lateral crash study, built from about 800 sample points. It is considered as representative of the expensive finite element model and was used at Stellantis in the optimization process for faster experiments. Each call to the model takes less than a minute. The abbreviation RSM stands for response surface model. The optimization problem considered aims at minimizing the mass of a basket of parts in the battery area of the vehicle. It is a constrained mixed-variable problem with 24 inequality constraints that represent performance features to satisfy, among which deceleration, stress and displacement. There are 34 variables which correspond to the choice of the materials of 10 parts of the

[1] `https://www.minlplib.org/index.html`

Table 1 Problems from the literature described with the dimension n, the number of continuous $|C|$, integer $|I|$ and discrete $|D|$ variables, respectively, and the number of inequality constraints $|J|$.

problem	n	$\|C\|$	$\|I\|$	$\|D\|$	$\|J\|$
C1 [19]	13	13	0	0	9
C2 [30]	10	10	0	0	8
C3 [29]	15	15	0	0	1
C4 [58,57,40]	14	14	0	0	15
C5 [24,40]	10	10	0	0	3
C6 [79,40]	30	30	0	0	91
I1 [19,55]	13	0	13	0	9
I2 [13,55]	16	0	16	0	7
I3 [19,55]	13	0	13	0	9
MI1 [9,54]	11	7	4	0	7
MI2 [84,54]	11	7	4	0	13
MI3 [41,54]	10	5	5	0	3
MI4 [41,54]	10	5	5	0	3
MI5 [25,40]	10	7	3	0	10
MI6 [27,40]	10	9	1	0	9
MV1 [83,40]	22	0	8	14	86
MV2 [30,15]	10	2	2	6	8
MV3 [26,22]	11	9	0	2	10
MV4 [73,22]	10	4	2	4	11

Table 2 Application problems from Stellantis described with the dimension n, the number of continuous $|C|$, integer $|I|$ and discrete $|D|$ variables, respectively, and the number of inequality constraints $|J|$.

problem	n	$\|C\|$	$\|I\|$	$\|D\|$	$\|J\|$
RSMLateralCrash	34	0	10	24	24
LateralCrash	48	0	3	45	64

vehicle, treated as integers according to some ranking based on the material properties, and the thicknesses of 24 parts, that are granular variables. Details on the variable bounds and granularities are given in Table 3. For instance, x_{11} allows values between 1 and 4 with a granularity of 0.05, so its admissible values belong to $\{1, 1.05, 1.1, \ldots, 4\}$.

Table 3 Variable bounds and granularities for RSMLateralCrash.

variable	lower bound	granularity	upper bound
1 to 10	1	1	10
11 to 18	1	0.05	4
19 to 34	1	0.05	2.5

3.2.2 Vehicle barrier lateral crash

The second problem, denoted LateralCrash, uses an expensive finite element simulation of a barrier lateral crash that takes more than 12 hours at each

call. The optimization problem is similar to the first one, minimizing the mass of a bench of parts of the vehicle, and has 64 inequality constraints. There are 48 variables, among which 3 materials and 45 granular thicknesses. Table 4 gives details on the bounds and granularity of each variable. The 48^{th} variable allows negative values because it intervenes in a formula for the thickness computation of the corresponding part.

Table 4 Variable bounds and granularities for LateralCrash.

variable	lower bound	granularity	upper bound
1	1	1	2
2 to 3	1	1	3
4 to 5	1.2	0.1	2
6 to 9	2	0.1	6.5
10	0.6	0.05	1.1
11	0.6	0.05	1.1
12 to 16	0.6	0.05	1.15
17 to 18	0.6	0.05	1.2
19	0.6	0.05	1.3
20 to 21	0.6	0.05	1.35
22	0.6	0.05	1.4
23 to 24	0.6	0.05	1.6
25	0.65	0.05	1.65
26 to 27	0.75	0.05	1.75
28 to 37	0.8	0.05	1.8
38	0.85	0.05	1.35
39	0.95	0.05	1.45
40	1.1	0.05	2.1
41	1.3	0.05	1.8
42	1.4	0.05	2.4
43 to 44	1.5	0.05	2.5
45	1.6	0.05	2.6
46	1.9	0.05	2.9
47	2	0.05	3
48	-0.1	0.05	0.1

4 Computational experiments

The surrogate models used in BOA are built on variables and objective values that are scaled in $[0, 1]$. The constraint values are scaled in $[-1, 0]$ for violated constraints and in $[0, 1]$ for satisfied ones. After every blackbox evaluation, the surrogate models are updated and a rescaling is performed. Similarly as performed in SO-MI, truncations are applied to the output values in order to avoid high variations of the surrogate values. Once the number of evaluated points is greater than twice the initial DOE size, the positive and negative constraint values are truncated to the median of the positive and the median of the negative constraint values respectively. Unlike what is done in SO-

MI, only the feasible objective values are truncated to their median once the number of feasible points evaluated is greater than twice the initial DOE size.

In order to lead numerical experiments, some parameters of BOA had to be set, in particular $\epsilon_{\max}, \sigma_{\texttt{inc}}, \rho, \lambda_0, k_{\texttt{feas}}, threshold$ and $d_{\min}$. To do this, a sensitivity analysis based on an experimental design of 27 points was performed. The latter consisted of 16 points from a Plackett-Burman design [59], added with 10 points from a space-filling algorithm and one point which corresponds to the middle of the considered bounds of the parameters. The Plackett-Burman design is used to investigate the main effects of the most important parameters as it detects linear correlations. It only uses the boundary values defined for the parameters to analyse. In practice, we initialize the parameters as follows: $\epsilon_{\max} = 10^{-3}$, $\sigma_{\texttt{inc}} = 1.1$, $\rho = 0.5$, $\lambda_0 = 0.5$ and $k_{\texttt{feas}} = \max(\lceil 2 \cdot \sqrt{n} \rceil, \lceil 2 \cdot \sqrt{|J|} \rceil)$, $threshold = 10^{-1}$. For the distance parameter setting, we choose $\Delta = \{5 \cdot 10^{-4}, 10^{-3}, 5 \cdot 10^{-3}, 10^{-2}, 5 \cdot 10^{-2}, 10^{-1}\}$ and $d_{\min}$ is initialized using $\nu = 3$, meaning $d_\nu = 5 \cdot 10^{-3}$. The lower bound of the minimum distance, γ, is fixed during the optimization and its value is set according to the problem, as described in Sect. 2.1.3.

Our implementation of BOA uses MATLAB R2020b and the subproblems of the algorithm are solved using the MATLAB version of NOMAD v3.9.1 with the option `ORTHO N+1 NEG`. The latter showed good performance in [16] compared with the other direction types of OrthoMads [2] both on continuous and mixed-integer optimization problems.

Two sets of experiments are considered with different evaluation budgets. The first one aims, in the one hand, at comparing different types of surrogate models and, on the other, at investigating the contribution of the parameter λ in Phase I. The second set is given a higher budget and compares BOA with NOMAD on the problems from the literature and on the instances from Stellantis.

In the presentation of the results, we denote the types of surrogates cubic RBF, MARS, and the four kinds of kriging (Gaussian, exponential, Matérn 3/2 and Matérn 5/2) as follows: R, M, KG, KE, K3 and K5, respectively.

The construction of the RBF surrogates borrows from the dedicated part of the code of MISO that we adapted to our algorithm. MARS models are constructed using the ARESLab[2] MATLAB toolbox [33]. Finally, Gaussian and exponential kriging models are built thanks to the DACE[3] MATLAB toolbox [47]. We implemented the correlation functions Matérn 3/2 and Matérn 5/2 embedded in the DACE framework. When applicable, the parameterization of the surrogates stems from preliminary experiments. Besides, we use ordinary kriging models, that is the deterministic mean is assumed to be an unknown constant value and is therefore estimated.

[2] `http://www.cs.rtu.lv/jekabsons/regression.html`

[3] `https://www.omicron.dk/dace.html`

4.1 **Medium-budget experiments**

In the first experiments, the performances of several implementations of BOA are evaluated on 30 runs performed, starting from different DOEs that are although common to all implementations, and with a blackbox evaluation budget of 200. Each subproblem resolution inside BOA is done by NOMAD with a maximum of $25 \cdot n$ surrogate function evaluations. The performance is evaluated according to different measures starting with the number of runs out of the 30 launched that ended up with a feasible solution. The other measures used consider only the feasible runs and are the mean number of evaluations used to leave Phase I, the mean objective values after Phase I and Phase II, respectively, and the minimum feasible objective value found over all runs. Comparisons are done using the 19 problems from the literature.

4.1.1 Comparisons of surrogate models

Firstly, cubic RBF, MARS and the 4 types of kriging (Gaussian, exponential, Matérn 3/2 and Matérn 5/2) are compared inside BOA. In the names of the columns, "Pb" stands for the problem name and $\mathcal{S}$ for the type of surrogate model. Tables 5, 6, 7 and 8 summarize the results for each family of problems, respectively on continuous, integer, mixed-integer and mixed-variable problems.

We first consider problems MV1 and C6 as there are no feasible results for all surrogates on these instances. Problem MV1, whose ratio of the feasible region is less than 10^{-4}, is considered difficult to solve by [40]. Indeed, the results show that kriging models hardly find feasible solutions on this problem, only the one using Matérn 3/2 manages to find one. MARS has the best results on MV1 regarding all the considered performance measures. On C6, MARS was stopped due to long computational times. Moreover, globally longer computational times seem to be needed for the construction of MARS models. While kriging models performed badly on MV1, they have the best mean objective values on C6 and, especially, the use of Matérn 3/2 gives the best mean objective values at the end of the optimization. Cubic RBF also exhibit good results on this problem and reaches the minimum feasible objective value among the surrogates.

On the other 17 problems, MARS followed by cubic RBF finds the highest number of feasible runs. These surrogates also globally reach the best qualities in solutions. It can be noted that MARS finds the global optimum on the 30 runs on I1, and so does cubic RBF on I2. On the contrary, considering each family of problems, exponential kriging reaches the lowest numbers of feasible runs. The difference with the other kriging types is especially noticeable on the integer problems I1 and I3 where it finds only 4 and 3 feasible runs respectively, whereas most runs are feasible for the others. Moreover, this kind of kriging globally finds the worst mean feasible objective values on mixed-integer and mixed-variable problems.

Table 5 Results on continuous problems for each surrogate type: number of feasible runs $\#F$, mean number of function evaluations to reach a feasible solution $N^{(I)}$, its standard deviation $\sigma_{N^{(I)}}$, mean first feasible objective value $f^{(I)}$, mean best feasible objective value $f^{(II)}$, its standard deviation $\sigma_{f^{(II)}}$ and minimum feasible objective value $f^{(II)}_{\min}$ on the 30 runs. A star $^{(*)}$ indicates that the experiment was stopped due to long computational times.

Pb	$\mathcal{S}$	$\#F$	$N^{(I)}$	$\sigma_{N^{(I)}}$	$f^{(I)}$	$f^{(II)}$	$\sigma_{f^{(II)}}$	$f^{(II)}_{\min}$
C1	R	24	85.29	66.64	-9.05	-9.94	3.29	-14.81
	M	30	78.87	43.85	-10.13	-14.01	0.76	-14.96
	KG	30	55.27	34.63	-3.81	-11.93	3.24	-15.00
	KE	16	73.56	43.17	-3.99	-13.92	1.97	-15.00
	K3	30	47.13	27.13	-2.93	-14.86	0.33	-15.00
	K5	30	47.13	29.39	-1.79	-14.49	1.07	-15.00
C2	R	30	38.07	10.88	126.93	26.09	1.01	24.94
	M	30	44.37	14.61	211.78	33.44	5.10	28.37
	KG	30	32.87	7.18	239.72	26.80	1.22	25.02
	KE	30	39.37	23.22	515.70	59.44	32.64	30.27
	K3	30	31.47	6.37	344.95	27.31	1.38	25.42
	K5	30	29.37	6.07	296.91	27.07	1.29	25.06
C3	R	30	43.87	17.12	1482.38	175.64	107.12	61.32
	M	30	49.93	19.23	4936.40	316.14	289.76	65.75
	KG	30	68.27	19.60	4783.45	463.37	421.21	101.59
	KE	30	71.73	27.78	8983.71	2312.13	2140.65	384.51
	K3	30	56.73	13.06	4068.69	690.08	1310.29	144.48
	K5	30	60.67	15.41	4507.89	436.88	240.44	66.50
C4	R	20	140.95	34.23	100220.25	99202.03	422055.30	32.71
	M	30	111.10	28.55	9722.38	4091.61	3709.92	6.76
	KG	21	115.43	33.49	1271738.01	154861.04	258460.79	15.10
	KE	10	89.30	14.13	1686509.88	123967.71	206323.43	2275.04
	K3	29	119.59	34.41	1003787.88	166552.78	289010.56	92.54
	K5	25	126.12	35.38	1312627.27	612411.72	1916379.73	32.10
C5	R	30	29.07	7.28	684.60	552.03	5.78	537.53
	M	30	23.80	6.57	883.09	569.58	17.49	548.18
	KG	30	20.40	5.04	857.74	547.44	3.58	541.68
	KE	30	20.10	4.60	871.21	562.82	11.49	548.02
	K3	30	21.67	6.63	831.20	556.14	15.98	542.45
	K5	30	20.77	5.79	831.13	554.01	10.56	541.79
C6	R	30	57.63	8.24	-4467.56	-5292.86	220.87	-5674.37
	M$^{(*)}$	-	-	-	-	-	-	-
	KG	30	52.23	7.54	-4583.62	-5307.92	231.72	-5598.43
	KE	30	52.63	5.68	-4530.97	-5298.64	143.72	-5601.75
	K3	30	52.30	4.70	-4510.78	-5371.81	171.01	-5663.18
	K5	30	53.07	5.85	-4563.47	-5325.31	171.15	-5657.87

4.1.2 Investigations on λ

The parameter λ is used in the surrogate subproblem ($OBJ^{\lambda,\epsilon}_{\texttt{PhaseI}}$) to take into account the objective function in the first phase of BOA. The introduction of this parameter is a special feature of our algorithm by comparison with the other existing methods. The effects of λ are investigated by comparing the performance of the proposed method when the parameter is classically used and when it equals 0 during all the optimization. To do so, cubic RBF models are considered in BOA as this type of model outperformed most of the

Table 6 Results on integer problems for each surrogate type: number of feasible runs $\#F$, mean number of function evaluations to reach a feasible solution $N^{(I)}$, its standard deviation $\sigma_{N^{(I)}}$, mean first feasible objective value $f^{(I)}$, mean best feasible objective value $f^{(II)}$, its standard deviation $\sigma_{f^{(II)}}$ and minimum feasible objective value $f^{(II)}_{\min}$ on the 30 runs.

Pb	$\mathcal{S}$	$\#F$	$N^{(I)}$	$\sigma_{N^{(I)}}$	$f^{(I)}$	$f^{(II)}$	$\sigma_{f^{(II)}}$	$f^{(II)}_{\min}$
I1	R	30	32.20	15.58	-11.27	-14.17	1.29	-15.00
	M	30	37.00	14.20	-12.93	-15.00	0.00	-15.00
	KG	30	34.97	16.61	-4.73	-11.00	2.44	-15.00
	KE	4	114.00	73.87	-8.25	-10.75	1.50	-12.00
	K3	26	37.65	28.83	-5.35	-11.35	2.23	-15.00
	K5	26	32.50	15.30	-5.15	-11.00	2.40	-15.00
I2	R	30	34.20	4.84	20.67	13.00	0.00	13.00
	M	30	54.57	25.39	20.30	15.97	3.67	13.00
	KG	30	49.33	18.53	33.57	18.50	10.96	13.00
	KE	30	45.47	21.83	29.07	17.07	7.07	13.00
	K3	30	47.90	31.25	26.63	16.17	7.05	13.00
	K5	30	47.87	25.91	30.67	18.57	10.99	13.00
I3	R	30	39.40	28.96	-43507.57	-49971.70	109.17	-50128.00
	M	30	83.23	35.00	-44050.30	-50143.50	67.54	-50200.00
	KG	30	39.07	14.63	-36306.83	-50186.27	15.38	-50200.00
	KE	3	161.67	7.02	-49958.00	-50064.00	84.11	-50159.00
	K3	29	45.48	8.91	-38440.90	-50183.41	20.83	-50200.00
	K5	30	38.20	8.67	-37572.90	-50180.93	23.44	-50200.00

others in the previous experiments of Sect. 4.1.1 and it is computationally less expensive than MARS.

The same performance measures are used and presented in Table 9 for the 19 problems from the literature. In the surrogate column entitled $\mathcal{S}$, "$\mathrm{R}_{\lambda=0}$" indicates the cubic RBF-based BOA that does not use λ.

Considering all problems except MV1, both variants are globally equivalent regarding the number of feasible runs. The use of λ generally leads to more evaluations spent in Phase I but the objective values when exiting Phase I and Phase II are globally better. There can be a considerable difference in the quality of the solution as shown on problem C4: the mean objective value $f^{(II)}$ of the traditional BOA is almost 3 times better than the variant that does not use λ.

Nevertheless, on two of the three integer problems tested, although the objective values after Phase I are better when λ is used for the three of them, there are slight advantages at the end of the optimization regarding $f^{(II)}$ when $\lambda = 0$. The problems concerned are alterations of the same problem and only their bounds differ. This explains why the algorithm behaves similarly on them. The effect of λ on these instances may be specific to the problems. Besides, the number of evaluations spent in Phase I for I1 and I3 is higher in the traditional BOA and, therefore, there are less evaluations left in Phase II for improving the solutions. In addition, considering the minimum feasible objective values among all runs, the absence of λ leads to lower objective values on 10 of the problems.

Table 7 Results on mixed-integer problems for each surrogate type: number of feasible runs $\#F$, mean number of function evaluations to reach a feasible solution $N^{(I)}$, its standard deviation $\sigma_{N^{(I)}}$, mean first feasible objective value $f^{(I)}$, mean best feasible objective value $f^{(II)}$, its standard deviation $\sigma_{f^{(II)}}$ and minimum feasible objective value $f^{(II)}_{\min}$ on the 30 runs.

Pb	$\mathcal{S}$	$\#F$	$N^{(I)}$	$\sigma_{N^{(I)}}$	$f^{(I)}$	$f^{(II)}$	$\sigma_{f^{(II)}}$	$f^{(II)}_{\min}$
MI1	R	30	20.97	2.66	-0.19	-0.91	0.04	-0.94
	M	30	22.83	3.06	-0.31	-0.93	0.03	-0.94
	KG	30	29.10	12.29	-0.25	-0.92	0.05	-0.95
	KE	30	25.17	7.66	-0.33	-0.91	0.05	-0.94
	K3	30	27.27	9.05	-0.37	-0.92	0.03	-0.95
	K5	30	27.57	5.78	-0.33	-0.91	0.05	-0.95
MI2	R	30	19.90	4.37	14.81	5.90	0.31	4.73
	M	30	23.53	9.50	13.07	5.80	0.23	4.59
	KG	30	29.57	9.41	10.42	6.23	0.62	4.58
	KE	26	52.73	40.01	10.70	8.04	1.61	5.83
	K3	30	27.73	9.74	11.23	6.62	0.74	5.82
	K5	30	31.47	9.22	9.14	6.27	0.55	5.15
MI3	R	30	37.20	14.41	-0.95	-0.99968	0.00021	-0.99988
	M	30	38.63	14.09	-0.54	-0.99813	0.00188	-0.99975
	KG	30	51.03	35.79	-0.88	-0.99928	0.00133	-0.99985
	KE	23	49.35	32.23	-0.83	-0.99916	0.00151	-0.99980
	K3	27	32.89	20.27	-0.83	-0.99944	0.00051	-0.99988
	K5	24	30.08	13.43	-0.87	-0.99931	0.00146	-0.99984
MI4	R	30	41.10	10.16	-0.96	-0.99949	0.00034	-0.99998
	M	30	35.57	14.01	-0.72	-0.99928	0.00073	-0.99991
	KG	26	48.65	35.01	-0.69	-0.99734	0.01200	-0.99999
	KE	18	62.94	49.88	-0.81	-0.99592	0.01091	-0.99999
	K3	28	49.00	33.75	-0.75	-0.99812	0.00418	-0.99998
	K5	25	37.76	31.27	-0.70	-0.99889	0.00145	-0.99999
MI5	R	30	30.47	9.70	99190.66	61782.15	5174.17	58558.89
	M	30	21.93	8.19	149128.59	64093.16	4638.25	56576.26
	KG	30	35.07	12.10	112865.34	60907.19	3614.95	58652.64
	KE	30	31.47	8.84	108068.20	63041.05	7340.51	58710.03
	K3	30	29.80	9.66	100686.43	60248.08	3424.47	58632.38
	K5	30	28.17	7.94	112179.49	59898.48	2883.39	58597.03
MI6	R	30	22.83	3.71	23547.31	16989.35	29.34	16959.62
	M	30	22.13	7.77	24184.04	17094.23	674.18	16959.18
	KG	30	23.73	4.99	21710.40	16966.08	11.31	16958.33
	KE	30	25.07	6.48	24557.69	18279.88	2095.08	16958.69
	K3	30	23.10	5.09	25701.99	16972.69	19.42	16958.55
	K5	30	23.90	6.38	23463.35	16967.75	15.35	16958.31

On MV1, the variant that does not use the tested parameter was stopped because the experiment was computationally too long. The use of λ seems to help in the optimization of this hard problem.

The experiments show promising results regarding the utility of λ in the problem formulation of Phase I. It globally leads to better $f^{(II)}$ values.

Table 8 Results on mixed-variable problems for each surrogate type: number of feasible runs $\#F$, mean number of function evaluations to reach a feasible solution $N^{(I)}$, its standard deviation $\sigma_{N^{(I)}}$, mean first feasible objective value $f^{(I)}$, mean best feasible objective value $f^{(II)}$, its standard deviation $\sigma_{f^{(II)}}$ and minimum feasible objective value $f_{\min}^{(II)}$ on the 30 runs.

Pb	$\mathcal{S}$	$\#F$	$N^{(I)}$	$\sigma_{N^{(I)}}$	$f^{(I)}$	$f^{(II)}$	$\sigma_{f^{(II)}}$	$f_{\min}^{(II)}$
MV1	R	9	157.00	28.34	113.17	106.98	32.57	67.49
	M	16	142.63	25.71	73.69	73.23	19.87	52.96
	KG	0	-	-	-	-	-	-
	KE	0	-	-	-	-	-	-
	K3	1	186.00	0.00	73.10	67.28	0.00	67.28
	K5	0	-	-	-	-	-	-
MV2	R	30	31.97	7.84	210.93	33.06	4.89	31.43
	M	30	28.20	7.45	451.54	35.37	2.51	32.53
	KG	30	30.10	11.76	415.25	40.81	22.31	31.53
	KE	27	33.67	23.56	638.59	93.34	67.74	32.72
	K3	30	29.93	20.31	288.77	43.11	22.07	31.49
	K5	30	27.57	12.43	309.46	36.24	8.75	31.44
MV3	R	30	27.43	9.61	27.36	23.78	0.58	23.57
	M	30	32.53	10.33	28.43	23.73	0.16	23.59
	KG	30	28.57	11.07	29.67	23.79	0.30	23.58
	KE	30	25.30	6.44	31.31	24.41	0.71	23.58
	K3	30	25.70	5.75	30.33	23.89	0.41	23.58
	K5	30	23.73	5.45	30.56	23.90	0.45	23.57
MV4	R	30	19.17	2.68	79543.73	66962.58	1766.70	64430.08
	M	30	17.20	1.30	89018.46	65948.35	1407.97	64414.72
	KG	30	23.17	6.58	90982.70	67213.17	1819.30	64358.21
	KE	30	22.63	5.53	86514.34	68573.29	2655.00	64478.30
	K3	30	20.93	4.56	88296.62	67347.29	2218.25	64392.13
	K5	30	21.83	4.76	85685.56	67467.49	1753.36	64425.09

4.2 **Large-budget experiments**

BOA used with cubic RBF surrogates is compared with two surrogate-assisted variants of NOMAD. To do so, cubic RBF and Gaussian kriging models are given to NOMAD v3.9.1 as external surrogates and are updated after each blackbox evaluation. Each run of NOMAD is given as starting point the first point of the initial DOE used for BOA, that is the first point of the SLHD. The truncation to the median is applied to NOMAD for the surrogate construction: it is applied to the constraints after $2 \cdot (n+1)$ blackbox evaluations, and to the feasible objective values when the number of feasible points evaluated is greater than $2 \cdot (n+1)$.

Note that external surrogates are not employed in NOMAD when it is used inside BOA since the subproblems tackled are already based on models of the blackbox functions. In this study, we only perform comparisons with variants of NOMAD using external surrogates. This choice conforms with the study of [4] where four ordering strategies for the poll step of NOMAD are compared. The results show that the strategies using external surrogates to choose the points to be evaluated in the poll perform better (in terms of the proportion of problems solved) than the two others that use the default quadratic model only

Table 9 Results on all problems for RBF surrogate with and without λ: number of feasible runs $\#F$, mean number of function evaluations to reach a feasible solution $N^{(I)}$, its standard deviation $\sigma_{N^{(I)}}$, mean first feasible objective value $f^{(I)}$, mean best feasible objective value $f^{(II)}$, its standard deviation $\sigma_{f^{(II)}}$ and minimum feasible objective value $f^{(II)}_{\min}$ on the 30 runs. A star $^{(*)}$ indicates that the experiment was stopped due to long computational times.

Pb	$\mathcal{S}$	$\#F$	$N^{(I)}$	$\sigma_{N^{(I)}}$	$f^{(I)}$	$f^{(II)}$	$\sigma_{f^{(II)}}$	$f^{(II)}_{\min}$
C1	R	24	85.29	66.64	-9.05	-9.94	3.29	-14.81
	$R_{\lambda=0}$	30	21.50	1.80	-3.71	-8.87	4.08	-14.95
C2	R	30	38.07	10.88	126.93	26.09	1.01	24.94
	$R_{\lambda=0}$	30	38.40	9.57	1231.45	26.10	0.81	24.85
C3	R	30	43.87	17.12	1482.38	175.64	107.12	61.32
	$R_{\lambda=0}$	30	34.80	7.85	14444.98	198.63	130.02	57.50
C4	R	20	140.95	34.23	100220.25	99202.03	422055.30	32.71
	$R_{\lambda=0}$	19	121.95	47.99	949918.50	283360.89	658017.50	200.87
C5	R	30	29.07	7.28	684.60	552.03	5.78	537.53
	$R_{\lambda=0}$	30	17.73	1.20	897.62	552.66	5.22	542.38
C6	R	30	57.63	8.24	-4467.56	-5292.86	220.87	-5674.37
	$R_{\lambda=0}$	30	57.73	10.25	-4530.51	-5306.63	174.03	-5713.49
I1	R	30	32.20	15.58	-11.27	-14.17	1.29	-15.00
	$R_{\lambda=0}$	30	23.03	5.62	-3.63	-14.50	0.86	-15.00
I2	R	30	34.20	4.84	20.67	13.00	0.00	13.00
	$R_{\lambda=0}$	30	34.10	6.83	24.57	13.00	0.00	13.00
I3	R	30	39.40	28.96	-43507.57	-49971.70	109.17	-50128.00
	$R_{\lambda=0}$	30	35.20	18.25	-23134.57	-50018.80	120.62	-50199.00
MI1	R	30	20.97	2.66	-0.19	-0.91	0.04	-0.94
	$R_{\lambda=0}$	30	19.27	1.53	-0.10	-0.91	0.03	-0.95
MI2	R	30	19.90	4.37	14.81	5.90	0.31	4.73
	$R_{\lambda=0}$	30	22.90	5.00	14.45	6.03	0.44	4.76
MI3	R	30	37.20	14.41	-0.95	-0.99968	0.00021	-0.99988
	$R_{\lambda=0}$	30	22.17	4.78	-0.11	-0.99966	0.00022	-0.99987
MI4	R	30	41.10	10.16	-0.96	-0.99949	0.00034	-0.99998
	$R_{\lambda=0}$	30	21.97	4.91	-0.47	-0.99958	0.00046	-0.99996
MI5	R	30	30.47	9.70	99190.66	61782.15	5174.17	58558.89
	$R_{\lambda=0}$	30	19.23	2.27	166354.88	61734.22	5538.20	58545.80
MI6	R	30	22.83	3.71	23547.31	16989.35	29.34	16959.62
	$R_{\lambda=0}$	30	25.30	7.37	34421.18	16984.71	37.35	16959.44
MV1	R	9	157.00	28.34	113.17	106.98	32.57	67.49
	$R^{(*)}_{\lambda=0}$	-	-	-	-	-	-	-
MV2	R	30	31.97	7.84	210.93	33.06	4.89	31.43
	$R_{\lambda=0}$	30	33.20	9.28	1654.05	33.78	5.51	31.47
MV3	R	30	27.43	9.61	27.36	23.78	0.58	23.57
	$R_{\lambda=0}$	30	19.27	1.66	36.19	23.64	0.10	23.54
MV4	R	30	19.17	2.68	79543.73	66962.58	1766.70	64430.08
	$R_{\lambda=0}$	30	17.23	1.28	98682.65	66789.78	1800.28	64392.92

and increasing angle with the last direction of success, respectively. In an earlier study [72], different subproblem formulations are compared on 20 problems from the literature and 2 simulation-based multidisciplinary problems showing equivalent or better performance of NOMAD when using surrogates other than the default quadratic model in the search step. In particular, significant advantages were observed on non-smooth, noisy, and non-convex problems.

In the following experiments, the blackbox evaluation budget is set at 400 and each subproblem resolution of BOA uses a maximum of $100 \cdot n$ surrogate evaluations. We denote $\mathrm{B_R}$ the cubic RBF-based BOA and, respectively, $\mathrm{N_R}$ and $\mathrm{N_K}$ the NOMAD variants assisted with cubic RBF and Gaussian kriging.

4.2.1 Algorithm comparisons on benchmark problems

The first experiments with the larger evaluation budget are performed using the problems from the literature.

Table 10 presents the results on 18 of the problems as the two NOMAD variants did not find any feasible run on MV1 and BOA was stopped on this problem due to a long computational time (subsequent to the increase of the budget). The notation $\mathcal{A}$ stands for the type of algorithm used.

All algorithms find a similar number of feasible solutions on 4 of the 6 continuous problems and BOA performs better regarding the number of feasible runs on C4.

Note that on C1 the methods using cubic RBF models perform less than the kriging-based NOMAD regarding the number of feasible runs, suggesting that kriging better captures the complexity of this instance. Comparing the RBF-based methods on this problem, the results show a similar number of feasible runs and that BOA uses less blackbox evaluations to find the first feasible solution, moreover whose objective value is better on average , which is interesting for restricted evaluation budgets. The best solution from all runs on C1 was found by BOA.

On the integer problems, the number of feasible solutions found by BOA is greater or equal than those of NOMAD. Besides, BOA uses less evaluations to find a first feasible candidate solution, and the mean feasible objective values $f^{(I)}$ after Phase I are better. The standard deviations of the number of evaluations used in Phase I and that of the function values at the end of the optimization, $\sigma_{N^{(I)}}$ and $\sigma_{f^{(II)}}$, respectively, are also smaller for BOA, suggesting more robust solutions for this algorithm. In particular, on the problem I2, BOA finds the same final solution for each of the 30 runs and it corresponds to the global minimum of the problem.

Regarding the mixed-integer problems, while the number of feasible runs is equivalent for all methods, BOA finds, on average, better objective values $f^{(I)}$ at the end of Phase I on most of the problems.

Finally, BOA behaves better on the 3 mixed-variable problems considered. Indeed, it always finds a feasible solution and the latter has on average better objective values after Phase I and Phase II, respectively, as well as the minimum objective function value of all runs $f_{\min}^{(II)}$ is also smaller. There is, in particular, an important gap on the application problem MV4 on which all BOA runs are feasible against only 2 for NOMAD.

Thus, except on C1, BOA finds more feasible solutions on the continuous, integer and mixed-variable problems and is equivalent to both variants of NOMAD on the mixed-integer problems. Moreover, BOA is globally better on the mixed-variable problems considered.

Table 10 Results on 18 problems for BOA with RBF, NOMAD with RBF and NOMAD with KG: number of feasible runs $\#F$, mean number of function evaluations to reach a feasible solution $N^{(I)}$, its standard deviation $\sigma_{N^{(I)}}$, mean first feasible objective value $f^{(I)}$, mean best feasible objective value $f^{(II)}$, its standard deviation $\sigma_{f^{(II)}}$ and minimum feasible objective value $f_{\min}^{(II)}$ on the 30 runs.

Pb	$\mathcal{A}$	$\#F$	$N^{(I)}$	$\sigma_{N^{(I)}}$	$f^{(I)}$	$f^{(II)}$	$\sigma_{f^{(II)}}$	$f_{\min}^{(II)}$
C1	B_R	18	113.61	115.73	-9.89	-10.09	3.59	-14.99
	N_R	19	177.68	94.39	-4.27	-11.95	2.15	-14.06
	N_K	27	211.19	88.39	-3.69	-10.72	2.86	-14.53
C2	B_R	30	45.80	19.10	52.68	24.82	0.71	24.33
	N_R	30	177.77	63.63	926.49	101.51	103.95	30.72
	N_K	30	181.70	61.52	630.81	78.06	54.80	31.91
C3	B_R	30	61.70	40.51	1079.54	105.30	105.20	47.57
	N_R	30	139.03	69.43	6662.72	1748.93	2573.30	76.16
	N_K	29	146.28	70.25	7874.00	2526.50	3282.22	62.99
C4	B_R	25	186.00	71.53	5006.10	3877.70	4147.88	6.62
	N_R	7	295.43	73.70	1480.99	1031.72	2637.85	0.38
	N_K	10	245.70	108.96	42649.96	6679.82	16668.94	0.74
C5	B_R	30	50.20	29.37	613.29	530.77	3.75	525.62
	N_R	30	21.10	15.55	1040.14	619.79	50.90	557.09
	N_K	30	21.43	15.62	1042.35	609.03	38.15	550.50
C6	B_R	30	55.77	9.77	-4435.25	-5407.09	148.44	-5642.83
	N_R	30	56.83	51.49	-4587.86	-5425.54	124.31	-5770.14
	N_K	30	42.90	33.29	-4634.85	-5457.70	144.14	-5825.52
I1	B_R	30	27.97	9.28	-12.17	-14.80	0.61	-15.00
	N_R	22	220.36	110.92	-6.73	-14.18	1.74	-15.00
	N_K	23	208.09	108.30	-6.96	-14.04	1.58	-15.00
I2	B_R	30	36.93	10.94	19.90	13.00	0.00	13.00
	N_R	30	84.60	66.00	20.27	13.20	0.76	13.00
	N_K	30	75.93	57.12	20.80	13.20	0.76	13.00
I3	B_R	30	36.03	33.46	-44120.83	-50046.83	102.62	-50199.00
	N_R	23	198.96	91.48	-41986.39	-50067.00	253.23	-50200.00
	N_K	25	231.76	80.64	-42354.84	-49622.92	1992.40	-50200.00
MI1	B_R	30	22.47	4.75	-0.31	-0.90	0.05	-0.94
	N_R	30	5.87	6.01	-0.09	-0.83	0.06	-0.92
	N_K	30	5.63	5.02	-0.10	-0.85	0.06	-0.92
MI2	B_R	30	29.47	49.03	14.84	6.10	0.58	5.54
	N_R	30	54.93	24.36	14.01	6.49	1.02	4.64
	N_K	30	59.47	28.51	12.71	5.74	0.66	4.63
MI3	B_R	29	60.62	32.88	-0.98	-0.99970	0.00017	-0.99988
	N_R	30	18.57	16.78	-0.06	-0.99917	0.00063	-0.99983
	N_K	30	23.30	30.99	-0.11	-0.99914	0.00058	-0.99977
MI4	B_R	30	56.33	36.63	-0.93	-0.99917	0.00065	-0.99998
	N_R	30	12.70	0.53	-0.46	-0.99996	0.00004	-0.99999
	N_K	30	12.70	0.53	-0.46	-0.99996	0.00005	-0.99999
MI5	B_R	30	55.53	24.46	73176.52	62192.92	5320.47	58505.73
	N_R	30	92.50	53.86	133934.17	76819.02	10584.92	54844.21
	N_K	30	97.77	71.98	143147.70	75466.98	10945.25	56623.22
MI6	B_R	30	28.37	23.97	19714.63	16984.42	26.82	16958.23
	N_R	30	47.43	24.64	27051.55	17042.98	144.22	16959.28
	N_K	30	52.13	35.88	30589.19	17021.32	106.00	16962.30
MV2	B_R	30	29.90	7.90	142.64	32.02	2.29	31.42
	N_R	30	147.83	58.26	1035.47	85.38	64.41	35.16
	N_K	30	157.53	55.40	872.32	90.61	107.76	35.58
MV3	B_R	30	38.97	18.51	24.86	23.62	0.09	23.53
	N_R	30	28.30	27.12	32.62	24.56	0.96	23.60
	N_K	30	26.87	22.60	33.90	24.32	0.62	23.67
MV4	B_R	30	22.80	7.13	75047.08	66812.74	2040.08	64335.74
	N_R	2	62.00	2.83	97030.00	70671.61	1229.81	69802.00
	N_K	2	126.50	16.26	75033.07	66892.43	836.45	66300.98

4.2.2 Algorithm comparisons on RSMLateralCrash in parallel mode

The methods tested in Sect. 4.2.1 are used on the RSM-based lateral crash study from Stellantis.

In order to reduce the total computational time of the optimization, the parallel version of BOA is used. For the same reason, block evaluations are allowed in both variants of NOMAD. The maximum number of parallel blackbox evaluations is set at 25 in BOA and NOMAD, and two runs are performed for each method.

Note that BOA performs the maximum number of parallel blackbox evaluations allowed at each iteration, and possibly less only at the last evaluation to not exceed the evaluation budget. Differently, NOMAD does not use the whole parallelization capacity at every iteration and can exceed the evaluation budget at its last iteration. Possible extra evaluations are not considered in the analyses.

Figures 1 and 2 present the evolution of the best feasible objective values for each method during the first and second run, respectively. The results are also summarized in Table 11 for each run. The number of blackbox evaluations to reach a feasible solution ($N^{(I)}$) is computed as the number of expensive evaluations performed at the end of the batch to which the feasible point belongs to.

Comparing the parallel methods, during the first run, NOMAD uses less evaluations to find a feasible solution, as shown in the lines of "run 1" of Table 11. This is partly due to the fact that BOA does not start with a single point but with a DOE with points chosen infeasible. Nonetheless, a gap in the objective values is observed on Fig. 1 when BOA finds its first feasible solution: 0.051 against 0.054 for both NOMAD with RBF and kriging. During the last fourth of the evaluations, the RBF-assisted NOMAD performs better than BOA.

Considering now the second run presented on Fig. 2 and in the lines corresponding to "run 2" in Table 11, there are clear gaps in the performance of the three solvers. The kriging-assisted NOMAD is effective in finding a feasible solution and performs better than the RBF-assisted NOMAD. The latter is outperformed by the other methods. BOA outperforms the surrogate-assisted NOMAD solvers as soon as it finds a feasible solution.

In both runs, although the plots corresponding to BOA start with a clear advantage on the objective value, the decrease seems globally slower than in NOMAD.

4.2.3 Algorithm comparisons on RSMLateralCrash in sequential mode

In order to see how parallelization affects the performance of the algorithms, one run with the classical sequential versions was also performed for each of them.

The comparison of the three methods is depicted in Figure 3 for the evolution of the best feasible objective values and summarized in Table 12. The

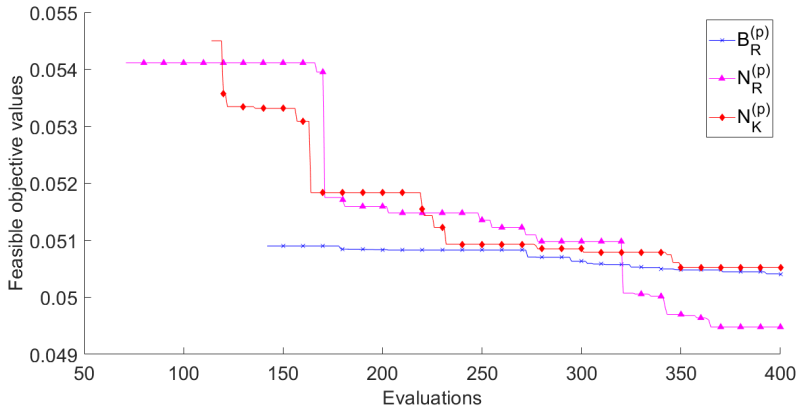

Fig. 1 Evolution of the best feasible objective values of RSMLateralCrash according to the number of evaluations for "run 1" of parallel BOA with cubic RBF ($B_R^{(p)}$), parallel NOMAD with cubic RBF ($N_R^{(p)}$) and parallel NOMAD with Gaussian kriging ($N_K^{(p)}$).

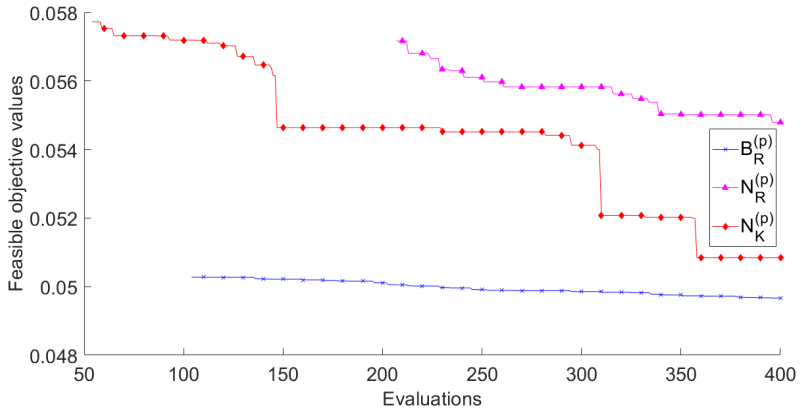

Fig. 2 Evolution of the best feasible objective values of RSMLateralCrash according to the number of evaluations for "run 2" of parallel BOA with cubic RBF ($B_R^{(p)}$), parallel NOMAD with cubic RBF ($N_R^{(p)}$) and parallel NOMAD with Gaussian kriging ($N_K^{(p)}$).

results of NOMAD are identical to those of the case using the block evaluation option. Thus, the parallelization does not seem to affect the performance of NOMAD on this problem. The situation is different for BOA whose internal strategy is modified to deal with simultaneous blackbox calls. In this setting, BOA outperforms NOMAD and both the first and best feasible solutions of BOA have better objective values than those found by the two variants of NOMAD.

Focusing on BOA, Figure 4 depicts the results for one run of the classical sequential version of BOA used with cubic RBF compared with its parallel

Table 11 Results for each run on RSMLateralCrash for parallel BOA with RBF ($B_R^{(p)}$), parallel NOMAD with cubic RBF ($N_R^{(p)}$) and parallel NOMAD with Gaussian kriging ($N_K^{(p)}$): number of function evaluations to reach a feasible solution $N^{(I)}$, first feasible objective value $f^{(I)}$ and best feasible objective value $f^{(II)}$.

Run	$\mathcal{A}$	$N^{(I)}$	$f^{(I)}$	$f^{(II)}$
1	$B_R^{(p)}$	152	0.051	0.050
	$N_R^{(p)}$	71	0.054	0.049
	$N_K^{(p)}$	114	0.054	0.051
2	$B_R^{(p)}$	127	0.050	0.050
	$N_R^{(p)}$	207	0.057	0.055
	$N_K^{(p)}$	54	0.058	0.051

Table 12 Results for one run on RSMLateralCrash for BOA with cubic RBF (B_R), NOMAD with cubic RBF (N_R) and NOMAD with Gaussian kriging (N_K): number of function evaluations to reach a feasible solution $N^{(I)}$, first feasible objective value $f^{(I)}$ and best feasible objective value $f^{(II)}$.

$\mathcal{A}$	$N^{(I)}$	$f^{(I)}$	$f^{(II)}$
B_R	97	0.047	0.047
N_R	71	0.054	0.049
N_K	114	0.054	0.051

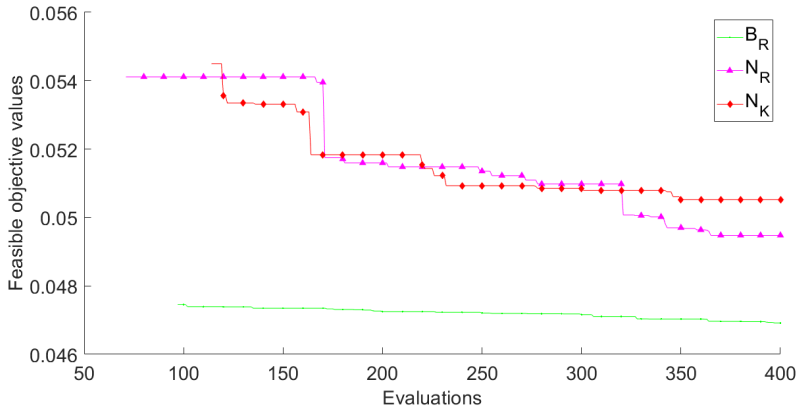

Fig. 3 Evolution of the best feasible objective values of RSMLateralCrash according to the number of evaluations for one run of BOA with cubic RBF (B_R), NOMAD with cubic RBF (N_R) and NOMAD with Gaussian kriging (N_K).

version. The sequential version needs less evaluations to leave Phase I and the feasible objective value of the best candidate is better than in the parallel version. Thus, the parallel resolution of the subproblems reduces the performance of the algorithm. This is partly due to a higher quality of the surrogate in the sequential strategy where it is updated after each expensive evaluation. On this optimization problem where the costs of the blackbox calls are significantly shortened as it uses surrogates instead of the expensive finite element

simulations, the total computational time was however reduced by two days thanks to the use of parallel evaluations (from approximately 9 to 7 calculation days).

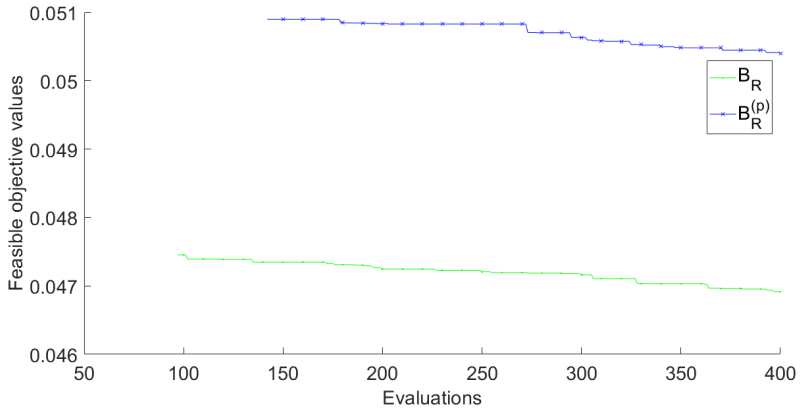

Fig. 4 Evolution of the best feasible objective values of RSMLateralCrash according to the number of evaluations for one run of BOA with cubic RBF (B_R) and the parallel version ($B_R^{(p)}$).

4.2.4 Algorithm comparisons on LateralCrash

The parallel versions of BOA using cubic RBF and cubic RBF-assisted NOMAD are tested on the lateral crash design problem from Stellantis. This problem is computationally very expensive as it makes use of finite element simulations of the vehicle and each call takes more than 12 hours. As a solution is commonly desired in a time window of two weeks, assuming that the time needed by a solver to generate a new candidate point is negligible, only a maximum of 28 evaluations are possible in a sequential mode, hence the need to evaluate candidate solutions in parallel. Furthermore, failures are frequently observed in vehicle simulations, which constitutes an additional challenge.

One run of parallel BOA is performed starting from an initial DOE that consists of 72 infeasible points generated with the same strategy used in the previous experiments. To cope with the long computational times, the maximum number of surrogate evaluations for each subproblem resolution is set to $50 \cdot n$ and block surrogate evaluations of maximum 50 points are allowed. One run of cubic RBF-assisted NOMAD is also performed where block evaluations are enabled. For both methods, the capacity of parallel blackbox evaluations is kept at 25 and the total blackbox evaluation budget is set at 400.

The run performed with the parallel version of NOMAD turned out to be infeasible: no feasible solution was found within the budget of 400 evaluations.

On the contrary, parallel BOA successfully completed its Phase I and found several feasible solutions.

The evolution of the best feasible objective values for parallel BOA using cubic RBF is presented in Figure 5 and Table 13 summarizes the results. The best solution of the initial DOE corresponds to an infeasible mass of 90.017kg and a best sum of squared constraint violations of $1.553 \cdot 10^5$. A feasible solution is found after 222 blackbox evaluations and the best feasible mass obtained at the end of the evaluation budget is 75.719kg. Hence, despite the difficulty of this mixed-variable problem with 48 variables and 64 inequality constraints, parallel BOA reached a feasible solution after only 6 iterations. Indeed, the algorithm first evaluates the initial DOE before entering any phase, so $222-72 = 150$ expensive evaluations are actually performed during Phase I, which corresponds to 6 batches of blackbox evaluations. Moreover, note that 31 failures of the blackbox occurred during the optimization but they did not stop BOA as its design comprises the handling of such hidden constraints.

$\mathcal{S}$	$N^{(I)}$	$f^{(I)}$	$f^{(II)}$
$\mathrm{B}_{\mathrm{R}}^{(\mathrm{p})}$	222	75.998	75.719

Table 13 Results for one run on LateralCrash for parallel BOA with RBF ($\mathrm{B}_{\mathrm{R}}^{(\mathrm{p})}$): number of function evaluations to reach a feasible solution $N^{(I)}$, first feasible objective value $f^{(I)}$ and best feasible objective value $f^{(II)}$.

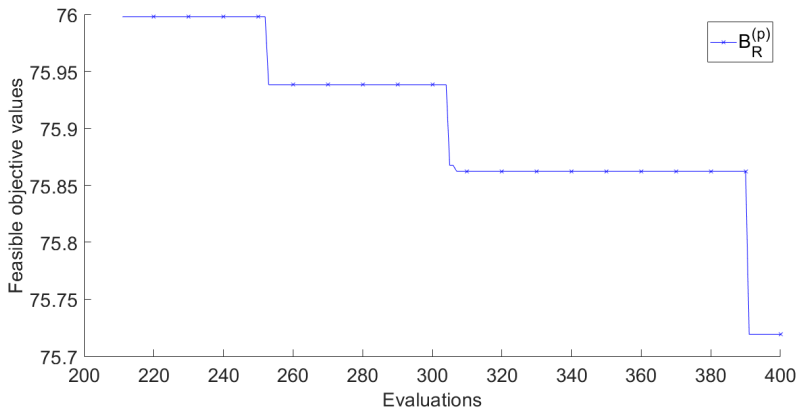

Fig. 5 Evolution of the best feasible objective values of LateralCrash according to the number of evaluations for one run of parallel BOA with RBF ($\mathrm{B}_{\mathrm{R}}^{(\mathrm{p})}$).

5 Conclusion

This paper presents a new surrogate-based generic method for solving expensive constrained blackbox optimization problems with mixed variables. The developed algorithm, entitled BOA, is flexible in terms of the type of models used or the solver used on its subproblems, and does not need feasible initial points thanks to its two-phase structure.

An extension of BOA for parallel evaluations is also described for real-world applications where launching batches of evaluations is essential to expect a reasonably good solution in a restricted time.

The results for different numerical experiments have been presented, using constrained optimization problems with more than 10 variables and up to 48. First comparisons of six kinds of surrogate models used for the subproblems of BOA are performed on 19 problems from the literature, including applications. They exhibit globally better performance of MARS and cubic RBF in terms of mean number of evaluations to reach a feasible solution and quality of the solution. In practice, the construction of MARS models is however computationally longer. Among the kriging types tested, the use of exponential correlation functions seems unfavourable in the presence of discrete variables.

Other experiments investigate the parameter λ that takes into account the objective value in the search of a feasible candidate solution. Comparing cubic RBF-based BOA with and without λ shows generally lower objective values at the end of Phase I when the parameter is used but more evaluations are performed in this phase of the algorithm. In general, however, the solution obtained at the end of the optimization is still better with λ.

Then, BOA using cubic RBF is compared with two surrogate-assisted NOMAD variants and using a higher evaluation budget. The results on the literature problems show an equivalent ability of the three methods to find feasible solutions on mixed-integer problems and a higher performance of BOA on the other types of problems. Regarding the quality of the solution, BOA globally finds lower objective values than the RBF- and kriging-assisted NOMAD. These methods are also tested on two automotive problems encountered at Stellantis.

On the response surface-based pole lateral crash problem, two runs are performed using parallel versions of each solver. They show that the first feasible solutions found by BOA have better objective values than the current best solutions of NOMAD variants at the corresponding number of evaluations. However, the number of evaluations used to find a feasible solution is better for at least one of the two NOMAD variants. On the final solutions identified, BOA is competitive or better than NOMAD. Comparing sequential and parallel versions of the three solvers on one run, the performance of NOMAD is unchanged whereas BOA outperforms its parallel extension both on the number of evaluations used in Phase I and on the best solution found.

Furthermore, BOA using parallel evaluations and cubic RBF was successfully applied to a real-world high-dimensional optimization problem from the automotive group Stellantis. Starting from an infeasible DOE, the optimiza-

tion reached a feasible solution in 6 iterations only and successfully managed the failures of the blackbox. On this problem NOMAD, also used with parallel evaluations and cubic RBF, did not find a feasible solution.

As a summary, this study exhibits an efficiency of the first phase of BOA in finding good feasible solutions and shows advantages of the method in a context of restricted evaluation budgets, which is often the case in industry. It can be considered as a relevant method for solving real-world expensive black-box optimization problems with mixed variables and inequality constraints. A better decrease of the objective value should be possible with improvements of the second phase of the algorithm, for instance by considering alternative formulations of the corresponding subproblem. Besides, a specific handling for categorical variables and handling of the presence of noise in the blackbox functions is a matter for future work.

Acknowledgements The authors wish to thank the reviewers for their time and efforts towards improving our manuscript and their valuable comments.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Abramson, M.A., Audet, C., Chrissis, J.W., Walston, J.G.: Mesh adaptive direct search algorithms for mixed variable optimization. Optimization Letters **3**(1), 35–47 (2009). DOI https://doi.org/10.1007/s11590-008-0089-2
2. Abramson, M.A., Audet, C., Dennis, Jr., J.E., Le Digabel, S.: OrthoMADS: A Deterministic MADS Instance with Orthogonal Directions. SIAM Journal on Optimization **20**(2), 948–966 (2009). DOI https://doi.org/10.1137/080716980
3. Asher, M.J., Croke, B.F.W., Jakeman, A.J., Peeters, L.J.M.: A review of surrogate models and their application to groundwater modeling. Water Resources Research **51**(8), 5957–5973 (2015)
4. Audet, C., Côté-Massicotte, J.: Dynamic improvements of static surrogates in direct search optimization. Optimization Letters **13**(6), 1433–1447 (2019). DOI 10.1007/s11590-019-01452-7
5. Audet, C., Dennis, Jr., J.: Mesh Adaptive Direct Search Algorithms for Constrained Optimization. SIAM Journal on Optimization **17**(1), 188–217 (2006). DOI https://doi.org/10.1137/040603371
6. Audet, C., Dennis, Jr., J.: A Progressive Barrier for Derivative-Free Nonlinear Programming. SIAM Journal on Optimization **20**(1), 445–472 (2009). DOI https://doi.org/10.1137/070692662
7. Audet, C., Le Digabel, S., Rochon Montplaisir, V., Tribes, C.: Algorithm 1027: NOMAD version 4: Nonlinear Optimization with the MADS algorithm. ACM Transactions on Mathematical Software **48**(3), 35:1–35:22 (2022). DOI https://doi.org/10.1145/3544489
8. Bagheri, S., Konen, W., Emmerich, M., Bäck, T.: Self-adjusting parameter control for surrogate-assisted constrained optimization under limited budgets. Applied Soft Computing **61**, 377–393 (2017). DOI https://doi.org/10.1016/j.asoc.2017.07.060
9. Berman, O., Ashrafi, N.: Optimization models for reliability of modular software systems. IEEE Transactions on Software Engineering **19**(11), 1119–1123 (1993). DOI https://doi.org/10.1109/32.256858

10. Booker, A.J., Dennis, J.E., Frank, P.D., Serafini, D.B., Torczon, V., Trosset, M.W.: A rigorous framework for optimization of expensive functions by surrogates. Structural optimization **17**(1), 1–13 (1999). DOI https://doi.org/10.1007/BF01197708
11. Bouhlel, M.A., Bartoli, N., Regis, R.G., Otsmane, A., Morlier, J.: Efficient global optimization for high-dimensional constrained problems by using the kriging models combined with the partial least squares method. Engineering Optimization **50**(12), 2038–2053 (2018). DOI https://doi.org/10.1080/0305215X.2017.1419344
12. Browne, T., Iooss, B., Gratiet, L.L., Lonchampt, J., Remy, E.: Stochastic simulators based optimization by gaussian process metamodels–application to maintenance investments planning issues. Quality and Reliability Engineering International **32**(6), 2067–2080 (2016). DOI https://doi.org/10.1002/qre.2028
13. Bussieck, M.R., Drud, A.S., Meeraus, A.: MINLPLib—a collection of test models for mixed-integer nonlinear programming. INFORMS Journal on Computing **15**(1), 114–119 (2003)
14. Conn, A.R., Deleris, L.A., Hosking, J.R., Thorstensen, T.A.: A simulation model for improving the maintenance of high cost systems, with application to an offshore oil installation. Quality and Reliability Engineering International **26**(7), 733–748 (2010). DOI https://doi.org/10.1002/qre.1136
15. Crélot, A.S., Beauthier, C., Orban, D., Sainvitu, C., Sartenaer, A.: Combining surrogate strategies with MADS for mixed-variable derivative-free optimization. Tech. Rep. G-2017-70, Les cahiers du GERAD (2017). DOI https://doi.org/10.13140/RG.2.2.25690.24008
16. Dahito, M.A., Genest, L., Maddaloni, A., Neto, J.: On the performance of the orthomads algorithm on continuous and mixed-integer optimization problems. In: A.I. Pereira, F.P. Fernandes, J.P. Coelho, J.P. Teixeira, M.F. Pacheco, P. Alves, R.P. Lopes (eds.) Optimization, Learning Algorithms and Applications, pp. 31–47. Springer International Publishing, Cham (2021)
17. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation **6**(2), 182–197 (2002)
18. Egea, J.A., Vazquez, E., Banga, J.R., Martí, R.: Improved scatter search for the global optimization of computationally expensive dynamic models. Journal of Global Optimization **43**(2-3), 175–190 (2009). DOI https://doi.org/10.1007/s10898-007-9172-y
19. Floudas, C.A., Pardalos, P.M.: A Collection of Test Problems for Constrained Global Optimization Algorithms, vol. 455. Springer, Berlin, Heidelberg (1990). DOI https://doi.org/10.1007/3-540-53032-0
20. Forrester, A.I., Keane, A.J.: Recent advances in surrogate-based optimization. Progress in Aerospace Sciences **45**(1), 50–79 (2009). DOI https://doi.org/10.1016/j.paerosci.2008.11.001
21. Friedman, J.H.: Multivariate adaptive regression splines. The annals of statistics pp. 1–67 (1991)
22. Gandomi, A.H., Yang, X.S., Alavi, A.H.: Mixed variable structural optimization using firefly algorithm. Computers & Structures **89**(23), 2325–2336 (2011). DOI https://doi.org/10.1016/j.compstruc.2011.08.002
23. Gramacy, R.B., Le Digabel, S.: The mesh adaptive direct search algorithm with treed Gaussian process surrogates. Pacific Journal of Optimization **11**(3), 419–447 (2015). URL `http://www.ybook.co.jp/online2/pjov11-3.html`
24. Grandhi, R., Venkayya, V.: Structural optimization with frequency constraints. AIAA journal **26**(7), 858–866 (1988). DOI https://doi.org/10.2514/3.9979
25. Grossmann, I.E., Sargent, R.W.H.: Optimum design of multipurpose chemical plants. Industrial & Engineering Chemistry Process Design and Development **18**(2), 343–348 (1979). DOI https://doi.org/10.1021/i260070a031
26. Gu, L., Yang, R., Tho, C.H., Makowskit, M., Faruquet, O., Y. Li, Y.L.: Optimisation and robustness for crashworthiness of side impact. International Journal of Vehicle Design **26**(4), 348–360 (2001). DOI https://doi.org/10.1504/IJVD.2001.005210
27. Gupta, S., Tiwari, R., Nair, S.B.: Multi-objective design optimisation of rolling bearings using genetic algorithms. Mechanism and Machine Theory **42**(10), 1418–1443 (2007). DOI https://doi.org/10.1016/j.mechmachtheory.2006.10.002

28. Hardy, R.L.: Multiquadric equations of topography and other irregular surfaces. Journal of geophysical research **76**(8), 1905–1915 (1971)
29. Himmelblau, D.M.: Applied nonlinear programming. McGraw-Hill Book Company, New York (1972)
30. Hock, W., Schittkowski, K.: Test examples for nonlinear programming codes. Journal of Optimization Theory and Applications **30**(1), 127–129 (1980). DOI https://doi.org/10.1007/BF00934594
31. Hüsken, M., Jin, Y., Sendhoff, B.: Structure optimization of neural networks for evolutionary design optimization. Soft Computing **9**(1), 21–28 (2005). DOI https://doi.org/10.1007/s00500-003-0330-y
32. Hutter, F., Hoos, H.H., Leyton-Brown, K.: Sequential model-based optimization for general algorithm configuration. In: C.A.C. Coello (ed.) Learning and Intelligent Optimization, pp. 507–523. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
33. Jekabsons, G.: ARESLab: Adaptive regression splines toolbox for matlab/octave, ver. 1.13.0 (2011)
34. Jin, L., Alpak, F.O., van den Hoek, P., Pirmez, C., Fehintola, T., Tendo, F., Olaniyan, E.: A comparison of stochastic data-integration algorithms for the joint history matching of production and time-lapse-seismic data. SPE Reservoir Evaluation & Engineering **15**(04), 498–512 (2012). DOI https://doi.org/10.2118/146418-PA
35. Jin, R., Chen, W., Simpson, T.W.: Comparative studies of metamodelling techniques under multiple modelling criteria. Structural and multidisciplinary optimization **23**(1), 1–13 (2001). DOI https://doi.org/10.1007/s00158-001-0160-4
36. 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). DOI https://doi.org/10.1023/A:1008306431147
37. Kianifar, M.R., Campean, F.: Performance evaluation of metamodelling methods for engineering problems: towards a practitioner guide. Structural and Multidisciplinary Optimization **61**(1), 159–186 (2020). DOI https://doi.org/10.1007/s00158-019-02352-1
38. Koch, P.N., Bagheri, S., Foussette, C., Krause, P., Bäck, T., Konen, W.: Constrained optimization with a limited number of function evaluations. In: F. Hoffmann, E. Hüllermeier (eds.) Proc. 24. Workshop Computational Intelligence, pp. 119–134. Universitätsverlag Karlsruhe (2014)
39. Krige, D.G.: A statistical approach to some basic mine valuation problems on the witwatersrand. Journal of the Southern African Institute of Mining and Metallurgy **52**(6), 119–139 (1951)
40. Kumar, A., Wu, G., Ali, M.Z., Mallipeddi, R., Suganthan, P.N., Das, S.: A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. Swarm and Evolutionary Computation **56**, 100693 (2020). DOI https://doi.org/10.1016/j.swevo.2020.100693
41. Kuo, W., Prasad, V.R., Tillman, F.A., Hwang, C.L.: Optimal Reliability Design: Fundamentals and Applications. Cambridge University Press, United Kingdom (2001)
42. Le Digabel, S., Wild, S.: A taxonomy of constraints in simulation-based optimization. Tech. Rep. G-2015-57, Les cahiers du GERAD (2015). URL `http://www.optimization-online.org/DB_HTML/2015/05/4931.html`
43. Li, R., Emmerich, M.T., Eggermont, J., Bovenkamp, E.G., Back, T., Dijkstra, J., Reiber, J.H.: Metamodel-assisted mixed-integer evolution strategies and their application to intravascular ultrasound image analysis. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pp. 2764–2771. IEEE (2008). DOI https://doi.org/10.1109/CEC.2008.4631169
44. Lim, D., Jin, Y., Ong, Y.S., Sendhoff, B.: Generalizing surrogate-assisted evolutionary computation. IEEE Transactions on Evolutionary Computation **14**(3), 329–355 (2010). DOI https://doi.org/10.1109/TEVC.2009.2027359
45. Lim, D., Ong, Y.S., Jin, Y., Sendhoff, B.: A study on metamodeling techniques, ensembles, and multi-surrogates in evolutionary computation. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, GECCO '07, p. 1288–1295. Association for Computing Machinery, New York, NY, USA (2007). DOI https://doi.org/10.1145/1276958.1277203

46. Liu, C., Wan, Z., Liu, Y., Li, X., Liu, D.: Trust-region based adaptive radial basis function algorithm for global optimization of expensive constrained black-box problems. Applied Soft Computing **105**, 107233 (2021). DOI https://doi.org/10.1016/j.asoc.2021.107233
47. Lophaven, S.N., Nielsen, H.B., Søndergaard, J.: DACE–a matlab kriging toolbox, version 2.0 (2002)
48. Marsden, A.L., Feinstein, J.A., Taylor, C.A.: A computational framework for derivative-free optimization of cardiovascular geometries. Computer Methods in Applied Mechanics and Engineering **197**(21), 1890–1905 (2008). DOI https://doi.org/10.1016/j.cma.2007.12.009
49. Marsden, A.L., Wang, M., Dennis, Jr., J.E., Moin, P.: Optimal aeroacoustic shape design using the surrogate management framework. Optimization and Engineering **5**(2), 235–262 (2004). DOI https://doi.org/10.1023/B:OPTE.0000033376.89159.65
50. Martinez, N., Anahideh, H., Rosenberger, J.M., Martinez, D., Chen, V.C., Wang, B.P.: Global optimization of non-convex piecewise linear regression splines. Journal of Global Optimization **68**(3), 563–586 (2017). DOI https://doi.org/10.1007/s10898-016-0494-5
51. 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. Technometrics **42**(1), 55–61 (2000). DOI https://doi.org/10.1080/00401706.2000.10485979
52. Müller, J.: MISO: mixed-integer surrogate optimization framework. Optimization and Engineering **17**(1), 177–203 (2016). DOI https://doi.org/10.1007/s11081-015-9281-2
53. Müller, J., Shoemaker, C.A.: Influence of ensemble surrogate models and sampling strategy on the solution quality of algorithms for computationally expensive black-box global optimization problems. Journal of Global Optimization **60**(2), 123–144 (2014). DOI https://doi.org/10.1007/s10898-014-0184-0
54. Müller, J., Shoemaker, C.A., Piché, R.: SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems. Computers & Operations Research **40**(5), 1383–1400 (2013). DOI https://doi.org/10.1016/j.cor.2012.08.022
55. Müller, J., Shoemaker, C.A., Piché, R.: SO-I: a surrogate model algorithm for expensive nonlinear integer programming problems including global optimization applications. Journal of Global Optimization **59**(4), 865–889 (2014). DOI https://doi.org/10.1007/s10898-013-0101-y
56. Müller, J., Woodbury, J.D.: GOSAC: global optimization with surrogate approximation of constraints. Journal of Global Optimization **69**(1), 117–136 (2017). DOI https://doi.org/10.1007/s10898-017-0496-y
57. Pant, M., Thangaraj, R., Singh, V.P.: Optimization of mechanical design problems using improved differential evolution algorithm. International Journal of Recent Trends in Engineering **1**(5), 21–25 (2009)
58. Paul H., T.: Optimal design of an industrial refrigeration system. In: Proceedings of International Conference on Optimization Techniques and Applications, pp. 427–435. Singapore, National University of Singapore (1987)
59. Plackett, R.L., Burman, J.P.: The design of optimum multifqctorial experiments. Biometrika **33**(4), 305–325 (1946). DOI 10.1093/biomet/33.4.305
60. Regis, R.G.: Stochastic radial basis function algorithms for large-scale optimization involving expensive black-box objective and constraint functions. Computers & Operations Research **38**(5), 837–853 (2011). DOI https://doi.org/10.1016/j.cor.2010.09.013
61. Regis, R.G.: Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. Engineering Optimization **46**(2), 218–243 (2014). DOI https://doi.org/10.1080/0305215X.2013.765000
62. Regis, R.G.: Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions. IEEE Transactions on Evolutionary Computation **18**(3), 326–347 (2014). DOI https://doi.org/10.1109/TEVC.2013.2262111
63. Regis, R.G.: Surrogate-assisted particle swarm with local search for expensive constrained optimization. In: P. Korošec, N. Melab, E.G. Talbi (eds.) Bioinspired Optimization Methods and Their Applications, pp. 246–257. Springer International Publishing, Cham (2018)

64. Regis, R.G.: Large-scale discrete constrained black-box optimization using radial basis functions. In: 2020 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 2924–2931. IEEE (2020). DOI https://doi.org/10.1109/SSCI47803.2020.9308581
65. Regis, R.G.: A survey of surrogate approaches for expensive constrained black-box optimization. In: H.A. Le Thi, H.M. Le, T. Pham Dinh (eds.) Optimization of Complex Systems: Theory, Models, Algorithms and Applications, pp. 37–47. Springer, Cham (2020). DOI https://doi.org/10.1007/978-3-030-21803-4_4
66. Regis, R.G., Wild, S.M.: CONORBIT: constrained optimization by radial basis function interpolation in trust regions. Optimization Methods and Software **32**(3), 552–580 (2017). DOI https://doi.org/10.1080/10556788.2016.1226305
67. Rios, L.M., Sahinidis, N.V.: Derivative-free optimization: a review of algorithms and comparison of software implementations. Journal of Global Optimization **56**(3), 1247–1293 (2013). DOI https://doi.org/10.1007/s10898-012-9951-y
68. Runarsson, T.P.: Constrained evolutionary optimization by approximate ranking and surrogate models. In: X. Yao, E.K. Burke, J.A. Lozano, J. Smith, J.J. Merelo-Guervós, J.A. Bullinaria, J.E. Rowe, P. Tiňo, A. Kabán, H.P. Schwefel (eds.) Parallel Problem Solving from Nature - PPSN VIII, pp. 401–410. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
69. Sacks, J., Welch, W.J., Mitchell, T.J., Wynn, H.P.: Design and analysis of computer experiments. Statistical science pp. 409–423 (1989)
70. Sasena, M.J., Papalambros, P., Goovaerts, P.: Exploration of metamodeling sampling criteria for constrained global optimization. Engineering Optimization **34**(3), 263–278 (2002). DOI https://doi.org/10.1080/03052150211751
71. Simpson, T.W., Mauery, T.M., Korte, J.J., Mistree, F.: Kriging models for global approximation in simulation-based multidisciplinary design optimization. AIAA Journal **39**(12), 2233–2241 (2001). DOI https://doi.org/10.2514/2.1234
72. Talgorn, B., Le Digabel, S., Kokkolaras, M.: Statistical Surrogate Formulations for Simulation-Based Design Optimization. Journal of Mechanical Design **137**(2), 021405–1–021405–18 (2015). DOI 10.1115/1.4028756
73. Thanedar, P., Vanderplaats, G.: Survey of discrete variable optimization for structural design. Journal of Structural Engineering **121**(2), 301–306 (1995)
74. Torczon, V.: On the convergence of pattern search algorithms. SIAM Journal on Optimization **7**(1), 1–25 (1997). DOI https://doi.org/10.1137/S1052623493250780
75. Villa-Vialaneix, N., Follador, M., Ratto, M., Leip, A.: A comparison of eight metamodeling techniques for the simulation of N2O fluxes and N leaching from corn crops. Environmental Modelling & Software **34**, 51–66 (2012). DOI https://doi.org/10.1016/j.envsoft.2011.05.003
76. Vu, K.K., d'Ambrosio, C., Hamadi, Y., Liberti, L.: Surrogate-based methods for black-box optimization. International Transactions in Operational Research **24**(3), 393–424 (2017). DOI https://doi.org/10.1111/itor.12292
77. Wang, G.G., Shan, S.: Review of metamodeling techniques in support of engineering design optimization. Journal of Mechanical Design **129**(4), 370–380 (2006). DOI https://doi.org/10.1115/1.2429697
78. Wang, H., Jin, Y., Doherty, J.: Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems. IEEE Transactions on Cybernetics **47**(9), 2664–2677 (2017). DOI https://doi.org/10.1109/TCYB.2017.2710978
79. Wang, Y., Liu, H., Long, H., Zhang, Z., Yang, S.: Differential evolution with a new encoding mechanism for optimizing wind farm layout. IEEE Transactions on Industrial Informatics **14**(3), 1040–1054 (2018). DOI https://doi.org/10.1109/TII.2017.2743761
80. Wild, S.M., Regis, R.G., Shoemaker, C.A.: ORBIT: Optimization by radial basis function interpolation in trust-regions. SIAM Journal on Scientific Computing **30**(6), 3197–3219 (2008). DOI https://doi.org/10.1137/070691814
81. Yang, H., Kim, J., Choe, J.: Field development optimization in mature oil reservoirs using a hybrid algorithm. Journal of Petroleum Science and Engineering **156**, 41–50 (2017). DOI https://doi.org/10.1016/j.petrol.2017.05.009
82. Ye, K.Q., Li, W., Sudjianto, A.: Algorithmic construction of optimal symmetric latin hypercube designs. Journal of Statistical Planning and Inference **90**(1), 145–159 (2000). DOI https://doi.org/10.1016/S0378-3758(00)00105-1

83. Yokota, T., Taguchi, T., Gen, M.: A solution method for optimal weight design problem of the gear using genetic algorithms. Computers & Industrial Engineering **35**(3), 523–526 (1998). DOI https://doi.org/10.1016/S0360-8352(98)00149-1. Selected Papers from the 22nd ICC and IE Conference
84. Yuan, X., Zhang, S., Pibouleau, L., Domenech, S.: Une méthode d'optimisation non linéaire en variables mixtes pour la conception de procédés. RAIRO-Operations Research **22**(4), 331–346 (1988)
85. Zhuang, L., Tang, K., Jin, Y.: Metamodel assisted mixed-integer evolution strategies based on kendall rank correlation coefficient. In: H. Yin, K. Tang, Y. Gao, F. Klawonn, M. Lee, T. Weise, B. Li, X. Yao (eds.) Intelligent Data Engineering and Automated Learning – IDEAL 2013, pp. 366–375. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)

A Formulations of the test problems

A.1 Problem C1 [19]

This is the well-known G01 benchmark problem.

$$\begin{cases} \min 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i \\ \text{subject to} \\ 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0 \\ 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0 \\ 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0 \\ -8x_1 + x_{10} \leq 0 \\ -8x_2 + x_{11} \leq 0 \\ -8x_3 + x_{12} \leq 0 \\ -2x_4 - x_5 + x_{10} \leq 0 \\ -2x_6 - x_7 + x_{11} \leq 0 \\ -2x_8 - x_9 + x_{12} \leq 0 \\ x_i \in [0,1],\ i = 1,2,\ldots,9,13 \\ x_i \in [0,100],\ i = 10,11,12. \end{cases}$$

A.2 Problem C2 [30]

This is the well-known G07 benchmark problem.

$$\begin{cases} \min\ x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3-10)^2 + 4(x_4-5)^2 + (x_5-3)^2 + 2(x_6-1)^2 \\ \qquad +5x_7^2 + 7(x_8-11)^2 + 2(x_9-10)^2 + (x_{10}-7)^2 + 45 \\ \text{subject to} \\ \qquad -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0 \\ \qquad 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0 \\ \qquad -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0 \\ \qquad 3(x_1-2)^2 + 4(x_2-3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0 \\ \qquad 5x_1^2 + 8x_2 + (x_3-6)^2 - 2x_4 - 40 \leq 0 \\ \qquad x_1^2 + 2(x_2-2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leq 0 \\ \qquad 0.5(x_1-8)^2 + 2(x_2-4)^2 + 3x_5^2 - x_6 - 30 \leq 0 \\ \qquad -3x_1 + 6x_2 + 12(x_9-8)^2 - 7x_{10} \leq 0 \\ x_i \in [-10,10],\ i = 1,2,\ldots,10. \end{cases}$$

A.3 Problem C3 [29]

This is the well-known G19 benchmark problem.

$$a = \begin{bmatrix} 16 & 2 & 0 & 1 & 0 \\ 0 & -2 & 0 & 0.4 & 2 \\ -3.5 & 0 & 2 & 0 & 0 \\ 0 & -2 & 0 & -4 & -1 \\ 0 & -9 & -2 & 1 & -2.8 \\ 2 & 0 & -4 & 0 & 0 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -2 & -3 & -2 & -1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad b = \begin{bmatrix} -40 \\ -2 \\ -0.25 \\ -4 \\ -4 \\ -1 \\ -40 \\ -60 \\ 5 \\ 1 \end{bmatrix} \quad c = \begin{bmatrix} 30 & -20 & -10 & 32 & -10 \\ -20 & 39 & -6 & -31 & 32 \\ -10 & -6 & 10 & -6 & -10 \\ 32 & -31 & -6 & 39 & -20 \\ -10 & 32 & -10 & -20 & 30 \end{bmatrix} \quad d = \begin{bmatrix} 4 \\ 8 \\ 10 \\ 6 \\ 2 \end{bmatrix} \quad e = \begin{bmatrix} -15 \\ -27 \\ -36 \\ -18 \\ -12 \end{bmatrix}$$

$$\begin{cases} \min \sum_{j=1}^{5} \sum_{i=1}^{5} c_{ij} x_{(10+i)} x_{(10+j)} + 2 \sum_{i=1}^{5} d_i x_{(10+i)}^3 - \sum_{i=1}^{10} b_i x_i \\ \text{subject to} \\ -2 \sum_{i=1}^{5} c_{ij} x_{(10+i)} - 3 d_j x_{(10+j)}^2 - e_j + \sum_{i=1}^{10} a_{ij} x_i \leq 0, \ j = 1, 2, \ldots, 5 \\ x_i \in [0, 10], \ i = 1, 2, \ldots, 15. \end{cases}$$

A.4 Problem C4 [58,57,40]: *optimal design of an industrial refrigeration system*

A design problem expressed as a non-linear inequality constrained optimization problem.

$$\begin{cases} \min \ 63098.88 x_2 x_4 x_{12} + 5441.5 x_2^2 x_{12} + 115055.5 x_2^{1.664} x_6 + \\ \quad 6172.27 x_2^2 x_6 + 63098.88 x_1 x_3 x_{11} + 5441.5 x_1^2 x_{11} + 115055.5 x_1^{1.664} x_5 \\ \quad + 6172.27 x_1^2 x_5 + 140.53 x_1 x_{11} + 281.29 x_3 x_{11} + 70.26 x_1^2 + 281.29 x_1 x_3 \\ \quad + 281.29 x_3^2 + 14437 x_8^{1.8812} x_{12}^{0.3424} x_{10} x_{14}^{-1} x_1^2 x_7 x_9^{-1} + 20470.2 x_7^{2.893} x_{11}^{0.316} x_1^2 \\ \text{subject to} \\ 1.524 x_7^{-1} - 1 \leq 0 \\ 1.524 x_8^{-1} - 1 \leq 0 \\ 0.07789 x_1 - 2 x_7^{-1} x_9 - 1 \leq 0 \\ 7.05305 x_9^{-1} x_1^2 x_{10} x_8^{-1} x_2^{-1} x_{14}^{-1} - 1 \leq 0 \\ 0.0833 x_{13}^{-1} x_{14} - 1 \leq 0 \\ 47.136 x_2^{0.333} x_{10}^{-1} x_{12} - 1.333 x_8 x_{13}^{2.1195} + 62.08 x_{13}^{2.1195} x_{12}^{-1} x_8^{0.2} x_{10}^{-1} - 1 \leq 0 \\ 0.04771 x_{10} x_8^{1.8812} x_{12}^{0.3424} - 1 \leq 0 \\ 0.0488 x_9 x_7^{1.893} x_{11}^{0.316} - 1 \leq 0 \\ 0.0099 x_1 x_3^{-1} - 1 \leq 0 \\ 0.0193 x_2 x_4^{-1} - 1 \leq 0 \\ 0.0298 x_1 x_5^{-1} - 1 \leq 0 \\ 0.056 x_2 x_6^{-1} - 1 \leq 0 \\ 2 x_9^{-1} - 1 \leq 0 \\ 2 x_{10}^{-1} - 1 \leq 0 \\ x_{12} x_{11}^{-1} - 1 \leq 0 \\ x_i \in [0.001, 5], \ i = 1, 2, \ldots, 14. \end{cases}$$

A.5 Problem C5 [24,40]: *10-bar truss design*

The aim is to minimize the weight of a truss structure subject to frequency constraints. The truss is represented as a finite element structure that has 10 two-dimensional bar elements and 6 nodes.

$$\begin{cases} \min \ f(\overline{x}) = \sum_{i=1}^{10} L_i \rho A_i \\ \text{subject to} \\ \frac{7}{\omega_1(\overline{x})} - 1 \leq 0 \\ \frac{15}{\omega_2(\overline{x})} - 1 \leq 0 \\ \frac{20}{\omega_3(\overline{x})} - 1 \leq 0 \end{cases}$$

with bounds:

$$A_i \in [6.45 \cdot 10^{-5} \ , \ 5 \cdot 10^{-3}], \ i = 1, 2, \ldots, 10,$$

where

$$\overline{x} = \{A_1, A_2, \ldots, A_{10}\}, \ \rho = 2770,$$

$$L_i = \begin{cases} 9.144 & \text{if } i \leq 6 \\ 9.144 \cdot \sqrt{2} & \text{otherwise.} \end{cases}$$

The functions $\omega_1(\overline{x}), \omega_2(\overline{x})$ and $\omega_3(\overline{x})$ are computed from matrices K and M, that need to be assembled from smaller matrices, and their lowest eigenvalues.

Let

$$M^{(i)} = \frac{1}{6}\rho L_i A_i \begin{bmatrix} 2&0&1&0\\ 0&2&0&1\\ 1&0&2&0\\ 0&1&0&2 \end{bmatrix} \text{ and } K^{(i)} = \frac{EA_i}{L_i^3} \begin{bmatrix} -l_i m_i & -m_i^2 & l_i m_i & m_i^2\\ -l_i^2 & -l_i m_i & l_i^2 & l_i m_i\\ l_i m_i & m_i^2 & -l_i m_i & -m_i^2\\ l_i^2 & l_i m_i & -l_i^2 & -l_i m_i \end{bmatrix}$$

with

$$E = 6.98 \cdot 10^{10}, l_i = \begin{cases} 0 & \text{if } i \in \{5,6\}\\ 9.144 & \text{otherwise} \end{cases} \text{ and } m_i = \begin{cases} 0 & \text{if } i \leq 4\\ -9.144 & \text{if } i \in \{7,9\}\\ 9.144 & \text{otherwise.} \end{cases}$$

Let $\mathcal{I} = \begin{bmatrix} 5&6&9&10\\ 1&2&5&6\\ 7&8&11&12\\ 3&4&7&8\\ 5&6&7&8\\ 1&2&3&4\\ 7&8&9&10\\ 5&6&11&12\\ 3&4&5&6\\ 1&2&7&8 \end{bmatrix}$, we denote $\mathcal{I}_{i,:} = \begin{bmatrix} \mathcal{I}_{i,1} & \mathcal{I}_{i,2} & \mathcal{I}_{i,3} & \mathcal{I}_{i,4} \end{bmatrix}$, the i^{th} line of $\mathcal{I}$ where, for all $j \in \{1,2,3,4\}$, $\mathcal{I}_{i,j}$ is the element of the i^{th} line and j^{th} column of $\mathcal{I}$.

Let $A \in \mathbb{R}^{12\times 12}$ be a real square matrix, and $v = [a\ b\ c\ d]$ be a line vector with $\{a,b,c,d\} \in \{1,2,\ldots,12\}^4$, we denote $A[v] = A[a\ b\ c\ d] = \begin{bmatrix} A_{aa} & A_{ab} & A_{ac} & A_{ad}\\ A_{ba} & A_{bb} & A_{bc} & A_{bd}\\ A_{ca} & A_{cb} & A_{cc} & A_{cd}\\ A_{da} & A_{db} & A_{dc} & A_{dd} \end{bmatrix}$.

The following procedure describes how $\omega_1(\overline{x}), \omega_2(\overline{x})$ and $\omega_3(\overline{x})$ are computed:

Algorithm 5: Compute ω_1, ω_2 and ω_3

Given $\forall i \in \{1,2,\ldots,10\}$, $M^{(i)}$, $K^{(i)}$
Given $\mathcal{I}$,
Initialize $K \leftarrow \begin{bmatrix} 0&\ldots&0\\ \vdots&\ddots&\vdots\\ 0&\ldots&0 \end{bmatrix}$, $M \leftarrow \begin{bmatrix} 0&\ldots&0\\ \vdots&\ddots&\vdots\\ 0&\ldots&0 \end{bmatrix}$
for *each* $i \in \{1,2,\ldots,10\}$ **do**
    $K[\mathcal{I}_{i,:}] \leftarrow K[\mathcal{I}_{i,:}] + K^{(i)}$
    $M[\mathcal{I}_{i,:}] \leftarrow M[\mathcal{I}_{i,:}] + M^{(i)}$
Get the 3 smallest eigenvalues $\mathcal{L}_1, \mathcal{L}_2$ and $\mathcal{L}_3$ of K and M such that $\mathcal{L}_1 \leq \mathcal{L}_2 \leq \mathcal{L}_3$
for *each* $j \in \{1,2,3\}$ **do**
    $\omega_j(\overline{x}) \leftarrow \frac{\mathcal{L}_j}{2\pi}$
Return $\omega_1(\overline{x}), \omega_2(\overline{x})$ and $\omega_3(\overline{x})$.

A.6 Problem C6 [79,40]: *wind farm layout problem*

The objective is to minimize the opposite sum of the expected power output of each wind turbine i with minimum distance constraints between the wind turbines. The optimization problem is as follows:

$$\begin{cases} \min -\sum_{i=1}^{N} E(P_i)\\ \text{subject to}\\ 5R - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq 0,\ j = 1,2,\ldots,N \text{ and } j \neq i \end{cases}$$

where

$\underline{x} + R \leq x_i \leq \overline{x} - R$ and $\underline{y} + R \leq y_i \leq \overline{y} - R, \forall i = 1, 2, \ldots, N$, with $N = 15$,

$\underline{x} = [0\ 0 \ldots 0]^\top$ and $\underline{y} = [0\ 0 \ldots 0]^\top$ are lower bounds for all components of x and y, respectively,

$\overline{x} = [2000\ 2000 \ldots 2000]^\top$ and $\overline{y} = [2000\ 2000 \ldots 2000]^\top$ are upper bounds for all components of x and y respectively.

$$E(P_i) = \sum_{n=1}^{h} \xi_n \left\{ P_r \left(e^{-\left(\nu_r / c_i'\left(\left(\theta_{n-1}+\theta_n\right)/2\right)\right)^{k_i\left(\left(\theta_{n-1}+\theta_n\right)/2\right)}} - e^{-\left(\nu_{co} / c_i'\left(\left(\theta_{n-1}+\theta_n\right)/2\right)\right)^{k_i\left(\left(\theta_{n-1}+\theta_n\right)/2\right)}} \right) + \sum_{j=1}^{s} \left(e^{-\left(\nu_{j-1} / c_i'\left(\left(\theta_{n-1}+\theta_n\right)/2\right)\right)^{k_i\left(\left(\theta_{n-1}+\theta_n\right)/2\right)}} - e^{-\left(\nu_j / c_i'\left(\left(\theta_{n-1}+\theta_n\right)/2\right)\right)^{k_i\left(\left(\theta_{n-1}+\theta_n\right)/2\right)}} \right) \frac{e^{\left(\nu_{j-1}+\nu_j\right)/2}}{\alpha + \beta e^{\left(\nu_{j-1}+\nu_j\right)/2}} \right\}$$

ξ_n is the frequency of the interval $[\theta_{n-1}, \theta_n)$.

The following parameters are set: $h = 24$, $s = 36$, $R = 40$, $P_r = 1500$, $\alpha = 6.0268$, $\beta = 0.0007$, $\nu_r = 14$, $\nu_{co} = 25$ and $\nu_{ci} = 3.5$.

$\forall n \in \{1, 2, \ldots, h\}, \theta_n = \theta_{n-1} + \frac{360}{h}$ with $\theta_0 = 0°$.

$\forall j \in \{1, 2, \ldots, s\}, \nu_j = \nu_{j-1} + \frac{(\nu_r - \nu_{ci})}{s}$ with $\nu_0 = \nu_{ci}$.

For all $n \in \{1, 2, \ldots, h\}$, we denote $\theta^{(n)} = \frac{\theta_{n-1}+\theta_n}{2}$.

For all $n \in \{1, 2, \ldots, h\}$, $k_i(\theta^{(n)}) = 2$ and the following table gives the values of $c_i(\theta^{(n)})$ and χ_n for each n.

n	$c_i(\theta^{(n)})$	χ_n	n	$c_i(\theta^{(n)})$	χ_n	n	$c_i(\theta^{(n)})$	χ_n
1	7	0.0003	9	7	0.0626	17	4.6	0.0041
2	5	0.0072	10	7	0.0802	18	2.6	0.0008
3	5	0.0237	11	8	0.1025	19	8	0.001
4	5	0.0242	12	9.5	0.1445	20	5	0.0005
5	5	0.0222	13	10	0.1909	21	6.4	0.0013
6	4	0.0301	14	8.5	0.1162	22	5.2	0.0031
7	5	0.0397	15	8.5	0.0793	23	4.5	0.0085
8	6	0.0268	16	6.5	0.0082	24	3.9	0.0222

Moreover, $c_i'(\theta^{(n)}) = c_i(\theta^{(n)})(1 - VD_i)$,

where $VD_i = 2a \sqrt{\sum_{j=1, j\neq i}^{N} \frac{1}{\left(1 + \frac{\kappa}{R} |(x_j - x_i)cos(\theta^{(n)}) + (y_j - y_i)sin(\theta^{(n)})|\right)^4}}$,

$a = 0.5 \cdot (1 - \sqrt{1 - C_T})$, $C_T = 0.8$ and $\kappa = 0.01$.

A.7 Problem I1 [19,55]

This is the problem G01 with integrality constraints on the variables.

$$\begin{cases} \min \quad 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i \\ \text{subject to} \\ 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0 \\ 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0 \\ 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0 \\ -8x_1 + x_{10} \leq 0 \\ -8x_2 + x_{11} \leq 0 \\ -8x_3 + x_{12} \leq 0 \\ -2x_4 - x_5 + x_{10} \leq 0 \\ -2x_6 - x_7 + x_{11} \leq 0 \\ -2x_8 - x_9 + x_{12} \leq 0 \\ x_i \in \{0, 1\},\ i = 1, 2, \ldots, 9, 13 \\ x_i \in \{0, 1, \ldots, 100\},\ i = 10, 11, 12 \end{cases}$$

A.8 Problem I2 [13,55]: *hmittelman*

The binary nonlinear problem hmittelman.

$$\begin{cases} \min \quad 10y_1 + 7y_2 + y_3 + 12y_4 + 8y_5 + 3y_6 + y_7 + 5y_8 + 3y_9 \\ \text{subject to} \\ 3y_1 - 12y_2 - 8y_3 + y_4 - 7y_9 + 2y_{10} + 2 \leq 0 \\ y_2 - 10y_3 - 5y_5 + y_6 + 7y_7 + y_8 + 1 \leq 0 \\ 5y_1 - 3y_2 - y_3 - 2y_8 + y_{10} + 1 \leq 0 \\ 3y_2 - 5y_1 + y_3 + 2y_8 - y_{10} - 1 \leq 0 \\ -4y_3 - 2y_4 - 5y_6 + y_7 - 9y_8 - 2y_9 + 3 \leq 0 \\ 9y_2 - 12y_4 - 7y_5 + 6y_6 + 2y_8 - 15y_9 + 3y_{10} + 7 \leq 0 \\ 5y_2 - 8y_1 + 2y_3 - 7y_4 - y_5 - 5y_7 - 10y_9 + 1 \leq 0 \\ y_1 = x_5x_7x_9x_{10}x_{14}x_{15}x_{16} \\ y_2 = x_1x_2x_3x_4x_8x_{11} \\ y_3 = x_3x_4x_6x_7x_8 \\ y_4 = x_3x_4x_8x_{11} \\ y_5 = x_6x_7x_8x_{12} \\ y_6 = x_6x_7x_9x_{14}x_{16} \\ y_7 = x_9x_{10}x_{14}x_{16} \\ y_8 = x_5x_{10}x_{14}x_{15}x_{16} \\ y_9 = x_1x_2x_{11}x_{12} \\ y_{10} = x_{13}x_{14}x_{15}x_{16} \\ x_i \in \{0,1\},\ i = 1,2,\ldots,16. \end{cases}$$

A.9 Problem I3 [19,55]

The G01 problem with integrality constraints and modified bounds.

$$\begin{cases} \min \quad 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i \\ \text{subject to} \\ 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0 \\ 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0 \\ 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0 \\ -8x_1 + x_{10} \leq 0 \\ -8x_2 + x_{11} \leq 0 \\ -8x_3 + x_{12} \leq 0 \\ -2x_4 - x_5 + x_{10} \leq 0 \\ -2x_6 - x_7 + x_{11} \leq 0 \\ -2x_8 - x_9 + x_{12} \leq 0 \\ x_i \in \{0,1,\ldots,100\},\ i = 1,2,\ldots,13. \end{cases}$$

A.10 Problem MI1 [9,54]

This is a modification of a reliability problem.

$$\begin{cases} \min \quad -x_5x_6x_7 \\ \text{subject to} \\ -x_8 - x_9 - x_{10} + 1 \le 0 \\ -x_1 - x_2 - x_{11} + 1 \le 0 \\ -x_3 - x_4 + 1 \le 0 \\ 2x_1 + x_2 + 3x_3 + 2x_4 + 3x_8 + x_9 + 2x_{10} + 3x_{11} - 10 \le 0 \\ -\log(1 - x_5) + \log(0.1)x_8 + \log(0.2)x_9 + \log(0.15)x_{10} \le 0 \\ \log(0.2)x_1 + \log(0.15)x_2 - \log(1 - x_6) + \log(0.05)x_{11} \le 0 \\ \log(0.02)x_3 + \log(0.06)x_4 - \log(1 - x_7) \le 0 \\ x_i \in [0, 1],\ i = 1, 2, 3, 4 \\ x_5 \in [0, 0.997],\ x_6 \in [0, 0.9985],\ x_7 \in [0, 0.9988] \\ x_i \in \{0, 1\}\ , i = 8, 9, 10, 11. \end{cases}$$

A.11 Problem MI2 [84,54]

This is a purely mathematical problem with linear and nonlinear constraints.

$$\begin{cases} \min(x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 - \log(x_4 + 1) + (x_5 - 1)^2 + (x_6 - 2)^2 + (x_7 - 3)^2 \\ \text{subject to} \\ x_8 + x_9 + x_{10} + x_5 + x_6 + x_7 - 5 \le 0 \\ x_3^2 + x_5^2 + x_6^2 + x_7^2 - 5.5 \le 0 \\ x_8 + x_5 - 1.2 \le 0 \\ x_9 + x_6 - 1.8 \le 0 \\ x_{10} + x_7 - 2.5 \le 0 \\ x_{11} + x_5 - 1.2 \le 0 \\ x_2^2 + x_6^2 - 1.64 \le 0 \\ x_3^2 + x_7^2 - 4.25 \le 0 \\ x_2^2 + x_7^2 - 4.64 \le 0 \\ x_1 - x_8 \le 0 \\ x_2 - x_9 \le 0 \\ x_3 - x_{10} \le 0 \\ x_4 - x_{11} \le 0 \\ x_i \in [0, 1],\ i = 1, ..., 4 \\ x_i \in [0, 10],\ i = 5, ..., 7 \\ x_i \in \{0, 1\},\ i = 8, ..., 11 \end{cases}$$

A.12 Problem MI3 [41,54]: *bridge system*

It is a reliability-redundancy allocation problem on a bridge network.

$$\begin{cases} \min -R_1R_2 - R_3R_4 - R_1R_4R_5 - R_2R_3R_5 + R_1R_2R_3R_4 + R_1R_2R_3R_5 \\ +R_1R_2R_4R_5 + R_1R_3R_4R_5 + R_2R_3R_4R_5 - 2R_1R_2R_3R_4R_5 \\ \text{subject to} \\ \sum_{i=1}^{5} p_i u_i^2 - 110 \le 0 \\ \sum_{i=1}^{5} (\alpha_i (\frac{-1000}{\log(x_i)})^{1.5})(u_i + e^{\frac{u_i}{4}}) - 175 \le 0 \\ \sum_{i=1}^{5} \omega_i u_i e^{\frac{u_i}{4}} - 200 \le 0, \end{cases}$$

where $R_i = 1 - (1 - x_i)^{u_i}$, $0 \le x_i \le 1 - 10^{-6}$, and $u_i \in \{1, 2, \ldots, 10\}, \forall i = 1, 2, \ldots, 5$. Besides, $\alpha = (2.330, 1.450, 0.541, 8.050, 1.950) \cdot 10^{-5}$, $p = (1, 2, 3, 4, 2)$ and $\omega = (7, 8, 8, 6, 9)$.

A.13 Problem MI4 [41,54]: *series-parallel system*

This is a reliability-redundancy allocation problem on a series-parallel system.

$$\begin{cases} \min -1 + (1 - R_1R_2)(1 - (1 - R_3)(1 - R_4)R_5)) \\ \text{subject to} \\ \sum_{i=1}^{5} p_i u_i^2 - 180 \leq 0 \\ \sum_{i=1}^{5} (\alpha_i (\frac{-1000}{\log(x_i)})^{1.5})(u_i + e^{\frac{u_i}{4}}) - 175 \leq 0 \\ \sum_{i=1}^{5} \omega_i u_i e^{\frac{u_i}{4}} - 100 \leq 0, \end{cases}$$

where $R_i = 1 - (1 - x_i)^{u_i}$, $0 \leq x_i \leq 1 - 10^{-6}$, and $u_i \in \{1, 2, \ldots, 10\}, \forall i = 1, 2, \ldots, 5$. Parameters: $\alpha = (2.5, 1.45, 0.541, 0.541, 2.1) \cdot 10^{-5}$, $p = (2, 4, 5, 8, 4)$ and $\omega = (3.5, 4, 4, 3.5, 4.5)$.

A.14 Problem MI5 [25,40]: *multi-product batch plant*

This problem aims at designing a multi-product batch plant with 3 serial batch processing stages manufacturing 2 different products.

$$\begin{cases} \min 250 \sum_{j=1}^{3} N_j V_j^{0.6} \\ \text{subject to} \\ 2B_1 + 4B_2 - V_1 \leq 0 \\ 3B_1 + 6B_2 - V_2 \leq 0 \\ 4B_1 + 3B_2 - V_3 \leq 0 \\ \frac{40000 T_{L1}}{B_1} + \frac{20000 T_{L2}}{B_2} - 6000 \leq 0 \\ 8 - N_1 T_{L1} \leq 0 \\ 20 - N_2 T_{L1} \leq 0 \\ 8 - N_3 T_{L1} \leq 0 \\ 16 - N_1 T_{L2} \leq 0 \\ 4 - N_2 T_{L2} \leq 0 \\ 4 - N_3 T_{L2} \leq 0, \end{cases}$$

where $1 \leq N_1, N_2, N_3 \leq 3$, $250 \leq V_1, V_2, V_3 \leq 2500$, $\frac{20}{3} \leq T_{L1}, T_{L2} \leq 20$, $\frac{20}{3} T_{L1} \leq B_1 \leq 625$ and $\frac{10}{3} T_{L2} \leq B_2 \leq \frac{1250}{3}$.

A.15 Problem MI6 [27,40]: *rolling-element bearing*

This problem aims at optimizing the internal geometry of a rolling bearing.

$$\begin{cases} \min \begin{cases} f_c z^{2/3} d_b^{1.8} & \text{if } d_b \leq 25.4 \\ 3.647 f_c z^{2/3} d_b^{1.4} & \text{otherwise} \end{cases} \\ \text{subject to} \\ z - \frac{\Phi_0}{2 \arcsin(d_b / d_m)} - 1 \leq 0 \\ k_{Dmin}(D - d) - 2d_b \leq 0 \\ 2d_b - k_{Dmax}(D - d) \leq 0 \\ \zeta B_w - d_b \leq 0 \\ 0.5(D + d) - d_m \leq 0 \\ d_m - (0.5 + e)(D + d) \leq 0 \\ \epsilon d_b - 0.5(D - d_m - d_b) \leq 0 \\ 0.515 - f_i \leq 0 \\ 0.515 - f_0 \leq 0, \end{cases}$$

where $0.5(D + d) \leq d_m \leq 0.6(D + d)$, $0.15(D - d) \leq d_b \leq 0.45(D - d)$, $4 \leq z \leq 50$, $0.515 \leq f_i, f_0 \leq 0.6$, $0.4 \leq k_{Dmin} \leq 0.5$, $0.6 \leq k_{Dmax} \leq 0.7$, $0.3 \leq \epsilon \leq 0.4$, $0.02 \leq e \leq 0.1$ and $0.6 \leq \zeta \leq 0.85$.

Besides, $f_c = 37.91(1 + (1.04(\frac{1-\gamma}{1+\gamma})^{1.72}(\frac{f_i(2f_0-1)}{f_0(2f_i-1)})^{0.41})^{10/3})^{-0.3}$, $\gamma = \frac{d_b}{d_m}$, $f_i = \frac{r_i}{d_b}$, $f_0 = \frac{r_0}{d_b}$, $\Phi_0 = 2\pi - 2\arccos(\frac{(\frac{D-d}{2} - 3\frac{T}{4})^2 + (\frac{D}{2} - \frac{T}{4} - d_b)^2 - (\frac{d}{2} + \frac{T}{4})^2}{2(\frac{D-d}{2} - 3\frac{T}{4})(\frac{D}{2} - \frac{T}{4} - d_b)})$, $T = D - d - 2d_b$, $D = 160$, $d = 90$ and $B_w = 30$.

A.16 Problem MV2[30,15]

This is the well-known G07 problem where some variables are imposed to be discrete.

$$\begin{cases} \min x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 \\ +2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45 \\ \text{subject to} \\ -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0 \\ 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0 \\ -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0 \\ 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0 \\ 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0 \\ x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leq 0 \\ 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leq 0 \\ -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0, \end{cases}$$

where $x_i \in \{-10, -5, 0, 1.3, 2.2, 5, 8.2, 8.7, 9.5, 10\}, \forall i = 1, 2, \ldots, 6$, $-10 \leq x_7, x_8 \leq 10$ and $x_9, x_{10} \in \{-10, -9, \ldots, 10\}$.

A.17 Problem MV3 [26,22]: *car side impact design*

This problem uses a simplified regression model of the finite element model of a car side impact. The aim is to minimize the weight of the car subject to nonlinear inequality constraints.

$$\begin{cases} \min 1.98 + 4.90x_1 + 6.67x_2 + 6.98x_3 + 4.01x_4 + 1.78x_5 + 2.73x_7 \\ \text{subject to} \\ 1.16 - 0.3717x_2x_4 - 0.00931x_2x_{10} - 0.484x_3x_9 + 0.01343x_6x_{10} - 1 \leq 0 \\ 0.261 - 0.0159x_1x_2 - 0.188x_1x_8 - 0.019x_2x_7 + 0.0144x_3x_5 + 0.0008757x_5x_{10} \\ +0.08045x_6x_9 + 0.00139x_8x_{11} + 0.00001575x_{10}x_{11} - 0.32 \leq 0 \\ 0.214 + 0.00817x_5 - 0.131x_1x_8 - 0.0704x_1x_9 + 0.03099x_2x_6 - 0.018x_2x_7 + 0.0208x_3x_8 \\ +0.121x_3x_9 - 0.00364x_5x_6 + 0.0007715x_5x_{10} - 0.0005354x_6x_{10} + 0.00121x_8x_{11} - 0.32 \leq 0 \\ 0.074 - 0.061x_2 - 0.163x_3x_8 + 0.001232x_3x_{10} - 0.166x_7x_9 + 0.227x_2^2 - 0.32 \leq 0 \\ 28.98 + 3.818x_3 - 4.2x_1x_2 + 0.0207x_5x_{10} + 6.63x_6x_9 - 7.7x_7x_8 + 0.32x_9x_{10} - 32 \leq 0 \\ 33.86 + 2.95x_3 + 0.1792x_{10} - 5.057x_1x_2 - 11.0x_2x_8 - 0.0215x_5x_{10} - 9.98x_7x_8 + 22.2x_8x_9 - 32 \leq 0 \\ 46.36 - 9.9x_2 - 12.9x_1x_8 + 0.1107x_3x_{10} - 32 \leq 0 \\ 4.72 - 0.5x_4 - 0.19x_2x_3 - 0.0122x_4x_{10} + 0.009325x_6x_{10} + 0.000191x_{11}^2 - 4 \leq 0 \\ 10.58 - 0.674x_1x_2 - 1.95x_2x_8 + 0.02054x_3x_{10} - 0.0198x_4x_{10} + 0.028x_6x_{10} - 9.9 \leq 0 \\ 16.45 - 0.489x_3x_7 - 0.843x_5x_6 + 0.0432x_9x_{10} - 0.0556x_9x_{11} - 0.000786x_{11}^2 - 15.7 \leq 0, \end{cases}$$

where $0.5 \leq x_1, x_3, x_4 \leq 1.5$, $0.45 \leq x_2 \leq 1.35$, $0.875 \leq x_5 \leq 2.625$, $0.4 \leq x_6, x_7 \leq 1.2$ $x_8, x_9 \in \{0.192, 0.345\}$, $-30 \leq x_{10}, x_{11} \leq 30$.

A.18 Problem MV4 [73,22]: *stepped cantilever beam design*

This problem minimizes the volume of a stepped cantilever beam.

$$\begin{cases} \min l(x_1x_2 + x_3x_4 + x_5x_6 + x_7x_8 + x_9x_{10}) \\ \text{subject to} \\ 6Pl - \sigma x_9x_{10}^2 \leq 0 \\ 6P2l - \sigma x_7x_8^2 \leq 0 \\ 6P3l - \sigma x_5x_6^2 \leq 0 \\ 6P4l - \sigma x_3x_4^2 \leq 0 \\ 6P5l - \sigma x_1x_2^2 \leq 0 \\ \frac{Pl^3}{E}(244x_3x_4^3x_5x_6^3x_7x_8^3x_9x_{10}^3 + \\ +148x_1x_2^3x_5x_6^3x_7x_8^3x_9x_{10}^3 + 76x_1x_2^3x_3x_4^3x_7x_8^3x_9x_{10}^3 + \\ +28x_1x_2^3x_3x_4^3x_5x_6^3x_9x_{10}^3 + 4x_1x_2^3x_3x_4^3x_5x_6^3x_7x_8^3) \\ -\delta x_1x_2^3x_3x_4^3x_5x_6^3x_7x_8^3x_9x_{10}^3 \leq 0 \\ x_2 - 20x_1 \leq 0 \\ x_4 - 20x_3 \leq 0 \\ x_6 - 20x_5 \leq 0 \\ x_8 - 20x_7 \leq 0 \\ x_{10} - 20x_9 \leq 0, \end{cases}$$

where $x_1 \in \{1, 2, \ldots, 5\}$, $x_2, x_4 \in \{45, 50, 55, 60\}$, $x_3, x_5 \in \{2.4, 2.6, 2.8, 3.1\}$, $x_6 \in \{30, 31, \ldots, 65\}$, $1 \leq x_7 \leq 5$, $30 \leq x_8, x_{10} \leq 65$, $1 \leq x_9 \leq 5$.

Besides, $P = 50000$, $l = 100$, $\delta = 2.7$, $\sigma = 14000$, $E = 2 \cdot 10^7$.

A.19 Problem MV1 [83,40]: *four-stage gear box problem*

The problem minimizes the weight of a gear box where all variables are discrete.

Let $c_i = \sqrt{(y_{gi} - y_{pi})^2 + (x_{gi} - x_{pi})^2}$, $K_0 = 1.5$, $d_{\min} = 25$, $J_R = 0.2$, $\phi = 120$, $W = 55.9$, $K_m = 1.6$, $CR_{\min} = 1.4$, $L_{\max} = 127$, $C_p = 464$, $\sigma_H = 3290$, $\omega_{\max} = 255$, $\omega_1 = 5000$, $\sigma_N = 2090$ and $\omega_{\min} = 245$,

$$
\begin{cases}
\min\left(\frac{\pi}{1000}\right)\sum_{i=1}^{4}\frac{b_i c_i^2(N_{pi}^2+N_{gi}^2)}{(N_{pi}+N_{gi})^2}\\
\text{subject to}\\
\left(\frac{366000}{\pi\omega_1}+\frac{2c_1N_{p1}}{N_{p1}+N_{g1}}\right)\left(\frac{(N_{p1}+N_{g1})^2}{4b_1c_1^2N_{p1}}\right)-\frac{\sigma_N J_R}{0.0167WK_0K_m}\le 0\\
\left(\frac{366000N_{g1}}{\pi\omega_1N_{p1}}+\frac{2c_2N_{p2}}{N_{p2}+N_{g2}}\right)\left(\frac{(N_{p2}+N_{g2})^2}{4b_2c_2^2N_{p2}}\right)-\frac{\sigma_N J_R}{0.0167WK_0K_m}\le 0\\
\left(\frac{366000N_{g1}N_{g2}}{\pi\omega_1N_{p1}N_{p2}}+\frac{2c_3N_{p3}}{N_{p3}+N_{g3}}\right)\left(\frac{(N_{p3}+N_{g3})^2}{4b_3c_3^2N_{p3}}\right)-\frac{\sigma_N J_R}{0.0167WK_0K_m}\le 0\\
\left(\frac{366000N_{g1}N_{g2}N_{g3}}{\pi\omega_1N_{p1}N_{p2}N_{p3}}+\frac{2c_4N_{p4}}{N_{p4}+N_{g4}}\right)\left(\frac{(N_{p4}+N_{g4})^2}{4b_4c_4^2N_{p4}}\right)-\frac{\sigma_N J_R}{0.0167WK_0K_m}\le 0\\
\left(\frac{366000}{\pi\omega_1}+\frac{2c_1N_{p1}}{N_{p1}+N_{g1}}\right)\left(\frac{(N_{p1}+N_{g1})^3}{4b_1c_1^2N_{g1}N_{p1}^2}\right)-\left(\frac{\sigma_H}{C_p}\right)^2\left(\frac{\sin(\phi)\cos(\phi)}{0.0334WK_0K_m}\right)\le 0\\
\left(\frac{366000N_{g1}}{\pi\omega_1N_{p1}}+\frac{2c_2N_{p2}}{N_{p2}+N_{g2}}\right)\left(\frac{(N_{p2}+N_{g2})^3}{4b_2c_2^2N_{g2}N_{p2}^2}\right)-\left(\frac{\sigma_H}{C_p}\right)^2\left(\frac{\sin(\phi)\cos(\phi)}{0.0334WK_0K_m}\right)\le 0\\
\left(\frac{366000N_{g1}N_{g2}}{\pi\omega_1N_{p1}N_{p2}}+\frac{2c_3N_{p3}}{N_{p3}+N_{g3}}\right)\left(\frac{(N_{p3}+N_{g3})^3}{4b_3c_3^2N_{g3}N_{p3}^2}\right)-\left(\frac{\sigma_H}{C_p}\right)^2\left(\frac{\sin(\phi)\cos(\phi)}{0.0334WK_0K_m}\right)\le 0\\
\left(\frac{366000N_{g1}N_{g2}N_{g3}}{\pi\omega_1N_{p1}N_{p2}N_{p3}}+\frac{2c_4N_{p4}}{N_{p4}+N_{g4}}\right)\left(\frac{(N_{p4}+N_{g4})^3}{4b_4c_4^2N_{g4}N_{p4}^2}\right)-\left(\frac{\sigma_H}{C_p}\right)^2\left(\frac{\sin(\phi)\cos(\phi)}{0.0334WK_0K_m}\right)\le 0\\
-N_{pi}\sqrt{\frac{sin^2(\phi)}{4}-\frac{1}{N_{pi}}+\left(\frac{1}{N_{pi}}\right)^2}+N_{gi}\sqrt{\frac{sin^2(\phi)}{4}+\frac{1}{N_{gi}}+\left(\frac{1}{N_{gi}}\right)^2}+\frac{\sin(\phi)(N_{pi}+N_{gi})}{2}\\
\qquad +CR_{\min}\pi\cos(\phi)\le 0,\forall i\in\{1,2,3,4\}\\
d_{\min}-\frac{2c_iN_{pi}}{N_{pi}+N_{gi}}\le 0,\forall i\in\{1,2,3,4\}\\
d_{\min}-\frac{2c_iN_{gi}}{N_{pi}+N_{gi}}\le 0,\forall i\in\{1,2,3,4\}\\
x_{p1}+\frac{(N_{p1}+2)c_1}{N_{p1}+N_{g1}}-L_{\max}\le 0\\
-L_{\max}+\frac{(N_{pi}+2)c_i}{N_{pi}+N_{gi}}+x_{g(i-1)}\le 0,\forall i\in\{2,3,4\}\\
-x_{p1}+\frac{(N_{p1}+2)c_1}{N_{p1}+N_{g1}}\le 0\\
\frac{(N_{pi}+2)c_i}{N_{pi}+N_{gi}}-x_{g(i-1)}\le 0,\forall i\in\{2,3,4\}\\
y_{p1}+\frac{(N_{p1}+2)c_1}{N_{p1}+N_{g1}}-L_{\max}\le 0\\
-L_{\max}+\frac{(N_{pi}+2)c_i}{N_{pi}+N_{gi}}+y_{g(i-1)}\le 0,\forall i\in\{2,3,4\}\\
\frac{(N_{p1}+2)c_1}{N_{p1}+N_{g1}}-y_{p1}\le 0\\
\frac{(N_{pi}+2)c_i}{N_{pi}+N_{gi}}-y_{g(i-1)}\le 0,\forall i\in\{2,3,4\}\\
-L_{\max}+\frac{(N_{gi}+2)c_i}{N_{pi}+N_{gi}}+x_{gi}\le 0,\forall i\in\{1,2,3,4\}\\
-x_{gi}\frac{(N_{gi}+2)c_i}{N_{pi}+N_{gi}}\le 0,\forall i\in\{1,2,3,4\}\\
y_{gi}+\frac{(N_{gi}+2)c_i}{N_{pi}+N_{gi}}-L_{\max}\le 0,\forall i\in\{1,2,3,4\}\\
-y_{gi}+\frac{(N_{gi}+2)c_i}{N_{pi}+N_{gi}}\le 0,\forall i\in\{1,2,3,4\}\\
-(b_i-8.255)(b_i-5.715)(b_i-12.70)(-N_{pi}+0.945c_i-N_{gi})\le 0,\forall i\in\{1,2,3,4\}\\
(b_i-8.255)(b_i-3.175)(b_i-12.70)(-N_{pi}+0.646c_i-N_{gi})\le 0,\forall i\in\{1,2,3,4\}\\
(b_i-5.715)(b_i-3.175)(b_i-12.70)(-N_{pi}+0.504c_i-N_{gi})\le 0,\forall i\in\{1,2,3,4\}\\
(b_i-5.715)(b_i-3.175)(b_i-8.255)(-N_{pi}-N_{gi})\le 0,\forall i\in\{1,2,3,4\}\\
-(b_i-8.255)(b_i-5.715)(b_i-12.70)(N_{pi}-1.812c_i+N_{gi})\le 0,\forall i\in\{1,2,3,4\}\\
(b_i-8.255)(b_i-3.175)(b_i-12.70)(N_{pi}-0.945c_i+N_{gi})\le 0,\forall i\in\{1,2,3,4\}\\
-(b_i-5.715)(b_i-3.175)(b_i-12.70)(N_{pi}-0.646c_i+N_{gi})\le 0,\forall i\in\{1,2,3,4\}\\
(b_i-5.715)(b_i-3.175)(b_i-8.255)(N_{pi}-0.504c_i+N_{gi})\le 0,\forall i\in\{1,2,3,4\}\\
\omega_{\min}-\frac{\omega_1(N_{p1}N_{p2}N_{p3}N_{p4})}{(N_{g1}N_{g2}N_{g3}N_{g4})}\le 0\\
\frac{\omega_1(N_{p1}N_{p2}N_{p3}N_{p4})}{(N_{g1}N_{g2}N_{g3}N_{g4})}-\omega_{\max}\le 0.
\end{cases}
$$