Nonlinear optimization of mixed continuous and discrete variables for black-box simulators

Thi Thoi Tran

École Doctorale n° 475
Mathématiques, Informatique, Télécommunications de Toulouse

Nonlinear optimization of mixed continuous and discrete variables for black-box simulators

Submitted by
Thi Thoi TRAN

PH.D. MID-TERM EVALUATION

on
MARCH 05, 2020

based on a work conducted at

under the supervision of

Marcel MONGEAU	Professeur, ENAC, Directeur de thèse
Delphine SINOQUET	Ingénieur de recherche, IFPEN, Promotrice
Sébastien DA VEIGA	Ingénieur de recherche, Safran Tech, Promoteur

Contents

Abstract

Black-box Mixed-integer Nonlinear Problems (MINLP) are optimization problems whose variables can take integer (or discrete) and continuous values, and the objective function (and possibly constraints) are performed as output of "black-box" simulations. Generally, black-box MINLP optimization problems are considered as particularly complex problems due to the combinatorial aspect and the lack of information about the cost function and the constraints.

In recent years, there has been a considerable amount of real industrial applications that involve mixed variables and time-consuming simulators, e.g., at Safran Tech and IFPEN, optimal designs of engine turbine in aircraft, of mooring lines of offshore wind turbines, of electric engine stators and rotors,... In these nonlinear optimization problems, derivatives of the objective function (and, possibly of the constraints) are not available and cannot be directly approximated. Another difficulty is that minimization of the cost involves complex variables, a varying number of components (integer variables), different materials (categorical variables, usually non-ordered), the presence or not of some components (binary variables) and continuous variables describing dimensions/characteristics of the structure pieces.

On this type of applications, even if the cost function can be relatively simple (e.g., simple relationship between the parameters to be optimized and the cost of the structure), some physical constraints are introduced to satisfy accurate specifications (reliability, operating, dimensioning constraints ...). These constraints often result from complex computations of a numerical simulator. It thus leads to an optimization problem involving one or more simulators that are often computationally expensive "black-box" (from closed commercial software or too complex simulator for a possible extraction of any information on the optimization variables).

In this work, we focus on derivative free optimization methods (DFO) [7, 15]. Among DFO methods, we consider two families: direct search methods (e.g., pattern search or Nelder Mead simplex) and surrogate optimization methods, especially trust-region methods based on simple interpolating or regression models (linear or quadratic). Convergence results to local minima are proved for such method but require adaptations to converge to a global optimal solution (e.g., a multi-start approach with several initial points). The direct search methods require a large number of simulations, whereas the methods of the second family are generally more efficient to converge to a local solution. The methods of the two families should be extended to mixed continuous and discrete variables constrained optimization (simple constraints and black-box constraints) in order to address the presented applications.

Keywords: Mixed integer non-linear programming, derivative free optimization, black-box simulation, necklace distance

Chapter 1

Introduction

Subject of the study

The optimization problem we first address is in the form of a Mixed Binary Non-linear Programming (MBNLP):

$$\begin{cases} \min\limits_{x,y} f_b(x,y) \\ x \in [x^L, x^U], y \in \{0,1\}^n \\ g(x,y) \leq 0 \\ g_b(x,y) \leq 0, \end{cases} \tag{1.1}$$

where the objective function f_b and the constraints functions g_b are outputs of "black-box" simulator, and g are given explicitly.
This report is concerned with problems with a "black-box" functions $f_b, g_b : \mathbb{R}^n \to \mathbb{R}$ with the two following characteristics

1. The derivatives $\nabla f_b, \nabla g_b$ are not available, we only have access to the function values,
2. It is computationally expensive to evaluate the functions f_b and g_b.

These problems are in general NP-hard and difficult to solve. MINLP or MBNLP are typically solved by branch and bound methods or meta-heuristic algorithms. It is generally meaningless to use relaxation techniques for discrete variables (e.g., for categorical variables). In most cases, the problem is not convex and the simulator cannot evaluate the black-box functions at fractional values of the discrete variables. In addition, the feasible set can be very thin and thus finding a feasible point becomes very difficult.

Derivative free methods dedicated to mixed continuous and discrete variables (with and without constraints) have been studied in the literature, in which we can point out some prominent articles, [4, 6, 33, 34]. Some well-known methods are for example pattern search methods, mesh adaptive direct search, filter pattern search methods, etc. One main idea is to alternatively do a continuous poll with the discrete variables held fixed, a discrete poll with the continuous variables held fixed and an extended poll in mixed domain. The extended poll step performs a continuous poll around each point found during the discrete poll whose objective function value lies within some user defined threshold of the best known objective value encountered so far.

Another class of methods, trust region methods, are described in [15]. In this book, the authors prove the local convergence based on the geometry of the interpolation set.

In this study, we focus on adapting DFO trust-region method to a black-box MIMLP application coming from Safran, more precisely, the optimization of blade shapes of engine turbines in

aircraft, and arranging optimally these optimal shapes on the disk of the turbine. The values of the objective function are computed by black-box mechanical simulators which can cost several hours of computation or even more. Therefore, the priority is not only to find a 'good' solution, but also to save simulations.

Structure of the report

We begin in Chapter 2 by introducing notations and reminding some necessary mathematical background. We also present the application from the aeronautics industry that motivates our study. In Chapter 3, an overview of trust-region methods and derivative-free trust-region methods are presented. In particular, we emphasize some important remarks about the model requirements based on the geometry of the interpolation set. We also show how to deal with the binary variables in this method. Chapter 4 focuses on the adaptation of the algorithm to the application: we introduce the concept of necklace from the combinatorics literature and propose an adapted distance for this type of variable configuration. The convergence properties of the adapted method are also developed. The first numerical results are given in the Chapter 5 for benchmark functions and for Safran's application. The last chapter discuss the conclusions and present perspectives.

Chapter 2

Notation and reminders

We begin by introducing the notations used in this report: vector and matrix norms, condition number, quadratic programming and mixed integer quadratic programming.

Vector and matrix norms

For a vector $x \in \mathbb{R}^n$ $p-$norms ($p \geq 1$) are given by

$$\|x\|_p = \Big(\sum_{i=1}^{n} |x_i|^p \Big)^{\frac{1}{p}},$$

and the $\infty-$norm by

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|.$$

Similarly, we define the $p-$norm of matrices. Let $A = (a_{ij})_{m \times n}$ be a $m \times n$ matrix. The $p-$norms, $p \geq 1$, are defined by

$$\|A\|_p = \sup_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p},$$

the $\infty-$norm by

$$\|A\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}|,$$

and the Frobenius norm by

$$\|A\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^2}.$$

Besides satisfying the three norm properties(positive definite, scalar multiply, triangle inequality), they also satisfy the "submultiplicative" property

$$\|AB\| \leq \|A\| \|B\|.$$

There are several useful inequalities for matrix norms, namely

$$\frac{1}{\sqrt{m}} \|A\|_1 \leq \|A\|_2 \leq \|A\|_F \leq \sqrt{n} \|A\|_2 \leq n \|A\|_1,$$

and for vector norms

$$\|.\|_\infty \leq \|.\|_2 \leq \sqrt{n} \|.\|_\infty.$$

Conditioning

One of the most important properties we use to study a matrix is its condition number which gives us the information as to whether a matrix is ill-conditioned or well-conditioned. The condition number of a $n \times n$ non-singular matrix A is given by

$$\chi(A) = \|A\|.\|A^{-1}\|,$$

which depends on the chosen norm. The value of the condition number has a large impact on the difficulty of solving a linear system: the system is easier to solve if its condition number is small (ideally $\chi(A)$ equal to 1) and it becomes difficult or unsolvable if the condition number is going to infinity. In particular, even a small perturbation in the right-hand side of a linear of equations can yield to an enormous change in the solution of this when the matrix is ill-conditioned. If the matrix A is symmetric, there is another way to see the condition number, which is

$$\chi(A) = \frac{\max_{1\leq i\leq n} |\alpha_i|}{\min_{1\leq i\leq n} |\alpha_i|},$$

where $\alpha_1, \ldots, \alpha_n$ are the eigenvalues of A.

Quadratic programming and mixed integer quadratic programming

Quadratic programming (QP)
Quadratic programming is widely studied in the literature as it is often a subproblem for optimization algorithms. An optimization problem with a quadratic objective function and linear constraints is called a quadratic program, which can be stated as

$$\begin{cases} \min\limits_{x} \dfrac{1}{2} x^T H x + x^T g \\ a_i^T x = b_i,\ i \in \mathcal{E}, \\ a_i^T x \geq b_i,\ i \in \mathcal{I}, \end{cases}$$

where H is a symmetric matrix, $\mathcal{E}$ and $\mathcal{I}$ are index sets associated with equality and inequality constraints. If the Hessian H is semi positive definite (SDP), the problem is said to be a convex QP, otherwise it is a non-convex problem. Solving non-convex problems is more challenging than solving convex problems as non-convex QPs are NP-hard problems. We can solve QP by using Interior-Point Methods or Active set methods, see [39].

Mixed Integer Quadratic Programming (MIQP)
MIQP is in the form of

$$\begin{cases} \min\limits_{x,z} f(x,z), \\ x \in \mathbb{R}^m, z \in \mathbb{Z}^n, \\ g(x,z) \leq 0. \end{cases} \tag{2.1}$$

where f, and g are quadratic and linear functions, respectively. To solve MIQP, the most common class of methods is Branch-and-bound algorithms and their extended versions, such as spatial branch-and-bound, branch-and-reduce, α branch-and-bound (for details see [13, 43]).

Chapter 3

Design of turbine blades for helicopter engine application

In this thesis, one of the motivating applications is the optimal design of the turbine blades of helicopter engine. The objective is to maximize the compressor efficiency under some stability constraints (minimal vibrations). This involves very expensive mechanical simulations (solid and fluid mechanics), typically, involving computer runs of the the order of several hours for evaluating one design configuration set. Some continuous parameters describe the blade shapes, e.g., the thickness, length ... of the blades, and binary variables locate the different types of blade geometries that are considered. This problem thus falls into the MINLP class.

This chapter distributes as follows: first, we introduce Safran's application context, then we describe the idea of reduced order model which is used in practice.

3.1 Application context

Motivation

Air traffic is one of the most important means of transport nowadays, especially in Europe. There are around 8000 daily flights. Moreover, the amount of people travelling by airplane increases every year by around 5 %. Air traffic is associated with very high costs of fuel [2, 3], and also with a huge budget for the maintenance and manufacturing of the new engines. Table 3.1 shows the fuel consumption for aircraft: if we can reduce of 0.5% the amount of fuel, we can gain around 410 million US $ per year. Therefore, reducing fuel consumption (by increasing engine efficiency) and maintenance costs (by decreasing vibrations) are two major concerns of the aeronautics industry.

Table 3.1: Illustration of fuel consummation for aircraft

World fuel consumption (litres)	Corresponding price (US $)
$\backsim 240.10^{12}$	$\backsim 83.10^{9}$
$0.5\% \downarrow$	$410.10^{6} \downarrow$

Turbomachines

There are several ways to optimize the costs in aviation: through optimizing trajectories, arrangements of passengers (through design of seat arrangement), cargo-storage, etc. Our study concentrates on optimizing the design of turbomachines by maximizing the efficiency (compressor) and by minimizing vibrations. Turbomachines or gas turbines are complex systems that are used in the aerospace, automotive and power generation industries. Blades are important components that allow the exchange of energy with the flow, Figure (3.1). During the operation of the turbomachine, vibrations occur, mainly due to the excitation by modification of the aerodynamic flow and also resulting from a coupling between the flow and the movement of the blades [37].

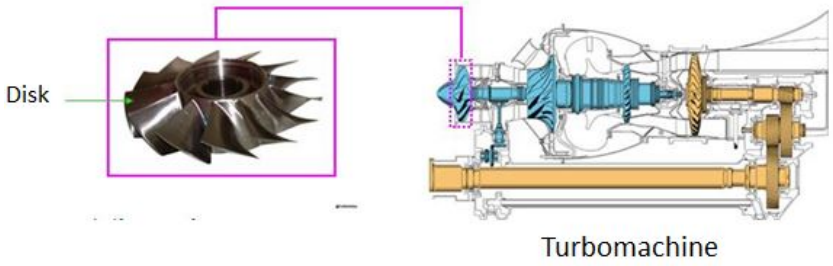

Figure 3.1: Illustration of a turbomachine from [37]

In the concrete application proposed by SAFRAN, the main idea is to optimize the so-called *tuning* and *mistuning* blade shapes in engine turbine and to find the optimal distribution of the resulting two shapes on the disk. The continuous variables are shape parameters such as the thickness of the axis length of the blades. A binary variable is associated to each blade: it takes value 0 for the reference shape, and value 1 for the other pre-defined shape (mistuning shape). Binary variables are used to locate these reference blade shapes on the disk, Figure (3.2).

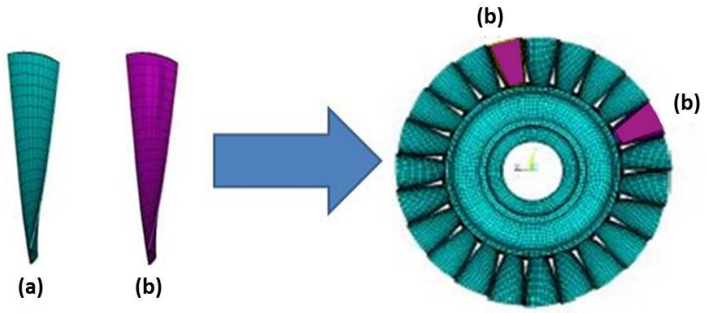

Figure 3.2: An example of blade configurations for two different shapes from [37]

Remark that the cyclic symmetry property of the problem yields to a huge number of redundant arrangements (see Tables 3.2). Due to the high cost of evaluating the simulations, we do not want to recompute equivalent (rotated) configurations.

In the next subsection we present the Reduced Order Model (ROM) which is used in practice to reduce the computational cost of optimization for such highly combinatorial problems.

Table 3.2: Number of distinct solutions for n blades. [37]

Total number of blades on the disk	Number of distinct arrangements	Total number of arrangements
2	3	2
3	4	3
6	14	64
12	352	4096
20	52488	1048576

3.2 Reduced Order Model (ROM)

Two blade disks that differ only by a rotation of the pattern around the disk lead to the same value of the forced response on the disk. Such arrangements should obviously be considered as a same solution. Note that when the number of blades is increasing, the number of such equivalent solutions rapidly increases (Table 3.2).

For an instance of our problem involving n blades, we can roughly approximate the number of distinct arrangements by $\frac{2^n}{n}$. Reducing these redundancies will significantly reduce the combinatorial complexity of the optimization problem and, consequently the computational cost of the envisaged optimization approach. Therefore, the methodology based on reduced-order modeling method is proposed.

Suppose we have two types of blades, A and B, that need to be located at n locations on the disk. In [49], the authors present a physical discussion about the problem of redundancy: in the presence of strong coupling, i.e., if there are few switches or if there exists a series of groupings of two consecutive blades of the form AA or BB (e.g., $7B5A$), the vibration responses between blades tend to be globally uniform at the wheel. While, in the case of low coupling, i.e., if there are many switches or if the basic grouping forming the pattern are no longer AA and BB but rather AB and BA (e.g., $1A1B2A2B$, the responses tend to be located on some blade neighbors.

Following the previous comments about inter-blade coupling level, if the optimal distribution presents few alternations of the two different shapes, it can be roughly reconstructed by a distribution of patterns AA or BB. On the other hand, if the optimal distribution presents many alternations of the two different shapes, we use patterns AB or BA. In both cases, the idea is to consider the distribution by groups of two blades in view of obtaining a model of lower dimension. Indeed, the idea of ROMT is to consider $n/2$ sectors of two blades with two possibilities for each sector. We group either patterns AA and BB if there are few switches (high coupling) or AB and BA in case there are many switches (low coupling). For illustration, a disk with 10 blades consists of 108 distinct distributions of two types of blades, whereas ROM leads to a problem of a disk with 5 sectors and only 8 distinct distributions.

The ROM optimization methodology consists of two main steps. It first optimizes in the two reduced spaces (for patterns AA/BB and then for patterns AB/BA considering on sectors of 2 blades) to provide initial guesses.Then, a local search limited to one flip from these initial guesses provides "local" solutions to the original problem.

ROM reduces the size of discrete space, which tends to reduce the computations in each sub-problem. A disadvantage however is that it removes a large number of possibilities which

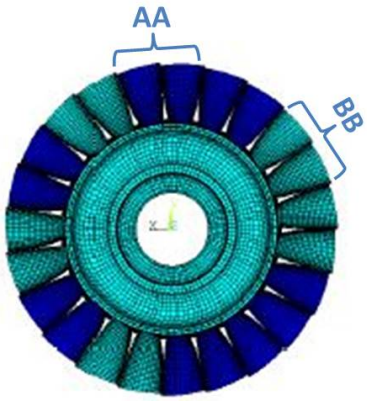

Figure 3.3: Illustration of a case with 23 blades with patterns $AA - BB$, from [37]

will not be explored. For instance, for the problem involving $n = 12$ blades, ROM removes 324 (each subproblem keeps 14 distinct solutions) in over 352 distinct solutions. This motivates our study to look for another method to address the above-mentioned cyclic symmetry property while ensuring that all the distinct arrangements are considered. In Chapter 5, we will propose an adapted method for this type of problem.

Chapter 4

Derivative-free optimization

In this chapter, we address the unconstrained problem of the form

$$\begin{cases} \min\limits_{x,y} f(x,y) \\ x \in [x^L, x^U], y \in \{0,1\}^n \end{cases} \tag{4.1}$$

where $x \in \mathbb{R}^m, y \in \{0,1\}^n$ are continuous and binary variables, respectively. The objective function f is the output of a "black-box" numerical simulator. We further assume that f is smooth and bounded from below to guarantee convergence of the algorithm, see ([7, 15]). We also assume that derivatives of f are unavailable. Derivative-free methods can be classified as direct search methods (pattern search methods, simplex search methods ...) and model-based methods (derivative free trust-region methods, surrogate optimization methods). In this report, we focus on derivative free trust region method.
We now begin with the useful background on trust-region framework basics. We first outline derivative-free trust-region methods and then discuss their extension to mixed binary problems.

4.1 Trust-region framework basics

We begin by providing a review of the trust-region framework for continuous optimization when derivatives of f are available, see [39] for details. In trust-region methods, at each iteration k, we build a quadratic model m_k around the current iterate x_k. This model is assumed to approximate the objective function sufficiently well in a neighbourhood of center x_k called the trust region, which is defined based on the center and radius pair $(x_k, \Delta_k > 0)$

$$B(x_k, \Delta_k) = \{x \in \mathbb{R}^n : \|x - x_k\|_k \leq \Delta_k\}.$$

The trust region norm $\|.\|_k$ can be taken from the standard $2-$norm $\|.\|_2$. To find the value of the next iterate x_{k+1}, we solve a quadratic sub-problem of the form

$$\min_{s \in B(0,\Delta_k)} m_k(x_k + s), \tag{4.2}$$

where $m_k(x_k + s) = f(x_k) + g_k^T s + \frac{1}{2} s^T H_k s$, g_k is the gradient of f at x_k, H_k is a symmetric approximation of the Hessian of f at x_k. The approximate solution s_k should satisfy the condition

$$m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa_d}{2} \|g_k\|_k \min\{\frac{\|g_k\|_k}{\|H_k\|_k}, \Delta_k\},$$

where $\kappa_d \in (0, 1]$ is a constant.

Taylor's theorem ensures the existence of a trust region that ensures previous condition, but does not give its precise size. Therefore, updating and adjusting the trust region radius after each iteration is necessary. Given an approximate solution s_k of (4.2), the trust region radius is updated depending on the ratio of actual improvement and predicted improvement

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}.$$

If the model reduction matches well the actual reduction of the objective function (when $\rho > 0$ and $\backsim 1$), the candidate s_k is accepted and the trust region radius is possibly increased. Otherwise, the candidate is rejected and the trust region radius is decreased. We run the loop until convergence criteria on minimal gradient norm and minimal trust region radius are reached.

Algorithm 1: Derivative based trust region algorithm, see [39]

Input: $x_0, 0 < \Delta_0 \leq \Delta_{max}, 0 < \gamma_0 < 1 < \gamma_1, 0 \leq \eta_0 \leq \eta_1 \leq 1$

0. Initialization. Compute $f(x_0), \triangledown f(x_0), k = 0$
1. Model definition. Build quadratic model m_k in $B(x_k, \Delta_k)$
2. Solve sub-problem (4.2)
$$s_k = \underset{s \in B(x_k, \Delta_k)}{\operatorname{argmin}} \; m_k(x_k + s)$$
3. Center update. Compute
$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}.$$
$$x_{k+1} = \begin{cases} x_k + s_k & \text{if } \rho_k \geq \eta_0, \\ x_k & \text{if } \rho_k < \eta_0. \end{cases}$$
4. Trust region update
$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{max}\} & \text{if } \rho_k \geq \eta_1 \\ \Delta_k & if\ \eta_0 \leq \rho_k < \eta_1, \\ \gamma_0 \Delta_k & \text{if } \rho_k < \eta_0. \end{cases}$$

When the derivatives are not available, we build on interpolation model based on a set of given simulations (f values). In this case, the model is considered as a valid approximation of the objective function under some conditions which mainly depend on the geometry of the interpolation set. If the model does not satisfy these conditions, a new point is added to improve the accuracy of the model. This is detailed in the next section.

4.2 Derivative-free trust region methods for continuous problems

The basic idea of DFO trust region methods [15] is to replace the problem, which involves expensive simulations with no information about the derivatives, by a simpler problem (linear or quadratic form) for which we have derivatives. Then we minimize the simpler problem using the idea of classical trust-region method. At each iteration, we solve the sub-problem to find

the possible candidate for the next iteration. Similarly to derivative based trust region method, if the model is qualified to valid, by computing the ratio between the actual improvement and the predicted improvement we will decide to increase, decrease or keep the current trust-region radius and choose the solution of sub-problem as the new center of trust-region or not. We keep running the algorithm till the stopping criteria are met, typically when the trust-region and the norm of gradient of the model are small enough.

The main differences between derivative free and derivative based trust region methods are listed below:

- the quadratic models are based on a given interpolation set (available simulations),
- the Taylor expansion error bound is replaced by fully linear or fully quadratic model properties, or model qualification condition.

In the next section, we will focus on the model construction and the necessary properties of the geometry of the interpolation set to control the error bounds of the models.

4.2.1 Geometry of the interpolation set

Consider a set of sample points given by

$$X = \{x^0, x^1, \dots, x^p\}, \;\; x^i \in \mathbb{R}^m. \tag{4.3}$$

where $p < (m+1)(m+2)/2$, we denote $p_1 = p + 1$ the number of points of the sample set.

We would like to build a quadratic model $m(x)$ which interpolates points in X. If $p_1 = (m+1)(m+2)/2$, the problem (4.1) is a quadratic interpolation problem; if $p_1 > (m+1)(m+2)/2$, we have over-determined problems, otherwise it is under-determined. In general, the problem is under-determined since the simulations are computationally expensive.

Let $m(x)$ denoted the polynomial of degree d (e.g., $d = 1, 1 < p_1 = m+1$ *or* $d = 2, m+1 < p_1 \leq (m+1)(m+2)/2$) that interpolates $f(x)$ at the points

$$m(x^i) = f(x^i), \;\; i = 0, \dots, p. \tag{4.4}$$

Let us consider $\mathcal{P}_m^d$ the space of polynomials of degree $\leq d$ in $\mathbb{R}^m$ and a basis $\phi = \{\phi_0, \phi_1, \phi_2, \dots, \phi_p\}$. We can express $m(x)$ as

$$m(x) = \sum_{i=0}^{p} \alpha_i \phi_i(x), \tag{4.5}$$

where $\alpha_i, i = 0, \dots, p$, are coefficients. It is clear that $m(x)$ is determined if the values of $\alpha_0, \dots, \alpha_p$ are determined. From (4.4) and (4.5), the coefficients α are found by solving the following linear system

$$\begin{pmatrix} \phi_0(x^0) & \phi_1(x^0) & \dots \phi_p(x^0) \\ \phi_0(x^1) & \phi_1(x^1) & \dots \phi_p(x^1) \\ \vdots & \vdots & \vdots \\ \phi_0(x^p) & \phi_1(x^p) & \dots \phi_p(x^p) \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_p \end{pmatrix} = \begin{pmatrix} f(x^0) \\ f(x^1) \\ \vdots \\ f(x^p) \end{pmatrix}, \tag{4.6}$$

or equivalently in matrix form:

$$M(\phi, X)\alpha = f(X).$$

If the matrix of the system $M(\phi, X)$ is non-singular (i.e., has full rank) then the system has unique solution. In this case, the interpolation set X is said to be poised. Thus, if X is a poised set, then the interpolating polynomial $m(x)$ exists and is unique. In [15] the authors prove that if the interpolation set is poised, then the condition number of $M(\phi, X)$ does not depend on the choice of the basis. The condition number of $M(\phi, \hat{X})$ can be considered as a measure of the poisedness of X, where $\hat{X}$ is a scaled interpolation set (detail in [15])

$$\hat{X} = \frac{1}{\Delta}[x^1 - x^0, \ldots, x^p - x^0],$$

with $\Delta = \Delta(X) = \max_{1 \le i \le p} \|x^i - x^0\|$.
In practice, we choose the natural basis

$$\{1, x_1, x_2, \ldots, x_m, \frac{1}{2}x_1^2, x_1x_2, \ldots, \frac{1}{(m-1)!}x_{m-1}^{d-1}x_m, \frac{1}{m!}x_m^d\}, \tag{4.7}$$

Definition 4.1. *(**Lagrange Polynomials**) Given a set of interpolation points $X = \{x^0, x^1, \ldots, x^p\}$, a basis of $p_1 = p + 1$ polynomials $l_j(x), j = 0, \ldots, p$, in $\mathcal{P}_m^d$ is called a basis of Lagrange polynomials if*

$$l_j(x^i) = \sigma_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$$

If X is poised, then the basis of Lagrange polynomials exists and is unique.

Definition 4.2. *($\Lambda-$ **poisedness**) Let $\Lambda > 0$ and a set $B \in \mathbb{R}^m$ be given. Let $\{\phi_i(x)\}_{i=0}^p$ be a basis of $\mathcal{P}_m^d$. A poised set $X = \{x^0, x^1, \ldots, x^p\}$ is said to be $\Lambda-$poised in B if and only if*

1. *for the basis of Lagrange polynomials associated with x*

$$\Lambda \ge \max_{0 \le i \le p} \max_{x \in B} |l_i(x)|,$$

or, equivalently,

2. *for any $x \in B$ there exists $\lambda(x) \in \mathbb{R}^{p_1}$ such that*

$$\sum_{i=0}^{p} \lambda_i(x)\phi(x^i) = \phi(x), \quad \text{with } \|\lambda(x)\|_\infty \le \Lambda,$$

or, equivalently,

3. *replacing any points in X by any $x \in B$ can increase the volume of the set $\{\phi(x^i), i = 0, \ldots, p\}$ at most by a factor Λ, where the volume is defined as*

$$vol(\phi(X)) = \frac{|det(M(\phi, X))|}{p_1!}.$$

One note that if a set is Λ_1- poisedness then it is also Λ_2-poisedness with $\Lambda_1 < \Lambda_2$, but the reverse does not hold.

4.2.2 Fully linear and fully quadratic models

As mentioned before, in trust region methods, the quality of the model approximation should be controlled just as for Taylor expansion models. To formalize this idea, we introduced the class of fully linear and fully quadratic model which is detailed in [7, 15].
We suppose that x_0 is given as the initial iterate. We define the level set

$$L(x_0) = \{x \in \mathbb{R}^m : f(x) \leq f(x_0)\}.$$

We do not only consider the objective function f within the level set, but also extend to the enlarge region

$$L_{enl}(x_0) = L(x_0) \cup \cup_{x \in L(x_0)} B(x, \Delta_{max}) = \cup_{x \in L(x_0)} B(x, \Delta_{max}).$$

Assumption 4.3. *Suppose x_0, Δ_{max} are given. Assume that f is continuously differentiable in an open domain containing the set $L_{enl}(x_0)$ and that ∇f is Lipschitz continuous on $L_{enl}(x_0)$*

Definition 4.4. ***(Class of fully linear models)*** *Let a function $f : \mathbb{R}^m \to \mathbb{R}$, that satisfies Assumption (4.3) , be given. A set of model functions $\mathcal{M} = \{m(x) : \mathbb{R}^m \to \mathbb{R}, m(x) \in \mathscr{C}^1\}$ is called a fully linear class of models if the following hold:*

1. *There exist positive constants κ_{ef}, κ_{eg} and ν_1^m such that for any $x \in L(x_0)$ and $\Delta \in (0, \Delta_{max}]$ there exists a model function $m(x+s) \in \mathcal{M}$, with Lipschitz continuous gradient and corresponding Lipschitz constant bounded by ν_1^m, and such that*

   - *The error between the gradient of the model and the gradient of the function satisfies*

   $$\|\nabla f(x+s) - \nabla m(x+s)\| \leq \kappa_{eg}\Delta, \forall s \in B(0, \Delta), \tag{4.8}$$

   - *the error between the model and the function satisfies*

   $$|f(x+s) - m(x+s)| \leq \kappa_{ef}\Delta^2, \forall s \in B(0, \Delta). \tag{4.9}$$

   Such a model m is called fully linear on $B(0, \Delta)$.

2. *For this class $\mathcal{M}$ there exists an algorithm which we will call a "model-improvement" algorithm, that in a finite, uniformly bounded (with respect to x and Δ) number of steps can*

   - *either establish that a given model $m \in \mathcal{M}$ is fully linear on $B(x; \Delta)$*
   - *or find a model $\tilde{m} \in \mathcal{M}$ that is fully linear on $B(x; \Delta)$.*

One example to illustrate the notion of fully linear model is the first-order Taylor approximation of a continuously differentiable locally Lipchitz function, i.e., $f \in \mathcal{C}^{1+}$.

Assumption 4.5. *Suppose x_0 and Δ_{max} are given. Assume that f is twice continuously differentiable in an open domain containing the set $L_{enl}(x_0)$ and that $\nabla^2 f$ is Lipchitz continuous on $L_{enl}(x_0)$.*

Definition 4.6. ***(Class of fully quadratic models)*** *Let a function $f : \mathbb{R}^m \to \mathbb{R}$, that satisfies Assumption (4.5) , be given. A set of model functions $\mathcal{M} = \{m(x) : \mathbb{R}^m \to \mathbb{R}, m(x) \in \mathscr{C}^2\}$ is called a fully quadratic class of models if the following hold:*

1. *There exist positive constants $\kappa_{ef}, \kappa_{eg}, \kappa_{eh}$ and ν_2^m such that for any $x \in L(x_0)$ and $\Delta \in (0, \Delta_{max}]$ there exists a model function $m(x+s) \in \mathcal{M}$, with Lipschitz continuous gradient and corresponding Lipschitz constant bounded by ν_2^m, and such that*

   - *The error between the Hessian of the model and the Hessian of the function satisfies*

   $$\|\nabla^2 f(x+s) - \nabla^2 m(x+s)\| \leq \kappa_{eh}\Delta, \forall s \in B(0,\Delta), \tag{4.10}$$

   - *The error between the gradient of the model and the gradient of the function satisfies*

   $$\|\nabla f(x+s) - \nabla m(x+s)\| \leq \kappa_{eg}\Delta^2, \forall s \in B(0,\Delta), \tag{4.11}$$

   - *the error between the model and the function satisfies*

   $$|f(x+s) - m(x+s)| \leq \kappa_{ef}\Delta^3, \forall s \in B(0,\Delta). \tag{4.12}$$

   Such a model m is called fully quadratic on $B(x;\Delta)$.

2. *For this class $\mathcal{M}$ there exists an algorithm which we will call a "model-improvement" algorithm, that in a finite uniformly bounded (with respect to x and Δ) number of steps can*

   - *either establishes that a given model $m \in \mathcal{M}$ is fully linear on $B(x;\Delta)$*
   - *or finds a model $\tilde{m} \in \mathcal{M}$ that is fully linear on $B(x;\Delta)$.*

It is clear that the class of fully quadratic models is better than the class of fully linear models in terms of Taylor-like bounded error, but we will see after that we need more simulations to build a fully quadratic model.

4.2.3 Building the trust region model

Next, we will indicate how to construct a fully linear (quadratic) model in the particular context of polynomial interpolation and regression. Note that we only consider DFO problems whose initial sample set has at least $m+1$ points.

In case the number of interpolation points is exactly equal to $m+1$.

We build a linear interpolation function $L_X(x) := \alpha_0 + \alpha^T x$ where (α_0, α_1) is the solution of

$$[1 \;\; X]\begin{pmatrix}\alpha_0\\ \alpha_1\end{pmatrix} = f(X). \tag{4.13}$$

If the interpolation set X is poised then there exists only one linear interpolation model whose coefficients satisfy (4.13).

When the number of interpolation points is larger than $m+1$ points and smaller than $\frac{(m_1)(x+2)}{2}$.

To build the model in this case, we can use a least square regression function $L_X(x)$ with the same form as in the case of linear interpolation, but (α_0, α_1) is now the solution of

$$\min_{\alpha_0,\alpha_1}\left[\sum_{i=0}^{m}(\alpha_0 + \alpha_1^T x^i - f(x^i))^2\right]. \tag{4.14}$$

We can also use the minimum Frobenius norm interpolation model $M_X = \alpha_0 + \alpha^T x + \frac{1}{2}x^T Hx$ where (α_0, α, H) is the solution of the following quadratic optimization problem

$$\begin{array}{l} \min\limits_{(\alpha_0,\alpha,H)} \dfrac{1}{2}\displaystyle\sum_{i=1}^{m}\sum_{j=1}^{m} h_{i,j}^2 \\ \alpha_0 + \alpha^T x^i + \frac{1}{2}(x^i)^T H x^i = f(x^i), \;\; i = 0, \ldots, p, \\ H = H^T. \end{array} . \tag{4.15}$$

When the number of interpolation points is exactly $\frac{(m+1)(m+2)}{2}$.
We build the quadratic interpolation function $M_X = \alpha_0 + \alpha^T x + \frac{1}{2}x^T Hx$, where (α_0, α^T, H) is the unique solution of

$$\alpha_0 + \alpha^T x^i + \frac{1}{2}(x^i)^T H x^i = f(x^i), \;\; i = 0, \ldots, p. \tag{4.16}$$

One note that the quadratic interpolation is fully quadratic if X is poised. Comparing with the linear interpolation, the error bounds of quadratic interpolation are tighter, and the function M_X provides both approximations of gradient and Hessian. Therefore, in term of iterations, quadratic interpolation can converge much faster, but it requires over-determined problem.
When the number of interpolation points is larger than $\frac{(m+1)(m+2)}{2}$.
In this case, we use regression to build the model as in the case where we have more than $m+1$ points for linear models.
To guarantee the fully linear (or fully quadratic) property of a model, we first have to check that the interpolation set is poised. If the sample set is not poised we need to improve its poisedness, as explained in the next section.

4.2.4 Model improvement

It is essential to check and improve the model during the optimization iterations. Firstly, we need to "improve " the sample set to ensure its poisedness. In [15], the authors give two main ways to improve the sample set: one based on the Lagrange polynomials and one based on LU factorization. In our study, we focus on LU factorization to improve the model. Model improvement algorithm is applied before each model update.

Algorithm 2: Improving poisedness of X via LU factorization, see [15]

0. Initialization

Initialize the pivot polynomial basis with some basis, e.g., the monomial basis (4.7), $u_i(x) = \phi_i(x), i = 0, \dots, p$. Select pivot threshold $\xi > 0$.

For $i = 0, \dots, p$

1. **Point selection:** If found j_i, i.e., $|u_i(x^{j_i})| \geq \xi$, swap x^{j_i} and x^i, otherwise, recompute x^i as
$$x^i \in \operatorname*{argmax}_{x \in B} |u_i(x)|,$$
stop if $|u_i(x^i)| < \xi$.

2. **Gaussian elimination:** For $j = i + 1, \dots, p$
$$u_j(x) \leftarrow u_j(x) - \frac{u_j(x^i)}{u_i(x^i)} u_i(x).$$

To illustrate the algorithm, we consider an example with a given initial set X_0 and three iterations of the improvement algorithm X_1, X_2 and X_3: (see Figure 4.1)

$$X_0 = \begin{pmatrix} 0 & 0 \\ 0.1 & 0 \\ 0.5 & 0 \\ 0.6 & 0.1 \\ 0 & 0.3 \\ 0 & 0.7 \end{pmatrix}, X_1 = \begin{pmatrix} 0 & 0 \\ 0.6 & 0.1 \\ 0 & 0.7 \\ 0.5 & 0 \\ 0.1 & 0 \\ -0.7 & -0.7 \end{pmatrix},$$

$$X_2 = \begin{pmatrix} 0 & 0 \\ -0.7 & -0.7 \\ 0 & 0.7 \\ 0.6 & 0.1 \\ 0.1 & 0 \\ 0.9899 & -0.9899 \end{pmatrix}, X_3 = \begin{pmatrix} 07 & 0 \\ 0.9899 & -0.9899 \\ -0.7 & -0.7 \\ 0 & 0.7 \\ 0.6 & 0.1 \\ -1.4 & 1.4 \end{pmatrix}.$$

Another way to improve the model is to use Lagrange polynomials. In [41] they use the definition of $\Lambda = \max_{0 \leq i \leq p} \max_{x \in B} |l_i(x)|$ and maintain the sample set X by choosing a point entering or leaving the set X so that the value of Λ is reduced. To do that the farthest point from the center of trust region is removed, i.e., the point of X associated with the largest value of Lagrange polynomials in absolute value.

Convergence results:

The derivative free trust region method is a local method. The local convergence for continuous problems is proven in [15]. The validity of the convergence of our model is based on the fully linear or fully quadratic property in definition (4.4), which is guaranteed by the poisedness of the interpolation set. Besides, the model must satisfy inequality (10.10) in [15]

$$\forall k,\ m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa}{2} \|\nabla m_k(x_k)\| \min\left(\frac{\|\nabla m_k(x_k)\|}{\|H_k\|}, \Delta_k\right),$$

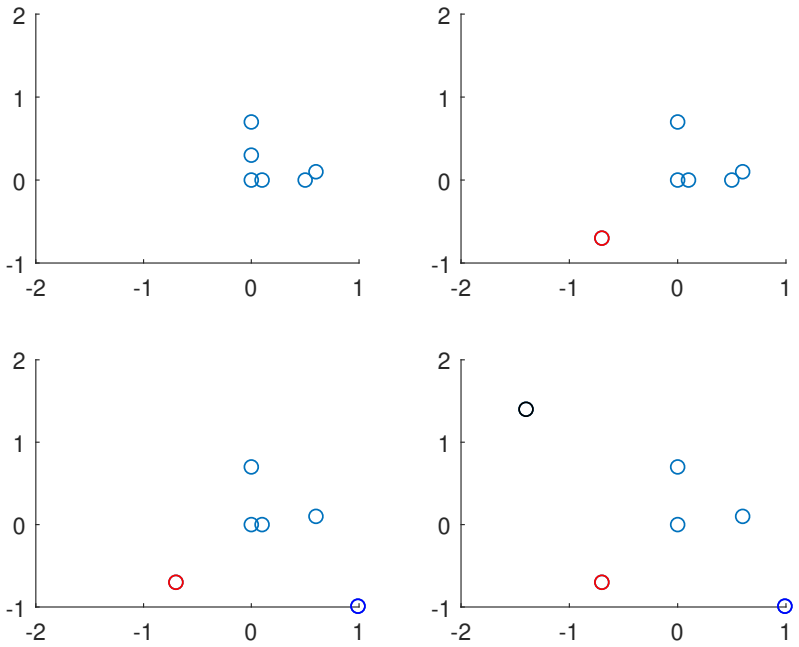

Figure 4.1: Improvement of the interpolation set via LU factorization

for some constant $\kappa \in (0, 1)$. The first order critical point convergence is shown in lemma (10.8) in [15]

$$\lim_{k \to +\infty} \|\nabla f(x_k)\| = 0.$$

4.3 Derivative free trust region method for mixed binary variables

In this section, we focus on extending derivative free methods for continuous optimization to mixed binary variables. We introduce binary variables $y \in \{0, 1\}^n$. The variables are extended from x to $z = (x, y), x \in \mathbb{R}^m, y \in \{0, 1\}^n$. By adding binary variables, the notation of the interpolation set becomes

$$Z = \{z^0, z^1, \ldots, z^p\}, z_i \in \mathbb{R}^m \times \{0, 1\}^n.$$

The classification of problems based on the value of p is kept, but with dimension $m + n$ (for mixed continuous and binary) instead of m (for continuous). We keep all the notations and definitions from before, but with mixed binary $z = (x, y)$ variables. Let us remark that when we deal with binary variables, in the natural basis $\phi_1(z), \ldots, \phi_p(z)$ we omit the quadratic binary terms due to the fact that $y_i^2 = y_i$:

$$\phi(z) = \{1, x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n, \frac{1}{2}x_1^2, \ldots, \frac{1}{2}x_m^2, \ldots, x_i x_j, \ldots, x_i y_j, \ldots, x_m y_n\}. \quad (4.17)$$

The proof of maintaining $\Lambda-$poised sample sets via a finite algorithm is mentioned in an extension of trust region method to binary variables of A.Conn, [14]. We recall the proof in the following lemma.

Lemma 4.7. *Let $v^T\phi(z)$ be a quadratic polynomial of degree at most d, where $\phi(z)$ is defined above and $\|v\|_\infty = 1$. Then, there exists a constant $\sigma_\infty > 0$ independent of v such that*

$$\max_{x\in B(0,1), y\in\mathbb{B}} |v^T\phi(z)| \geq \sigma_\infty. \tag{4.18}$$

For $d = 1, \sigma_\infty \geq 1$ and for $d = 2, \sigma_\infty \geq \frac{1}{4}$, where $B(x_k, \Delta_k)$ is the ball centered at x_k with radius Δ_k and $\mathbb{B}$ indicates binary.

Proof. We first show that such constant σ_∞ exists. Let us consider

$$\psi(v) = \max_{x\in B(0,1), y\in\mathbb{B}} |v^T\phi(z)|. \tag{4.19}$$

We see that $\psi(v)$ is a norm in the space of vector v. By using the norm relation in equation (2), we obtain the demonstration with the choice of σ_∞ as follows

$$\sigma_\infty = \min_{\|v\|_\infty=1} \psi(v). \tag{4.20}$$

Thus, if v has $l_\infty-$norm one then $\psi(v) \geq \sigma_\infty$ and there exits a continuous vector $x \in B(0,1)$ nad binary $y \in \mathbb{B}$ such that $|v^T\phi(z)| \geq \sigma_\infty$.

We demonstrate for two cases: for $d = 1, \sigma_\infty \geq 1$ and $d = 2, \sigma_\infty \geq \frac{1}{4}$.

For $d = 1$, we have the natural polynomial basis of degree one is

$$\phi(z) = [1, x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n]^T.$$

Let $u = [v_1, v_2, \ldots, v_{m+1}]^T$ and $w = [v_{m+2}, v_{m+3}, \ldots, v_{m+n+1}]^T$. Therefore, $\psi(v)$ becomes $\max_{x\in B(0,1), y\in\mathbb{B}} |v_1 + u^Tx + w^Ty|$ which reach the optimal value at $x = \frac{u}{\|u\|}$ and y are chosen as

$$y_i = 1, \text{if } v_{i+m+1} > 0,$$
$$y_i = 0, \text{otherwise},$$

or $x = -\frac{u}{\|u\|}$ and

$$y_i = 1, \text{if } v_{i+m+1} < 0,$$
$$y_i = 0, \text{otherwise}.$$

Thus, the optimal value of $\psi(v)$ is $|v_1| + \|u\| + \sum_{y_i\neq 0} |v_{i+m+1}| \geq 1$. We finish the proof of the first case.

For $d = 2$: the natural polynomial basis of degree 2 is given as

$$\phi(z) = \{1, x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n, \frac{1}{2}x_1^2, \ldots, \frac{1}{2}x_m^2, \ldots, x_ix_j, \ldots, x_iy_j, \ldots, x_my_n\}.$$

Since $\|v\|_\infty = 1$, by the definition of $l_{\text{inf}}-$norm, there exits $i : |v_i| = 1$. Thus, one of the coefficients of the polynomial $q(z) = v^T\phi(z)$ is equal $1, -1$ (corresponding to linear part in the basis $1, x_1, \ldots, x_m, y_1, \ldots, y_n$) or $\frac{1}{2}, -\frac{1}{2}$ (corresponding to quadratic part in the basis $\frac{1}{2}x_1^2, \ldots, \frac{1}{2}x_m^2, \ldots, x_ix_j, \ldots, x_iy_j, \ldots, x_my_n$).

Let us consider only the cases where one of the coefficients of $q(z)$ is 1 or $\frac{1}{2}$ (the case -1 or $-\frac{1}{2}$ would be analyzed similarly).

The largest coefficient in absolute value in v corresponds to a term which either a constant term, a linear term x_i or y_i, a quadratic term $\frac{1}{2}x_i^2$ or x_ix_j or x_iy_j. Let us restrict all variables that do not appear in this term to zero. We will show that the maximum absolute value of $q(z)$ is at least $\frac{1}{4}$ by considering 6 cases of different terms that correspond to the largest coefficient. In each case we will evaluate $q(z)$ at several points in the unit ball and show that at least at one of these points $|q(z)| \geq \frac{1}{4}$:

- $q(z) = 1$: it implies directly $|q(z)| \geq \frac{1}{4}$

- $q(z) = x_i + \frac{1}{2}\alpha x_i^2 + \delta$: in this case we have

$$q(1) = 1 + \frac{\alpha}{2} + \delta,$$
$$q(-1) = -1 + \frac{\alpha}{2} + \delta,$$

  If $\frac{\alpha}{2} + \delta < 0 \longrightarrow |q(-1)| > 1$, otherwise, if $\frac{\alpha}{2} + \delta \geq 0 \longrightarrow |q(1)| \geq 1$.

- $q(z) = y_i + \delta$: if $|\delta| \geq \frac{1}{4}$ then $|q(x, 0)| = |\delta| \geq \frac{1}{4}$. Otherwise $\frac{-1}{4} < \delta < \frac{1}{4}$ then $|q(x, 1)| = |1 + \delta| > \frac{3}{4}$.

- $q(z) = \frac{1}{2}x_i^2 + \alpha x_i + \delta$: in this case we have:

$$q(1) = \frac{1}{2} + \alpha + \delta,$$
$$q(-1) = \frac{1}{2} - \alpha + \delta,$$

  It is satisfied directly if one of the inequalities $|q(1)| \geq \frac{1}{4}$ or $|q(-1)| \geq \frac{1}{4}$ hold . In both cases $|q(1)| < \frac{1}{4}$ and $|q(-1)| < \frac{1}{4}$ hold, then, by adding these inequalities, we have $\frac{-1}{2} < 1 + 2\beta < \frac{1}{2}$ implies that $\beta < \frac{-1}{4}$, and also we have $q(0) = \beta < \frac{-1}{4}$, thus, we have $|q(0)| > \frac{1}{4}$.

- $q(z) = x_ix_j + \frac{1}{2}\alpha x_i^2 + \frac{1}{2}\beta x_j^2 + \gamma x_i + + \delta x_j + \epsilon$: in this case, we consider $q(z)$ at four points

$p_1 = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}), p_2 = (\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}), p_3 = (\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}), p_4 = (\frac{-1}{\sqrt{2}}, \frac{-1}{\sqrt{2}})$, we have

$$\begin{aligned} q(p_1) &= \frac{\alpha+\beta}{4} + \frac{1}{2} + \frac{\gamma+\delta}{\sqrt{2}} + \epsilon, \\ q(p_2) &= \frac{\alpha+\beta}{4} - \frac{1}{2} + \frac{\gamma-\delta}{\sqrt{2}} + \epsilon, \\ q(p_3) &= \frac{\alpha+\beta}{4} - \frac{1}{2} - \frac{\gamma-\delta}{\sqrt{2}} + \epsilon, \\ q(p_4) &= \frac{\alpha+\beta}{4} + \frac{1}{2} - \frac{\gamma+\delta}{\sqrt{2}} + \epsilon, \end{aligned}$$

As a result, we obtain $q(p_1) - q(p_2) = 1 + 2\delta$ and $q(p_3) - q(p_4) = -1 + 2\delta$. If $\delta \leq 0$, we have $q(p_1) - q(p_2) \geq 1$. We divide in two cases: if $|q(p_1)| < \frac{1}{2}$ then $q(p_2) \leq \frac{-1}{2}$, it implies that $|q(p_2)| \geq \frac{1}{2}$. Otherwise, $\delta > 0$, $q(p_3) - q(p_4) \leq -1$. Thus if $|q(p_3)| < \frac{1}{2}$, then $q(p_4) \leq \frac{-1}{2}$.

- $q(z) = x_i y_j + \frac{1}{2}\alpha x_i^2 + \beta x_i + \gamma y_j + \delta$: in this case, we evaluate $q(z)$ at six points with the evaluation as follows

$$\begin{aligned} q(0,0) &= \delta, \\ q(0,1) &= \gamma + \delta, \\ q(1,0) &= \frac{\alpha}{2} + \beta + q(0,0), \\ q(-1,0) &= \frac{\alpha}{2} - \beta + q(0,0), \\ q(1,1) &= \frac{\alpha}{2} + (1+\beta) + q(0,1), \\ q(-1,1) &= \frac{\alpha}{2} - (1+\beta) + q(0,1). \end{aligned}$$

From last two equation, we have

$$q(1,1) - q(-1,1) = 2 + 2\beta.$$

We will prove that with all the possible value of β, we can find at least one point which satisfies the lemma. If $\beta \geq 0$ we have

$$q(1,1) - q(-1,1) = 2 + 2\beta \geq 2.$$

If $|q(-1,1)| < 1$ then $q(1,1) > 1$, otherwise, if $|q(-1,1)| > 1$ it is trivial.
If $\frac{-1}{2} < \beta < 0$, we have

$$q(1,1) - q(-1,1) > 1,$$

then if $|q(1,1)| < \frac{1}{2}$ implies that $-q(1,1) > 1 - q(1,1) > \frac{1}{2}$, so we obtain $|q(-1,1)| > \frac{1}{2}$. Otherwise, if $|q(1,1)| > \frac{1}{2}$ it is trivial.
If $\beta \leq \frac{-1}{2}$, from the third and the fourth equations, we have

$$q(1,0) - q(-1,0) = 2\beta \leq -1,$$

in this case, if $|q(-1,0)| < \frac{1}{2}$ then $q(1,0) < \frac{-1}{2}$, it implies that $|q(1,0)| > \frac{1}{2}$. Otherwise, if $|q(-1,0)| \geq \frac{1}{2}$ then it is trivial.

□

The basic Derivative free Optimization with mixed Binary variables (DFOb) trust region method approach for mixed binary problems in iteration k is described mainly as follows:

- Step 0: build quadratic subproblem model $m_k(x, y)$ for both continuous and binary variables.
- Step 1: solve a continuous subproblem (QP) by temporary fixing the discrete variables y. In this step, we aim to find an accepted continuous candidate ($\rho_k \geq \eta_0$, see Algorithm (1)). If the solution of step 1 is not accepted, an improvement step is activated to improve the poisedness of the current interpolation set.
- Step 1.5a: if we succeed to find the continuous candidate in step 1, we attempt to improve x and y by solving a MBQP subproblem.
- Exploration phase: before termination, we use an exploration phase to force the algorithm to explore a different region of the binary space.
  The details of these steps will be indicate in the next part.

4.3.1 The trust region subproblems

When solving mixed binary optimization by DFOb trust region methods, we have to solve three different types of subproblems:

1. Model improvement subproblem:
$$\max_{z \in B} |u_i(z)|, \tag{4.21}$$
where $u_i(z)$ is natural basis in (4.7) and Algorithm (2). In practice, we use MIQP solvers such as CPLEX, BONMIN or NOMAD to solve this subproblem.

2. Model minimization subproblems
   - QP (Step 1)
$$\begin{cases} \min\limits_{x} m_k(x, y_k) \\ s.t. \quad \|x - x_k\|_\infty \leq \Delta_{x,k}, \end{cases} \tag{4.22}$$
   where $\Delta_{x,k}$ is the trust region radius with respect to continuous variables x at iteration k and (x_k, y_k) is the current center (current best point) in the feasible region. Notice that we use the infinite norm l_∞ to define the trust region with respect to continuous variables. We only consider the problem within the trust region in Step 1.
   - MBQP (Step 1.5a)
$$\begin{cases} \min\limits_{x,y} m_k(x, y) \\ s.t. \quad \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ \qquad \|y - y_k\|_H \leq \Delta_{y,k}, \end{cases} \tag{4.23}$$

with (x_k, y_k) the current center of the trust region. Here H denotes the classical Hamming distance for binary variables. The Hamming distance is the number of positions at which the corresponding symbol of two vectors of the same length are different. Thus, we define our trust-region for the binary variables by $\Delta_{y,k}$ which defines the right-hand side of the constraint on the Hamming distance between binary vector y and y_k, the centre of the binary trust region

$$\sum_{j:y_{k,j}=0} y_j + \sum_{j:y_{k,j}=1} (1 - y_j) \leq \Delta_{y,k}. \tag{4.24}$$

If we can not improve the current point (local solution), we want to force the algorithm to explore a different region of binary space. To achieve this, we relax the local branching constraint (4.24) and add a "no-good-cut" constraint to obtain a new center

$$\sum_{j:y_j^*=0} y_j + \sum_{j:y*_j=1} (1 - y_j) \geq k^*, \tag{4.25}$$

where (y^*, k^*) are respectively the center and the radius of the explored domain, $k^* \geq 1, k^* \in \mathbb{N}$. Note that we will have a bunch of constraints (4.25) for each explored region.

It is possible that the current trust region does not contain any feasible points due to a large number of no-good-cut constraints, or no better point could be found. In those cases, new simulations are added to try to improve the model accuracy.

4.3.2 Stopping criteria and convergence results

The algorithm stops either because the maximum budget of simulation or the maximal number of nogoodcuts are reached. The maximal number of nogoodcuts is theoretically equal to $2^n - 1$ (for n binary variables). We will see that in the case of cyclic symmetric problems, it is approximately $\frac{2^n}{n}$.

The convergence results proved in [15] can be extended to the algorithm for mixed binary problems (on the work from personal communication of A.Conn [14]).

A pseudo-code for the Derivative-free Optimization trust-region method (DFOb) for mixed binary problems is given in the following:

Algorithm 3: Derivative-Free trust-region method for mixed binary variables

Input: $\epsilon_{good} > \epsilon_{ok} > 0, \Delta_{x,\max} > \Delta_{0,x} > \Delta_{x,\min} > 0, \Delta_{y,\max} > \Delta_{0,y} > \Delta_{y,\min} > 0$

0. Initialization

Initial interpolation set of p points $\{(x_1, y_1, f_1), (x_2, y_2, f_2), \ldots, (x_p, y_p, f_p))\}$ with $f_i = f(x_i, y_i)$

Define trust-region center (x_0, y_0) associated with $f_0 = \min(f_1, \ldots, f_p)$

1. Main iteration loop

- **STEP 0:** Build quadratic sub-problem model $m_k(x, y)$
- **STEP 1a:** Solve the TR quadratic sub-problem for fixed y_k and model improvement step (MI)
$$x^* = \underset{x}{\operatorname{argmin}}\, m_k(x, y_k), s.t. \|x - x_k\|_\infty \leq \Delta_{x,k}, (QP)$$
- **STEP 1b:** Compute $\rho = (f_k(x_k, y_k) - f_k(x^*, y_k))/(m_k(x_k, y_k) - m_k(x^*, y_k))$,
- **STEP 1c:** $x_{k+1} \leftarrow x^*$ if $\epsilon_{tol} \leq \rho < \epsilon_{ok}$ or $\rho \geq \epsilon_{ok}$ (success step: $f(x^*, y)$ is smaller) Update quadratic model with new values of f (from Step 1.a: QP and MI)
- **STEP 1.5a :** If $x_{k+1} \neq x_k$ (successful STEP 1), attempt to improve y Solve the TR MIQP sub-problem for both (x, y)
$$(x^*, y^*) = \underset{x,y}{\operatorname{argmin}}\, m_k(x, y), s.t. \|x - x_k\|_\infty \leq \Delta_{x,k}, \|y - y_k\|_H \leq \Delta_{y,k},$$
- **STEP 1.5b:** Trust region management
  - If $y^* \neq y_k$ and $f(x^*, y^*)$ is smaller than $\min(f_i)_{i=0,\ldots,p}$: $k \leftarrow k + 1, (x_k, y_k) \leftarrow (x^*, y^*)$
  - If $y^* \neq y_k$ but no improvement: $k \leftarrow k + 1, \Delta_{y,k} \leftarrow \Delta_{y,k} - 1, \Delta_{x,k} \leftarrow 2\Delta_{x,k}$ if $\rho \geq \epsilon_{good}$ or $\Delta_{x,k} \leftarrow \Delta_{x,k}/2$ if $\rho < \epsilon_{ok}$
- **Check stopping criteria:** $\Delta_{x,k} < \Delta_{\min}$ or no change in (x, y) or very small improvement.
- **Before termination:** Add a nogoodcut constraint to force exploration in y
$$\|y - y_k\|_H \geq k^*, \; k^* > 0 \text{ for next step 1.5a .}$$

Chapter 5

An adapted distance for cyclic binary problems

We begin the chapter by reminding our target application: the optimal design of the blade shape of a turbine for aircraft's engine in order to minimize the vibrations. In practice, the aim is to optimize the blade arrangement on the disk of two different pre-defined shapes of blades: a reference shape called A and a mistuning shape B. A binary variable y_i is associated with each blade location, taking value 0 for shape A and value 1 for shape B. The optimization solution provides the distribution of the two shapes around the turbine disk.

In practice, the reduced order model (ROM) is used. However, it removes a large number of possibilities for optimization, in the sense that ROM considers two reduced sub-problems but the cyclic symmetry property remains and is not taken into account explicitly in the standard optimization workflow. For instance, in the case of 12 blades, ROM considers two sub-problems with 6 binary variables, which ends up with only 28 distinct arrangements (14 for each sub-problem), while the total number of distinct arrangements in the original problem is 352. But the removed configurations could be "good" candidates for optimization.

Therefore, we attempt to define a new distance which can model the cyclic symmetry and thus avoid the redundancy. This new distance is expected to have a simple form. As noted in the previous chapter, the standard Hamming distance does not appear appropriate for this target.

In this chapter, we focus on defining this new distance, which is inspired by the concept of "Necklace" in literature [22, 23]. In addition, the adaptation of the optimization method is discussed.

5.1 Necklace context

5.1.1 Concept of "Necklace" and associated distances

The idea of arranging two different types of blade shapes on the disk is similar to distribute two different colors of beads on a necklace. In the application field of this concept, we identify several promising distances.

Definition 5.1. *(**Necklace**) In combinatorics, a k-ary necklace of length n is an equivalent class of n-character strings over an alphabet $\sum^k = \{a_1, \ldots, a_k\}$ of size k, taking all rotations*

as equivalent. It represents a structure with n circularly connected beads which have k available colors.

Let $Neck(n)$ be the set containing all the necklaces of length n. For instance, considering the necklace with 2 colors and 4 beads ($n = 4$, $k = 2$, $\sum^k = \{0, 1\}$), we have the following list of representative necklaces (in red color) and all rotations (in black)

0000	0001	0011	0101	0111	11111
	0010	0110	1010	1110	
	0100	1100		1101	
	1000	1001		1011	

Our application can be seen as a 2-colors necklace optimization with a fixed number of beads. The number of distinct arrangements is approximately that given $\backsim \frac{2^n}{n}$ in the turbine application [37, 49]. By using the result from [22], we obtain the exact number of necklaces for a given number of beads n is equal to $\frac{1}{n}\sum_{d|n}\phi(d)2^{n/d}$, where ϕ is Euler's totient function, i.e. the function that counts the positive integers up to n that are relatively prime to n, and where the summation is taken over all divisors d of n.

There are a large number of necklace's applications, many of them based on necklace distances such as similarity of different types of music [44, 45, 47], the dissimilarity of DNA in biology [28], calculating the leap years and design calendars [19], painting car in manufacturing [21]... In the the next section, we review some usual "necklace" distances.

5.1.2 Survey of "necklace" distances

The study of distances in continuous space is well-known and widely extended in the literature. Nevertheless, in the discrete space, there is a limitation of defining a distance: the choice of a distance depends on the problem formulation. Taking into account cyclic symmetry property is one difficulty of our problem. It can save a large number of simulations in the optimization procedure. There exists a wide variety of distances for measuring the difference between two discrete strings. We list the main ones in the following.

Definition 5.2. *(**The Hamming distance**) Given two binary strings $y = (y_1, \ldots, y_n)$ and $y' = (y'_1, \ldots, y'_n)$, the Hamming distance between y and y' is given by*

$$d_H(y, y') = \sum_{i=1}^{n} |y_i - y'_i|. \tag{5.1}$$

The Hamming distance is easily computed in $\mathcal{O}(n)$ operations. The constraint $d_H(y, y_{ref}) \leq c$ for a given y_{ref} and constant c is linear.

Definition 5.3. *(**Varshamov distance**) Given two binary strings $y = (y_1, \ldots, y_n)$ and $y' = (y'_1, \ldots, y'_n)$, the Varshamov distance between y and y' is given by*

$$d_V(y, y') = \max(N_{01}(y, y'), N_{10}(y, y')), \tag{5.2}$$

where $N_{01}(y, y') = \#\{(y_i, y'_i) : y_i = 0, y'_i = 1\}$ and $N_{10}(Y, Y') = \#\{(y_i, y'_i) : y_i = 1, y'_i = 0\}$.

For more details, see [42]. Varshamov distance can detect cases where the number of 1 in string y is larger than the number of 1 in string y'.

Definition 5.4. *(**The swap distance**) We represent y, y' above as $U = (u_1, \ldots, u_k), V = (v_1, \ldots, v_k)$ where u_i, v_i are the indices such that $y_{u_i} = 1$ and $y'_{v_i} = 1$, the swap distance is given by*

$$d_{swap}(y, y') = d_{swap}(U, V) = \sum_{i=1}^{k} |u_i - v_i|. \tag{5.3}$$

In [20] the authors use the swap distance to measure the similarity of two rhythms (application in classification of type of music) or bioinformatics where the two strings to be compared are chain polymers. Computing U, V from y, y' costs $\mathcal{O}(n)$ operations and Toussaint [46] states that the swap distance can be computed in $\mathcal{O}(n^2)$. However, in [20], the authors introduce algorithms computing in $\mathcal{O}(k^2)$ and $\mathcal{O}(k^3)$ operations.

Definition 5.5. *(**The Hamming distance with shifts**) Given y, y' two binary sequences of the same length, we define three types of operations on y*

- *An insertion $ins(i)$ changes $y[i]$ from 0 to 1 with cost c_{ins};*
- *A deletion $del(i)$ changes $y[i]$ from 1 to 0 with cost c_{del};*
- *A shift $sh(i, j)$ changes $y[i]$ from 1 to 0 and $y[j]$ from 0 to 1 with cost $\|i - j\|c_{sh}$.*

The Hamming distance with shifts between y and y' is the minimum cost of a sequence of operations that transform y to y'.

The Hamming distance with shifts was mentioned in [29], it measures not only the number of mismatches but also how far apart the mismatches occur.
We note that with the values of $c_{ins} = c_{del} = 1, c_{sh} = \infty$ it becomes the Hamming distance and $c_{ins} = c_{del} = \infty, c_{sh} = 1$ it becomes the swap distance.
In [30] Jiang states that the Hamming distance with shifts can be computed in $\mathcal{O}(n)$ operations.

Definition 5.6. *(**The Euclidean interval vector distance**) Given y, y' two binary sequences of the same length, we represent them as their interval vectors $U' = (u'_1, \ldots, u'_k), V' = (v'_1, \ldots, v'_k)$ where u'_i, v'_i are the number of 0 elements between two elements of 1. For instance, $X = (1100100000)$ can be represented as $X = (0, 2, 5)$. Then the Euclidean interval vector distance for two binary strings with the same length and the same number of 1 is given by*

$$d_E(y, y') = \sqrt{\sum_{i=1}^{k} (u'_i - v'_i)^2}. \tag{5.4}$$

The interval vectors can be computed from the given binary sequences in $\mathcal{O}(n)$ operations, $d_E(y, y')$ can be computed in $\mathcal{O}(k)$ operations, where k is the number of 1 in the sequence. We note that this distance does not take into account the cyclic symmetry property.

Definition 5.7. *(**The interval-difference vector distance**) Given y, y' two binary sequences and $U' = (u'_1, \ldots, u'_k), V' = (v'_1, \ldots, v'_k)$ their interval vectors. For each sequence, we define new sequences, $Z = (z_1, \ldots, z_k)$, with $z_i = u'_{i+1}/u'_i, i = 1, \ldots, k-1, z_k = y_k/u'_1$ and $Z^1 =$*

$(z_1^1, \ldots, z_k^1)$, *with* $z_i^1 = v'_{i+1}/v'_i, i = 1, \ldots, k-1, z_k^1 = y'_k/v'_1$. *the interval difference vector distance is*

$$d_{ID}(y, y') = \Big(\sum_{i=1}^{k} \frac{\max(z_i, z_i^1)}{\min(z_i, z_i^1)}\Big) - k. \tag{5.5}$$

In [44] $d_{ID}(y, y')$ is computed in $\mathcal{O}(n)$ operations.

Definition 5.8. *(**Minimum distance of pairs assignment (MDPA)**) Given* y, y' *two integer sequences, MDPA is defined as*

$$d_{MDPA}(y, y') = \min_{Y,Y'}\Big(\sum_{i,j=1}^{n} |y_i - y'_j|\Big). \tag{5.6}$$

The definition can be applied to two binary strings.

There are other distances related to the necklace application field, such as the chronotonic distance or the geometry distance [44, 46]. We listed above the main ones to see how the formulations are and how the cyclic symmetry can be taken into account. In the next section, we discuss about a new necklace distance adapted to our application.

5.2 Necklace distance adapted to DFO

The cyclic symmetry property motivates the search for an appropriate distance to adapt to the algorithm. Inspire from the concept of necklace and its distances, we build a new distance and adapt it to DFO.

5.2.1 Distance formulation and its properties

In this section, we indicate the new distance formulation.

Formulation

Firstly, we recall Necklace Alignment Problem (NAP), from [8], that gives us ideas of building the new distance. The goal of NAP is to find the angle rotation offset $c \in [0, 1)$ and the perfect matching shift $s \in \{0, 1, \ldots, n\}$ between two sorted vectors of real numbers $y = (y_1, \ldots, y_n)$, $y' = (y'_1, \ldots, y'_n)$:

$$\min_{c,s} \sum_{i=1}^{n} (d^0((y_i + c) mod(1), y'_{(i+s) mod(n)}))^p, \text{ where } d^0(y_i, y'_i) = \min(|y_i - y'_i|, 1 - |y_i - y'_i|), \tag{5.7}$$

with $p = 1$ or 2. The l_1 necklace alignment problem, also called cyclic swap distance or necklace swap distance, is restricted to integer vectors.

Inspired by the main idea of NAP, we propose a new distance that is appropriate to our problem.

Definition 5.9. *(**The necklace distance**) Given two binary strings* y, y' *of length* n*, the necklace distance is defined as the minimal value of the Hamming distances between* y *and all the rotations of* y'*:*

$$d_{neck}(y, y') = \min(g_1(y, y'), \ldots, g_n(y, y')), \tag{5.8}$$

where $g_r(y, y') = d_H(y, Rot^r(y'))$, $r = 1, \ldots n$, $Rot^r(y')$ *is the rotation of* y' *by* r *positions, i.e.,* $Rot^r(y') = (y'_{r+1}, y'_{r+2} \ldots, y'_1, y'_2, \ldots, y'_r)$.

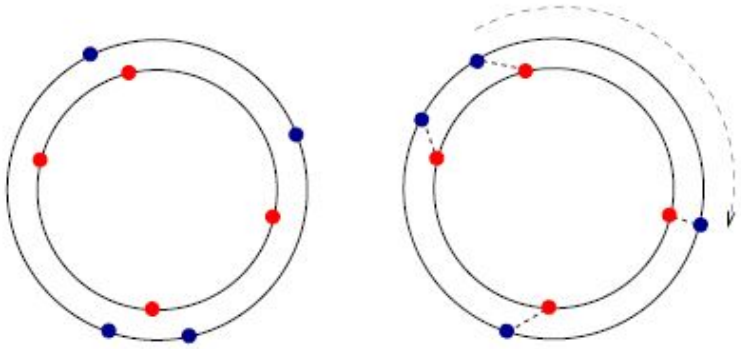

Figure 5.1: An example of necklace alignment, see [8]

We denote $Rot(y) := \{Rot^r(y), 0 \leq r < n\}$ and Π be the permutation matrix :

$$\Pi = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

By using the fact that $\Pi^r y = Rot^r(y)$, we obtain

$$Rot(y) = \{\Pi^r y : 0 \leq r < n\}.$$

Metric properties of the necklace distance d_{neck}
In the following, we show the metric properties of d_{neck}.

Proposition 5.10. *(**Non-negativity property**) d_{neck} has the non-negativity property*

$$d_{neck}(y, y') \geq 0, \forall y, y' \in \{0,1\}^n.$$

Proof. The statement is a corollary of the non-negativity property of the Hamming distance. □

Proposition 5.11. *(**Reflexivity property**) d_{neck} has the reflexivity property*

$$d_{neck}(y, y) = 0.$$

Proof. Since $d_H(y, y) = 0$ holds true, 0 is the minimum bound of $d_{neck}(y, y')$. Therefore $d_{neck}(y, y) = 0$. □

Proposition 5.12. *(**Commutativity property**) d_{neck} has the commutativity property*

$$d_{neck}(y, y') = d_{neck}(y', y).$$

Proof. Based on the fact that $d_H(y, Rot^r(y')) = d_H(y', Rot^{n-r}(y))$, thus $\min_r d_H(y, Rot^r(y')) = \min_r d_H(y', Rot^{n-r}(y)) = \min_r d_H(y', Rot^r(y))$. □

Proposition 5.13. *(**Triangle inequality property**) d_{neck} satisfies the triangle inequality property:*

$$d_{neck}(y, y') \leq d_{neck}(y, y") + d_{neck}(y", y').$$

Proof. Direct consequence of the triangle inequality property of Hamming distance. □

To state the next metric property, we need to introduce a definition:

Definition 5.14. *(**Cyclic symmetry property in distance**) We say a distance d has the cyclic symmetry property if*

$$d(y, y') = 0 \Longleftrightarrow y \in Rot(y'),$$

for any given two distinct vectors y, y'.

Proposition 5.15. *(**Cyclic symmetry property**) d_{neck} has cyclic symmetry property*

$$d_{neck}(y, y') = 0 \Longleftrightarrow y \in Rot(y'), \forall y, y' \in \{0, 1\}^n.$$

Proof. To prove this statement, we prove both implications:

1. Proof of the first implication " $\Longrightarrow$ ": "$d_{neck}(y, y') = 0 \Longrightarrow y \in Rot(y')$
   Assume $d_{neck}(y, y') = 0 \Longrightarrow \min_r(d_H(y, Rot^r(y'))) = 0 \Longrightarrow \exists i : d_H(y, Rot^i(x)) = 0$, since the metric property of Hamming distance $d_H(y, Rot^i(y')) = 0 \Longleftrightarrow y = Rot^i(y')$.

2. Proof of the second implication " $\Longleftarrow$ ": $y \in Rot(y') \Longrightarrow d_{neck}(y, y') = 0$
   If $y \in Rot(y')$ then $\exists i : y = Rot^i(y') \longrightarrow d_H(y, Rot^i(y')) = 0$, then $\min_r d_h(y, Rot^r(y')) = 0$, hence $d_{neck}(y, y') = 0$.

□

The new distance d_{neck} defined in (5.8) takes into account the cyclic symmetry property. However, the appearance of the "min" operator leads to difficulties when it is used in the optimization procedure. In the next part, we propose a reformulation of d_{neck} adapted to DFO.

5.2.2 Observations about necklace distance imply to simplified simulations

In this section, we aim to study the regularity of the objective function with respect to necklace distance, compare to the Hamming distance.

We display in Figure (5.2) the differences in objective functions of simplified simulation provided by Safran (see Section (6.5)) with respect to the necklace distance and the Hamming distance for all of arrangements (with fixed value of continuous variable).

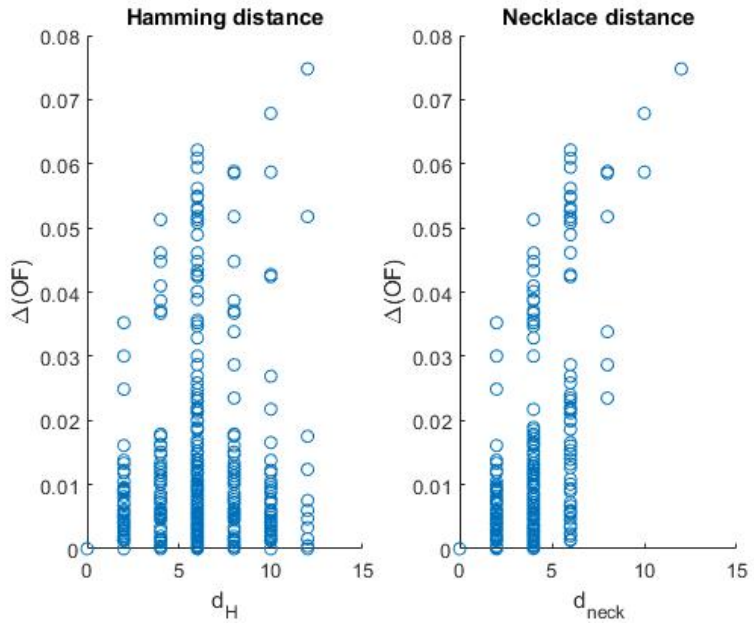

Figure 5.2: Regularity of the objective function with respect to the necklace distance and the Hamming distance.

This figure shows that necklace distance has a better regularity performance: if two points are closed, then the difference of associated objective functions is small.

5.2.3 Reformulation of QP problems with necklace distance

The derivative-free trust-region method uses quadratic models to approximate the objective function. Here, we would like to keep the QP structure of the subproblems even when using the necklace distance. Nevertheless, the "min" operator in the necklace distance prevents.

In this section, we propose a QP reformulation of the subproblems involved in the DFO method.

Reformulation of nogoodcuts constraints: For the MIQP subproblem in step 1.5a of Algorithm 3, to explore the binary space, we use the constraints

$$\sum_{j:y_j^*=0} y_j + \sum_{j:y_j^*=1} (1 - y_j) \geq k^* > 1,$$

associated with the Hamming distance d_H. To adapt the necklace distance to this step, we use the following remark

REMARK 5.1. Given $a_1, a_2, \ldots, a_n$ and k^* real constants, we have

$$\min_{i=1,\ldots,n}(a_1, a_2, \ldots, a_n) \geq k^* \iff a_i \geq k^*, \forall i = 1, \ldots, n, \tag{5.9}$$

Using remark (5.1) with $a_i = d_H(y, Rot^i(y_c)),\ \ i = 1, \ldots, n$, we obtain

$$\min(d_H(y, Rot(y_c)), \ldots, d_H(y, Rot^n(y_c))) \geq k^* \iff d_H(y, Rot^i(y_c)) \geq k^*, i = 1, \ldots, n. \tag{5.10}$$

Consequently, we can add n linear constraints based on the Hamming distance to reformulate nogoodcuts associated to the necklace distance.

Reformulation of the trust region problem (without nogoodcuts): We recall the problem to be reformulated

$$\begin{cases} \min\limits_{z} f(z) \\ s.t. \quad \min\limits_{i=1,\dots,n} \{g_i(z)\} \leq \Delta, \end{cases} \tag{P_1}$$

with $z = (x, y), g_i(z) = d_H(y, y^c), i = 1, \dots, n$.

Before starting the new proposition, let us define two optimization problems to be *equivalent* if an optimal solution of one problem straightforwardly gives an optimal solution of the other problem.

Proposition 5.16. *(Mini-min formulation) Let $M > 0$ be a sufficiently large positive real constant, let $\mu > 0$ be a given constant, n, m be positive integers, and let $g_i : \mathbb{R}^m \times \mathbb{Z}^n \to \mathbb{R}, i = 1, 2, \dots, n$, be real-valued functions satisfying $0 \leq g_i(z) \leq M, \forall z$. Then, the two following optimization problems (P_1) and (P_2) are equivalent*

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

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

Proof. **Problem (P_1) is clearly equivalent to**

$$\begin{cases} \min\limits_{z,t} f(z) + \mu t \\ s.t. \quad t = \min\limits_{i=1,\dots,n} \{g_i(z)\}. \end{cases} \tag{P_1'}$$

We prove the proposition in two steps: first, (P_2) is a relaxation of (P_1'), and, if there exists solutions of (P_2) then necessarily they are feasible points of (P_1').

Let us consider the first assertion: **(P_2) is a relaxation of (P_1')**. More precisely, if $(\bar{z}, \bar{t})$ is a feasible solution of (P_1'),then $(\bar{z}, \bar{y}, \bar{t})$ is a feasible solution of P_2, where

$$\bar{y}_i := \begin{cases} 0 \,\text{if } i \text{ is the smallest index such that } t = \min\limits_{i=1,\dots,n} \{g_i(\bar{z})\}, \\ 1, \text{otherwise.} \end{cases} \tag{5.11}$$

Let I be the index such that $\bar{y}_i = 0$. From the definition of $\bar{y}$, we note that $\bar{t} = g_I(\bar{z}), \bar{y}_I = 0 \,\text{and}\, \bar{y}_i = 1 \,\text{for all}\, i \neq I$.

Let us show that $(\bar{z}, \bar{y}, \bar{t})$ is indeed feasible for problem (P_2). For $i \neq I$, the constraint $\bar{t} \geq g_i(\bar{z}) - M$ holds since $\bar{t} = g_I(\bar{z}) = \min\limits_{i=1,\dots,n} \{g_i(\bar{z}) \geq 0 \geq g_i(\bar{z}) - M\}$ with M is sufficiently large. For $i = I$, the ith constraint of (P_2) reads $\bar{t} = g_I(\bar{z}) \geq g_I(z) - M\bar{y}_I = g_I(\bar{z})$. Hence $(\bar{z}, \bar{y}, \bar{t})$ is feasible for (P_2).

Let us now show that: **if (z^*, y^*, t^*) is an optimal solution of (P_2) then (z^*, t^*) is feasible for (P_1'), i.e.,** $t^* = \min\limits_{i=1\dots,n} \{g_i(z^*)\}$.

By contradiction, let us suppose that this optimal solution of (P_2) is such that $t^* \neq \min_{i=1....,n}\{g_i(z^*)\}$. Let us consider two cases: either $t^* < \min_{i=1....,n}\{g_i(z^*)\}$ or $t^* > \min_{i=1....,n}\{g_i(z^*)\}$.
Let I_{y^*} denote the index i such that $y_i^* = 0, \text{then } y_{I_{y^*}} = 0$ and $y_i^* = 1$, for all $i \neq I_{y^*}$. Using the fact that (z^*, y^*, t^*) is a feasible solution for (P_2), we have

$$t^* \geq g_{I_{y^*}}(z^*). \tag{5.12}$$

In the first case, we have $t^* < \min_{i=1....,n}\{g_i(z^*)\}$. Then we have $g_{I_{y^*}}(z^*) \geq \min_{i=1....,n}\{g_i(z^*)\} > t^*$, which contradicts (5.12).

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

$$\begin{aligned} \bar{z} &= z^*, \\ \bar{t} &= \min_{i=1....,n}\{g_i(z^*)\}, \end{aligned} \tag{5.13}$$

and

$$\bar{y}_i := \begin{cases} 0, & \text{if i is the smallest index s.t } g_i(z^*) = \min_{i=1....,n}\{g_i(z^*)\}, \\ 1, & \text{otherwise.} \end{cases} \tag{5.14}$$

Let I^* denote this smallest index i such that $g_i(z^*) = \min_{i=1....,n}\{g_i(z^*)\}$. This new solution $(\bar{z}, \bar{y}, \bar{t})$ is

- feasible for (P_2). Indeed, consider the two cases: if $i \neq I^*$, the i^{th} constraint $t \geq g_i(z) - My_i$, is satisfied since M is an upper bound for the g_i and $\bar{y}_i = 1$. If $i = I^*$, then on the one hand $\min_{i=1....,n}\{g_i(z^*)\} = \bar{t}$ and on the second hand $g_{I^*}(\bar{z}) = g_{I^*}(z^*) = \min_{i=1....,n}\{g_i(z^*)\}$ by definition of I^*. Therefore, the I^*th constraint of (P_2) is satisfied by $(\bar{z}, \bar{y}, \bar{t})$.
- In term of objective function values, it is clear that $f(z^*) + \mu t^* > f(\bar{z}) + \mu\bar{t}$ since by hypothesis $t^* > \min_{i=1....,n}\{g_i(z^*)\}$while $\bar{t} = \min\{g_i(z^*)\}$. This contradicts the optimality of (z^*, y^*, t^*).

□

The original model to minimize is $m(z)$. To use the proposition (5.16), we build a perturbed model $m_\epsilon(z) = m(z) + \epsilon$ and prove that, if $m(z)$ is fully linear, then the perturbed model $m_\epsilon(z)$ is also fully linear, which guarantees the convergence of the algorithm.

Assumption 5.17. *We assume that $Z = \{z_0, \ldots, z_n\} \subset \mathbb{R}^m \times \{0,1\}^n$ is a set of sample points poised in the linear interpolation sense (or regression sense).*
We assume that the function f is continuously differentiable in an open domain Ω containing $B(z_0, \Delta)$ and ∇f is Lipschizt continuous in Ω with the Lipschitz constant $\nu > 0$.

Problem 5.1. *(MIQP with original model)*

$$\begin{cases} \min\limits_{x,y} m(x,y) \\ \min\limits_{i=1,\ldots,n}(g_i(y)) \leq \Delta_{y,k}, \\ \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ x \in \mathbb{R}^m, y \in \{0,1\}^n, \end{cases} \tag{5.15}$$

where $g_i, i = 1, \ldots, n$ is the Hamming distance $g_i(y) = d_H(y, Rot^i(y_k))$, y_k is the binary parts of the trust region centre. By adding a slack variable t the problem (5.15) is equivalent to

$$\begin{cases} \min\limits_{x,y,t} m(x,y) \\ t = \min\limits_{i=1,\ldots,n} (g_i(y)), \\ t \leq \Delta_{y,k}, \\ \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ x \in \mathbb{R}^m, y \in \{0,1\}^n, t \in \mathbb{Z}_+ \end{cases} \tag{5.16}$$

Our goal is to introduce the necklace distance inside this MIQP. Thus, we propose:

- To perturb the model without impacting the fully linear property of the original model (to ensure the convergence of the algorithm)
- to use exact reformulation in proposition (5.16) to obtain linear constraints, i.e. to avoid the operator "min" in the new distance formulation.

First, the perturbed problem is defined as

Problem 5.2.

$$\begin{cases} \min\limits_{x,y,t} m(x,y) + \mu t \\ t = \min\limits_{i=1,\ldots,n} (g_i(y)) \\ t \leq \Delta_y \\ \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ x \in \mathbb{R}^m, y \in \{0,1\}^n, t \in \mathbb{Z}_+ \end{cases} \tag{5.17}$$

with $\mu = \epsilon/(n-1)$ in order to ensure $\mu t \leq \epsilon$, ϵ is a given small constant. By using exact reformulation in proposition (5.16) we obtain

$$\begin{cases} \min\limits_{x,y,\tilde{y},t} f(x,y) + \mu t \\ s.t.\ t \geq g_i(y) - M\tilde{y}_i, \forall i = 1, \ldots, n \\ \quad \|x - x_k\|_\infty \leq \Delta_{x,k}, \\ \quad \sum_{i=1}^n \tilde{y}_i = n - 1, \\ \quad x \in \mathbb{R}^m, y \in \{0,1\}^n, t \in \mathbb{Z}_+, \tilde{y}_i \in \{0,1\}, \end{cases} \tag{5.18}$$

5.2.4 Fully linear property of the perturbed model

We will indicate that if $m(x,y)$ is fully linear, then with a given ϵ small enough, we still have fully linear property for the new model $m_\epsilon(x,y) = m(x,y) + \epsilon$. To do this, we use some fundamental theorems of calculus, see Appendix (B).

To be convenient, we denote

$$err^g = \nabla m_\epsilon(z) - \nabla f(z)$$
$$err^f = m_\epsilon(z) - f(z)$$

Now we will show that m_ϵ is fully linear for a small perturbation ϵ if m is fully linear.

Theorem 5.18. *Let assumption (4.3) hold. Then if a quadratic model $|m_\epsilon(z_i) - f(z_i)| \leq \epsilon$ for all points in $B(z_0, \Delta)$ we have*

- *The error between the gradient of the interpolating quadratic model and the gradient of the function satisfies*

$$\|err^g\| = \|\nabla m_\epsilon(z) - \nabla f(z)\| \leq \sqrt{n}\Delta\|\hat{Z}^{-1}\|(\frac{5}{2}\nu + \frac{5}{2}\|H\|_F + \frac{1}{\Delta^2}2\epsilon) \tag{5.19}$$

- *The error between the interpolating quadratic model and the function satisfies*

$$|err^f| = |m_\epsilon(z) - f(z)| \leq \Delta^2(\nu + \|H\|_F)\frac{5\sqrt{n}\|\hat{Z}\|^{-1} + 1}{2} + \epsilon(2\sqrt{n}\|\hat{Z}\|^{-1} + 1) \tag{5.20}$$

Proof. First, we expand

$$\begin{aligned}(z_i - z)^T err^g(z) &= (z_i - z)^T(\nabla m_\epsilon(z) - \nabla f(z)) \\ &= (z_i - z)^T(Hz + g - \nabla f(z)) \\ &= (z_i - z)^T Hz + (z_i - z)^T g - (z_i - z)^T \nabla f(z) \\ &= (z_i - z)^T Hz + (z_i - z)^T g - f(z_i) + f(z) + (f(z_i) - f(z) - (z_i - z)^T \nabla f(z))\end{aligned} \tag{5.21}$$

Set $z_i - z = d$. Using lemma (B.1), equation (5.21) is equivalent to

$$\begin{aligned}(z_i - z)^T err^g(z) &= \int_0^1 d^T(\nabla f(z + td) - \nabla f(z))dt - f(z_i) + f(z) + (z_i - z)^T Hz + (z_i - z)^T g \\ &= \int_0^1 d^T(\nabla f(z + td) - \nabla f(z))dt + err^f(z_i) - err^f(z) - \frac{1}{2}d^T Hd\end{aligned} \tag{5.22}$$

The next step is to subtract for z_0

$$\begin{aligned}(z_i - z_0)^T err^g(z) &= (z_i - z)^T err^g(z) - (z_0 - z)^T err^g(z) \\ &= \int_0^1 (z_i - z)^T(\nabla f(z + t(z_i - z)) - \nabla f(z))dt \\ &- \int_0^1 (z_0 - z)^T(\nabla f(z + t(z_0 - z)) - \nabla f(z))dt \\ &+ err^f(z_i) - err^f(z_0) - \frac{1}{2}(z_i - z)^T H(z_i - z) + \frac{1}{2}(z_0 - z)^T H(z_0 - z)\end{aligned} \tag{5.23}$$

Using both lemma (B.1, B.2) and the definition of m_ϵ we have

$$err^f = |m_\epsilon(z_i) - f(z_i)| \leq \epsilon, \forall z_i \in Z.$$

Combining this result with the relation

$$(z_i - z)^T H(z_i - z) \leq \|H\|_F\|z_i - z\|^2,$$

and the fact that $\|z - z_i\| \leq 2\Delta, \|z - z_0\| \leq \Delta$, we find the bound

$$|(z_i - z_0)^T err^g(z)| \leq \frac{5}{2}\nu\Delta^2 + \frac{5}{2}\|H\|_F\Delta^2 + 2\epsilon. \tag{5.24}$$

Setting $\hat{Z} = \frac{1}{\Delta}(z_1 - z_0, \ldots, z_n - z_0)$, we have

$$\hat{Z}^T err^g(z) = \frac{1}{\Delta} \begin{pmatrix} (z_1 - z_0)^T \\ (z_2 - z_0)^T \\ \cdots \\ (z_n - z_0)^T \end{pmatrix} err^g(z_0). \tag{5.25}$$

In particular,

$$\begin{aligned} \|\hat{Z}^T err^g\|_\infty^2 &= \frac{1}{\Delta^2} \max_i ((z_i - z_0) err^g(z))^2 \\ &\leq \frac{1}{\Delta^2} (\nu \|z_i - z_0\|^2 + \|H\|_F \|z_i - z_0\|^2 + 2\epsilon)^2 \\ &\leq \frac{1}{\Delta^2} \max_i ((z_i - z_0) err^g(z))^2 \\ &\leq \frac{1}{\Delta^2} (\frac{5}{2}\nu\Delta^2 + \frac{5}{2}\|H\|_F\Delta^2 + 2\epsilon)^2 \\ &= \Delta^2 (\frac{5}{2}\nu + \frac{5}{2}\|H\|_F + \frac{1}{\Delta^2}2\epsilon)^2 \end{aligned} \tag{5.26}$$

Noting that $\|A\| = \|A^T\|$, we have

$$\begin{aligned} \|err^g(z) &= \|\hat{Z}^{-T}\hat{Z}^T(\nabla m_\epsilon(z) - \nabla f(z_0))\| \\ &\leq \|\hat{Z}^{-T}\| \|\hat{Z}^T err^g(z)\| \\ &\leq \sqrt{n}\|\hat{Z}^{-T}\| \|\hat{Z}^T err^g(z)\|_\infty \\ &\leq \sqrt{n}\Delta\|\hat{Z}^{-1}\| (\frac{5}{2}\nu + \frac{5}{2}\|H\|_F + \frac{1}{\Delta^2}2\epsilon) \end{aligned} \tag{5.27}$$

Taking the norm both sides of the equation with $z = z_0$, we have

$$\begin{aligned} &|err^f(z)| \leq \|err^g(z)\|\Delta + \frac{1}{2}\nu\Delta^2 + \frac{1}{2}\|H\|_F\Delta^2 + 2\epsilon + |err^f(z_0)| \\ \implies &|err^f(z)| \leq \sqrt{n}\Delta^2\|\hat{Z}^{-1}\|(\frac{5}{2}\nu + \frac{5}{2}\|H\|_F + \frac{1}{\Delta^2}2\epsilon) + \frac{1}{2}\Delta^2(\nu + \|H\|_F) + \epsilon \\ &= \Delta^2(\nu + \|H\|_F)\frac{2\sqrt{n}\|\hat{Z}\|^{-1} + 1}{2} + \epsilon(5\sqrt{n}\|\hat{Z}\|^{-1} + 1) \end{aligned} \tag{5.28}$$

□

Remark 5.2. We note that if the value of Δ is decreased significantly, the term $\frac{\epsilon}{\Delta^2}$ in the bound (5.19) tends to infinity which cause the divergence of the algorithm. In order to avoid that, we can control the value of ϵ if we take the value of $\mu = \frac{\Delta^2}{n-1}\epsilon'$ then the perturbation is bounded by ϵ'.

Remark 5.3. With the value $\epsilon = 0$, we obtain the bound in Theorem 5.4 (page 79, [15])

In this chapter, we proposed an appropriate distance adapted to the cyclic symmetric problems. Its integration in DFO trust region algorithm is shown along with a convergence proof.

By adapting the necklace distance to DFO trust region method, thanks to the exploration of distinct binary arrangements, we would also expect to obtain a good performance. In the next chapter, we present some numerical results obtained with the presented adapted trust region method applied to cyclic symmetric mixed binary problems.

Chapter 6

Application of DFOb-TR on mixed binary cyclic problems

In the previous chapter, we introduced the adapted DFOb trust region method with the necklace distance for cyclic symmetric problems.

The distribution of this chapter is as follows: the first section gives an illustration of DFOb in action with one given example. In the next section, we detail the methodology for applying our method and comparing with state-of-the-art methods. In the last section, numerical results of DFOb-d_{neck}, DFOb-d_H, NOMAD, RBFopt are presented.

6.1 DFOb in action with one given example

DFOb algorithm is presented in Chapter 4. To clarify how does the algorithm work step by step we give an illustration of running DFOb for a given example. We choose MAD1 function from benchmark set, details in Appendix (C).

In Figure (6.1), we plot the function for the three distinct binary configurations.

This function has two continuous variables, range in $[-2, 6]$, three levels (that we transform into binaries by using the trick in (6.2) that will be discussed in the Subsection 6.3): the first level for $y = [0, 0]$ is polynomial of degree 2 with one global solution, the second level for $y = [1, 0]$ or $y = [0, 1]$ is sin function with bunch of solutions reached at $x_1 = \pi/2 \, \text{or} \, 3\pi/2$, the last level for $y = [1, 1]$ is cos function where the solution is reached at $x_2 = 0$.

We start from 5 starting points. The initial design is required to satisfy two properties which are space-filling and non-collapsing, the methodology details will be described in the Subsection (6.2.1).

After building models, we solve QP subproblems (in step 1.0) and MBQP subproblems (step 1.5) which require continuous variables to satisfy the trust-region constraints, thus, in Figure (6.1) except initial sample set, all other points is closed (distance is less than $\Delta^x_{initial}$).

We see that DFOb-d_{neck} discovers the global solution in the first level, since the quadratic model is appropriate for performing a polynomial of degree 2.

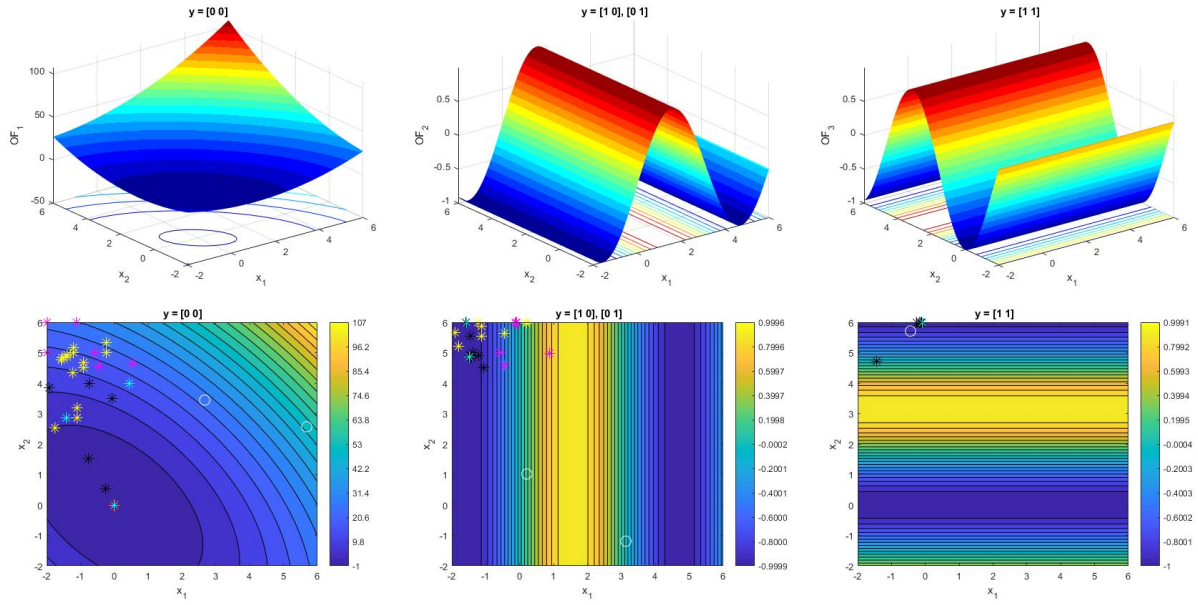

Figure 6.1: Function with three levels and Simulation locations along DFOb-d_{neck} for benchmark function MAD1, [35]: points of the initial design are represented by white circles, the solutions for step (1.0) by black asterisks, the points associated with model improvement step by magenta asterisks, the added points in step (1.5) by yellow asterisks, the local solutions by cyan asterisks and the global solution by a red diamond

6.2 Methodology for method comparison

6.2.1 Initial design of experiments

The first step in any DFO methods is to choose a set of starting points. The first model is built based on cost function evaluations at these points. A "good" design of experiments (DOE) should satisfy the following requirements:

- Space-filling property: the set of points should spread over the entire design space.
- Non-collapsing property: two arbitrary points should not share the same coordinate value.

There are several methods to pick up a choice of initial sample points in continuous space. When the dimension is small, to select the initial sample points we pick the 2^n corner points of the box constraint. [24] use a strategy to choose $n+1$ corner points of the box constraint and the central points. For larger dimensions, a popular strategy is called Latin Hypercube Sample (LHS). In our implementation, the chosen sampling procedure is based on LHS of size $n+m+1$. We note that LHS originally is used for generating samples in continuous space, $\mathbb{R}^m$, the procedure to create a sample set of size N is described is in Algorithm 4, see [36, 48] for details.

Algorithm 4: Uniform Latin Hypercube sampling of size N in $\mathbb{R}^m$

1. For $i = 1, \ldots, N$
   - For $j = 1, \ldots, m$
     Generate $U \backsim$ Uniform$[0, 1]$ and
     Set $X_{i,j} = \dfrac{U_i - 1}{N}$
   - Randomly permute elements of the ith row of $X, [X_{i,1}, \ldots, X_{i,m}]$
2. Output each of the N rows of the matrix X as sample point

Figure (6.2) shows an illustration of LHS design for continuous variables applying to problem QL, Appendix C, the range is $[-2, 6]$

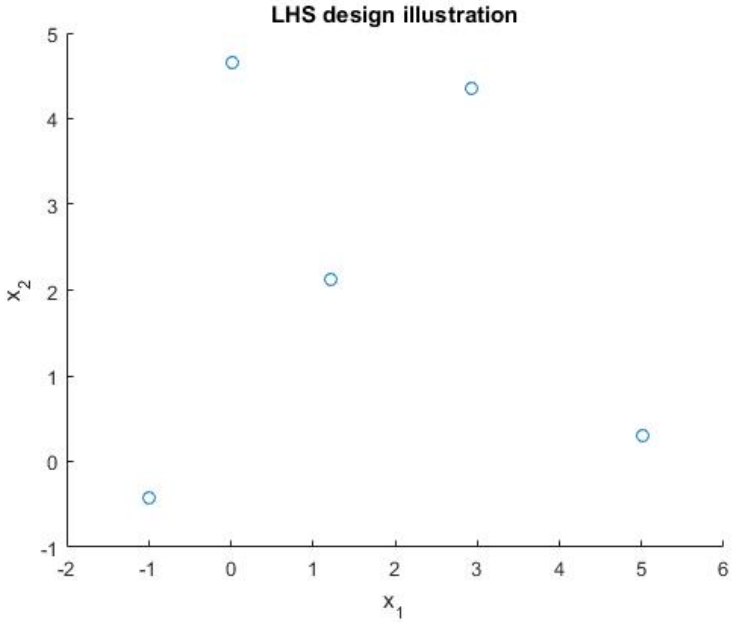

Figure 6.2: LHS design illustration for function with 2 continuous variables.

For binary part, we might modify this methodology adapt to binary domain. More precisely, to build a set of N binary (with the size of n) initial points: $y^i \in \{0, 1\}^n, i = 1, \ldots, N$ first by using LHS we generate a sample set with N continuous points, $x^i \in \mathbb{R}^n, i = 1, \ldots, N$. After this step, the values of each element of x^i is between 0 and 1. By rounding each element of x^i to the nearest integer, either 0 or 1, we obtain the binary initial sample set $LHS(y, n, N)$.

Rounding can recover integer domain but may destroy the properties of LHS, or even worse, it can provide several times the same points. Thus, a next study will be dedicated to improve this initial design procedure.

6.2.2 Compared methods

We compare our algorithm with direct search (NOMAD), [5, 31], and model-based (RBFopt), [16] algorithms. We consider only solvers that are designed to solve mixed integer black-box optimization problems. We use NOMAD because this method is popular among application scientists and it can solve a wide range of non-linear optimization problems. NOMAD solver requires one starting point z_0, i.e., chosen as the point associated to the best objective function in DFOb trust-region method. Remark that NOMAD can handle linear and nonlinear constraints.

RBFopt was first introduced in [24]. It is based on Radial Basis Functions to build surrogate models. The exploration and exploitation decision depends on a measure of bumpiness, thus it requires that the unknown function f does not oscillate too much. RBFopt was not designed for constrained optimization problems (only box constraints) , linear constraints are handled by penalization in our implementation.

Remember that both algorithms can handle binary variables.

6.2.3 Evaluation methodology

Our evaluation based on performance profiles and data profiles. We define a maximum number of function evaluations. The budget is set to 300 simulations in our implementation , it is reasonable number for many real applications.

A solver solves a problem if it returns a point $\bar{x}$ satisfying the following criterion

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

with f^* be the best function value found by any solver, x_0 the starting point for each solver, τ tolerance value, we use $\tau = 10^{-3}$. A solver fails otherwise.

For a set of n_p problems $P = \{p_1, p_2, \ldots, p_{n_p}\}$, and the set of n_s solvers $S = \{s_1, s_2, \ldots, s_{n_s}\}$, we define the performance criterion for a solver s, a problem p and a required precision tol by

$$t_{p,s} = \text{number of simulations required for s to solver p at precision tol.}$$

$t_{p,s} = \infty$ if solver s fails on solving problem p.

A performance ratio over all the solvers is defined by

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s}, s \in S\}} \geq 1, \text{for a given problem } p.$$

For $\eta \geq 1$, we define a distribution function ρ_s for the performance ratio for a solver s as

$$\rho_s(\eta) = \frac{1}{n_p} card\{p \in P, r_{p,s} \leq \eta\} \leq 1, \text{for a given solver } s,$$

with *card* denotes the cardinal of a set. This distribution computes the number of problems p that are solved with a performance ratio below a given threshold η. If the value of $\rho_s(\eta)$ is near 1, it means that the solver is good and can solve almost all the problems. The performance profiles is a graph of the function $\rho_s(\eta), \eta > 1$.

Performance profiles provide an accurate view of the relative performance of solvers with a given number of simulations. However, for expensive optimization problems it does not provide

sufficient information.

With expensive optimization problems, we are interested in the performance of solvers as a function of the number of function evaluations which leads to the definition of data profiles. The data profile for a solver s is the fraction of problems that are solved within a fixed simulation budget κ

$$d_s(\alpha) = \frac{1}{n_p} card\{p \in P : \frac{t_{p,s}}{n_v + 1} \leq \kappa\},$$

with n_v the number of variables of problem p. It is normalized by $n_v + 1$ since the number of simulations grows when the number of variables increase.

We present results of DFOb-d_{neck} in comparing to NOMAD, RBFopt and DFOb-d_H applied to benchmark functions, a toy problem that is closed to the application and a simulation given by Safran Tech. We indicate values of parameters using in optimization procedure, in Table (6.1).

Table 6.1: Parameters using in DFOb-TR

nsimul	nogoodcut max	Δ_x^0	Δ_y^0
300	$\min(14, 2^n - 1), n \leq 6, 20$ if $n = 12$	1	2

Note that for the simulation given by Safran, we would use the default option of RBFopt to generate its best initial design. DFOb-d$_{neck}$, DFOb-d$_H$ share the same initial sample set. NOMAD uses the the best point in our initial design. For benchmark functions, initial designs are taken the same for the 3 optimizers: RBFopt, DFOb-d_H and DFOb-d_{new}.

6.3 Benchmark functions

List of functions

Our benchmark consists in 25 different box-constrained problems from [1, 18, 25, 35, 38] and GLOBALLIB, listed in table 6.2.

Note that these benchmark functions are originally for continuous optimization (some include constraints, but we consider only box constraints). Thus, we suitably transform them into mixed binary problems with cyclic symmetry.

In [35], benchmark functions are associated with minimax problems for continuous optimization

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

We transform them into mixed categorical problems:

$$\min_{x \in [\underline{x}, \bar{x}], 1 \leq w \leq l} \tilde{F}(x, w) := \begin{cases} f_1(x) \text{ if } w = 1, \\ f_2(x) \text{ if } w = 2, \\ \dots \\ f_l(x) \text{ if } w = l. \end{cases} \tag{6.1}$$

Table 6.2: Benchmark functions

Test names	# continuous (m)	# binary (n)	Domain	Reference
CB1	2	2	[-2,6]	[35]
CB2	2	2	[-2,6]	[35]
QL	2	2	[-2,6]	[35]
WF	2	2	[1,6]	[35]
MAD1	2	2	[-2,6]	[35]
MAD1	2	2	[-2,6]	[35]
RosenSuzuki	4	3	[-2,6]	[35]
Pentagon	6	3	[0,2]	[35]
Wong2	10	4	[0,2]	[35]
Wong3	20	6	[0,2]	[35]
HS2	1	3	[-5,5]	[25]
HS3	1	3	[-5,5]	[25]
HS29log	1	3	[-5,5]	[25]
Branin	1	3	[-5,10]	[18]
Camel	1	3	[-3,3]	[18]
Goldstein-Price	1	3	[-2,2]	[18]
Hartman3	2	3	[0,1]	[18]
Hartman6	5	3	[0,1]	[18]
Shekel7	3	3	[0,10]	[18]
Shekel10	3	3	[0,10]	[18]
ex8-1-1	1	3	[-2,4]	GLOBALLIB
ex8-1-4	1	3	[-1,2]	GLOBALLIB
Perm6	5	3	[-6,6]	[38]
Perm8	7	3	[-1,1]	[38]
sporttournament	14	3	[0,1]	[1]

Instead of using the standard encoding of integer variables into binary variables:

$$w = \sum_{i=0}^{h} 2^i y_i,\ h = floor(\log 2(n) + 1),$$

we use the following trick to introduce a cyclic symmetry,: each categorical variable is associated with a necklace and its rotations. For instance, in case of 3 levels for an integer variable, $l = 3$, the problems are transformed as follows

$$\min_{x\in[\underline{x},\bar{x}],y\in\{0,1\}^2} \tilde{F}(x,y) := \begin{cases} f_1(x) \text{ if } y = (0,0), \\ f_2(x) \text{ if } y = (0,1) \text{ or } (1,0), \\ f_3(x) \text{ if } y = (1,1). \end{cases} \tag{6.2}$$

For benchmark functions of [1, 18, 25] and GLOBALLIB, we restrict the last continuous

variable to take only a finite number of values: $x^{end} \in X^{end}$ with

$$X^{end} = \left(\underline{x}^{end} + h \frac{\bar{x}^{end} - \underline{x}^{end}}{n_{level} - 1}, \text{ for } h = 0, \ldots, n_{level} - 1 \right).$$

where $\underline{x}^{end}$ and $\bar{x}^{end}$ respectively the lower and upper bound of original x^{end}. After this step, we obtain one integer variable of n_{level} (equal to 4 in our implementation). The last step is to apply the above trick, example in (6.2), to build cyclic symmetric problems.

The dimension of these test functions range from 2 to 20 in their original formulation, range from 4 to 27 for the new necklace formulation. For [25], functions are smooth while functions from [1,18,35,38] and GLOBALLIB are more complex with several local minima. With an effort of building a benchmark with a large range in the number of variables and different complexities of functions, we aim to study the behavior of our algorithm in comparison with NOMAD and RBFopt algorithms.

Preliminary results for benchmark functions

The results of running DFOb with necklace distance $DFOb - d_{neck}$, DFOb with Hamming distance $DFOb - d_H$, NOMAD and RBFopt over 25 benchmark functions are presented by data profiles, performance profiles, mean relative errors on $F(\%)$ compared to the best values among results (all the solvers) and number of iterations necessary to reach the best points, see in Figures (6.3, 6.4, 6.5,6.6). For all examples, DFOb-d_H are displayed by red lines, DFOb-d_{neck} by blue lines, NOMAD by black lines, RBFopt by cyan lines.

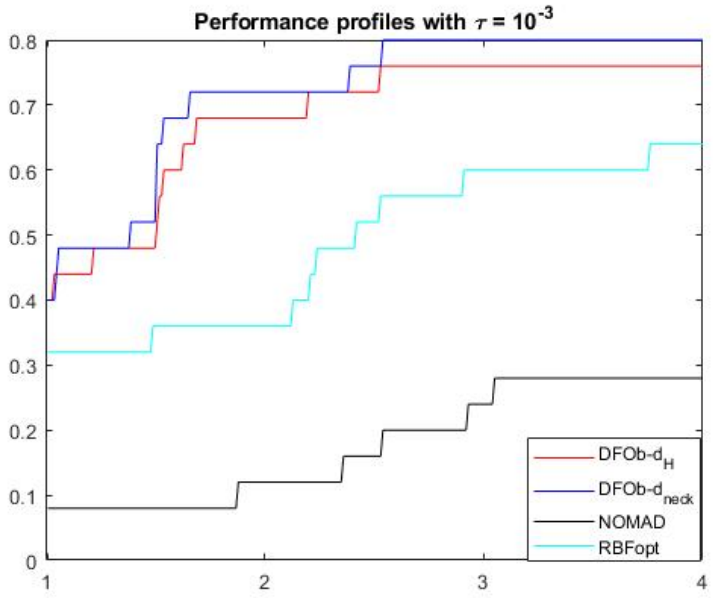

Figure 6.3: Performance profiles for 25 benchmark functions of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt

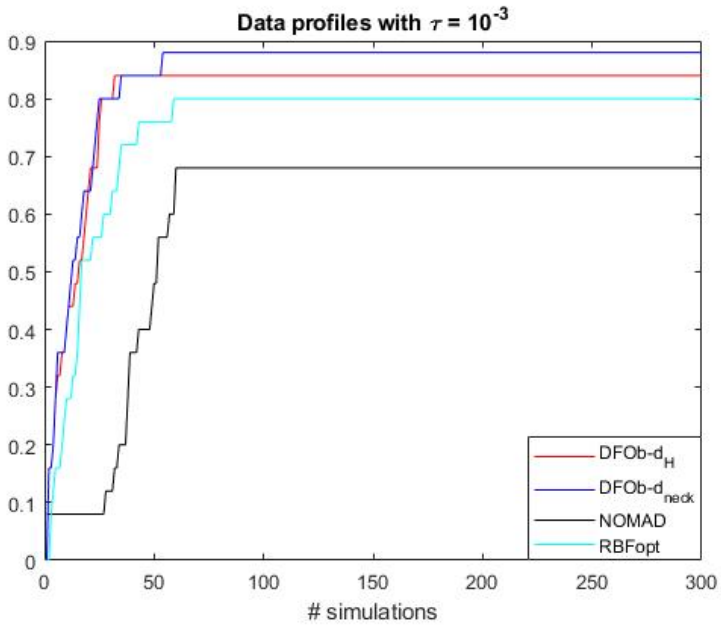

Figure 6.4: Data profiles for 25 benchmark functions of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt

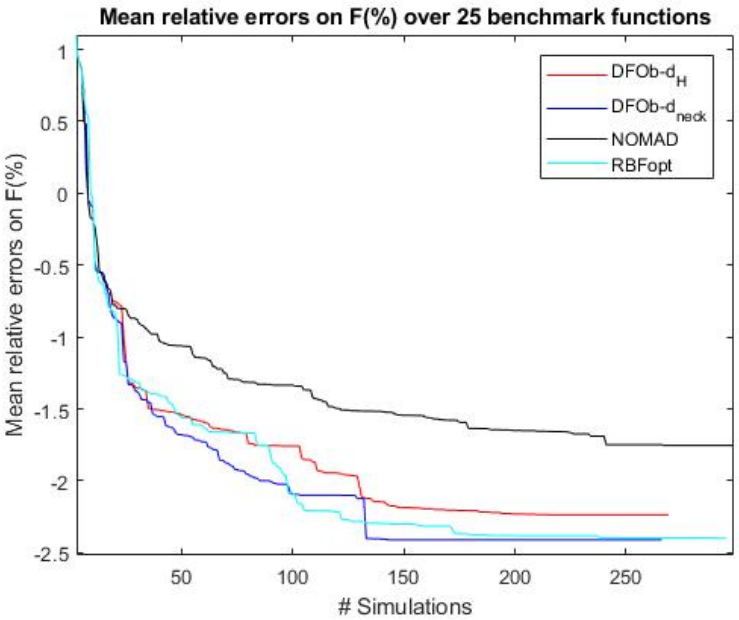

Figure 6.5: Mean relative errors (in log scale) on F(%) over the 25 benchmark functions of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt

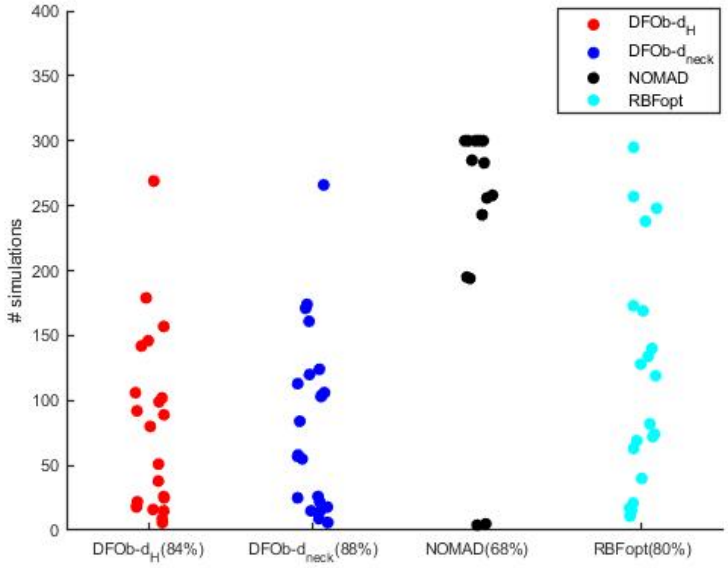

Figure 6.6: Number of simulation necessary to reach the best points over the 25 benchmark functions of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt

For the benchmark of 25 ebnchmark functions, DFOb-d_{neck} method outperforms the 3 other optimizers including NOMAD, RBFopt, DFOb-d_H: it succeeds to solve 88% of runs, whereas, DFOb-d_H, NOMAD, RBFopt succeeds to solve respectively $84\%, 68\%$ and 80% runs.

Performance profiles are reported in Figure (6.3). The results show that DFO-d_{neck} has big progress compare to DFOb-d_H. It is highly competitive with comparing solvers.

We report data profiles in Figure (6.4). We can see that DFOb-d_{neck} has the best performance in solving expensive simulators. Figure (6.5) and Figure (6.6) displays the median relative error for objective functions and the number of simulation necessary to reach the best points of 4 optimizers over 25 benchmark functions. As can be seen in Figure (6.5) all configurations of 4 solvers coincide on relative errors fr the first few data points, and they start to diverge after $\backsim$ 15 iterations, then at the end they coincide again since reaching of the best solutions. Figure (6.6) shows that DFOb-d_{neck}, DFOb-d_H and RBFopt use reasonable number of iterations to reach best points, whereas NOMAD requires much larger number of simulation.

Running over benchmark functions also highlight the fact that the optimizers will have different performance on different problems. It is important also to think of the compromise between saving simulations and finding the best solution. If a solver can find a better solution for an optimization problem in an acceptable number of simulations, we consider it has better performance for this problem. In derivative free optimization, since the computationally expensive of the simulations, we would always limit the number of simulations.

6.4 Toy problem closed to SAFRAN's application

We build a toy problem closed to Safran's application as described in Appendix A. It results in minimizing the maximal amplitude of the excitation of the system with respect to a continuous

parameter (frequency) and a binary vector (mistuning blade locations)

$$\begin{aligned} \underset{\omega,y}{\text{minimize}} \quad & \|A\|_\infty \\ \text{subject to} \quad & \omega \in [\omega_{min}, \omega_{max}], \\ & y \in \{0,1\}^n, \end{aligned} \tag{6.3}$$

where n denotes the number of blades on the disk.

Figure (6.7, 6.8, 6.9, 6.10) report the performance profiles, data profiles, mean best function values and number of iterations to reach the best points for 10 randomly chosen initial design of experiments.

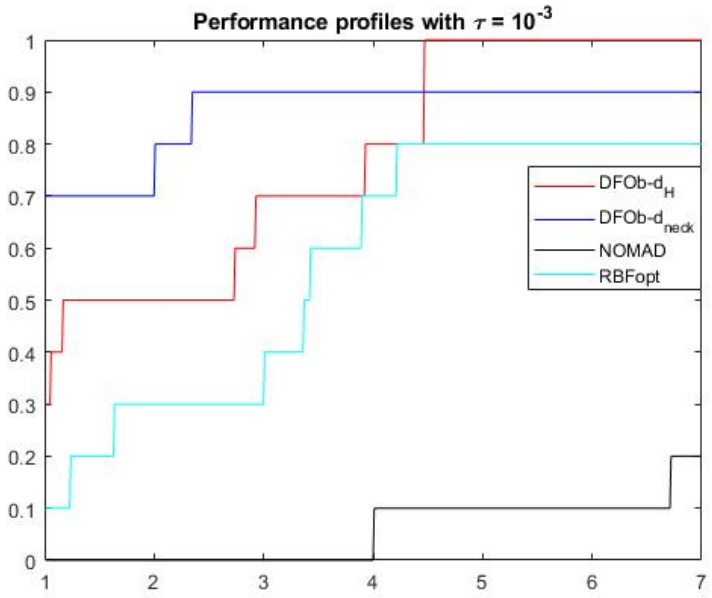

Figure 6.7: Performance profiles for toy problem of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt with 10 random initial designs

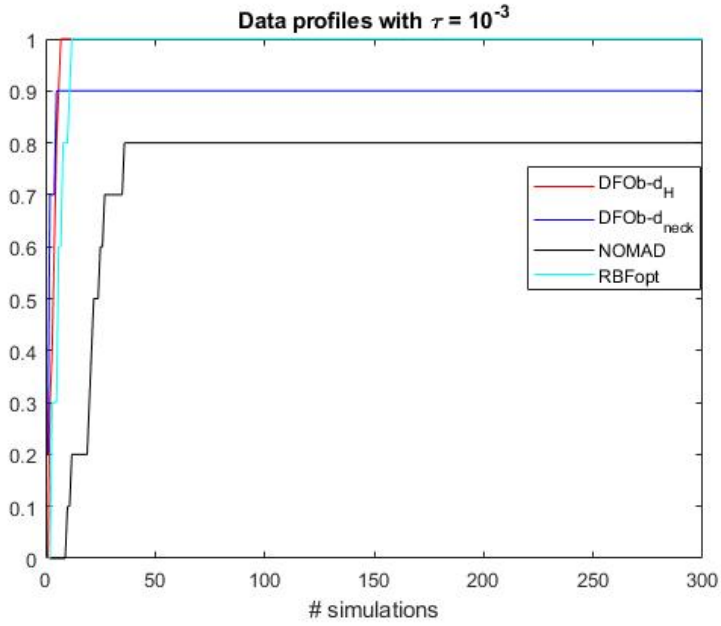

Figure 6.8: Data profiles for toy problem of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt with 10 random initial designs

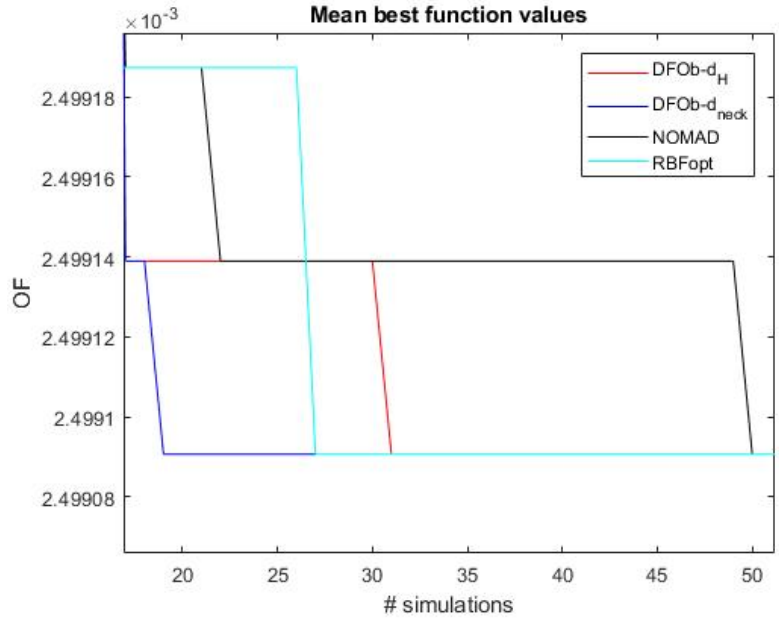

Figure 6.9: Mean relative errors on F(%) for toy problem of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt with 10 random initial designs

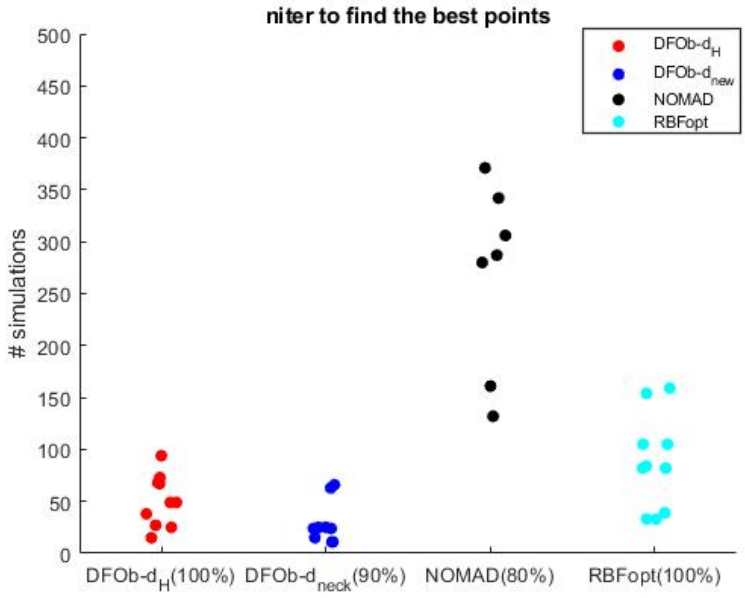

Figure 6.10: Number of iteration to reach the best points for toy problem of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt

For this example, DFO-d_{neck} has the second rank in data profiles and performance profiles after DFOb-d_H. However, Figure (6.9) and Figure (6.10) illustrate that DFOb-d_{neck} is more efficient in terms of number of simulations.

6.5 Safran's application

Safran's application was provided by means of a simplified simulation (with 12 blades on the disk) based on a response surface model to avoid the huge computational cost for these preliminary tests. Recall that the objective is to minimize the vibration of the compressor, a continuous variable describe the blade frequency (which cause from shape parameters such as the thickness of the axis, the length of the blades) and binary variables locate the two pre-defined blade geometries (tuning and mistunig) on the disk.

In this part, we present results on Safran's application. Figure (6.11, 6.12, 6.13, 6.14) report the Performance profiles, Data profiles, Mean best function values and number of simulations to reach the best points for 10 randomly chosen initial design of experiments.

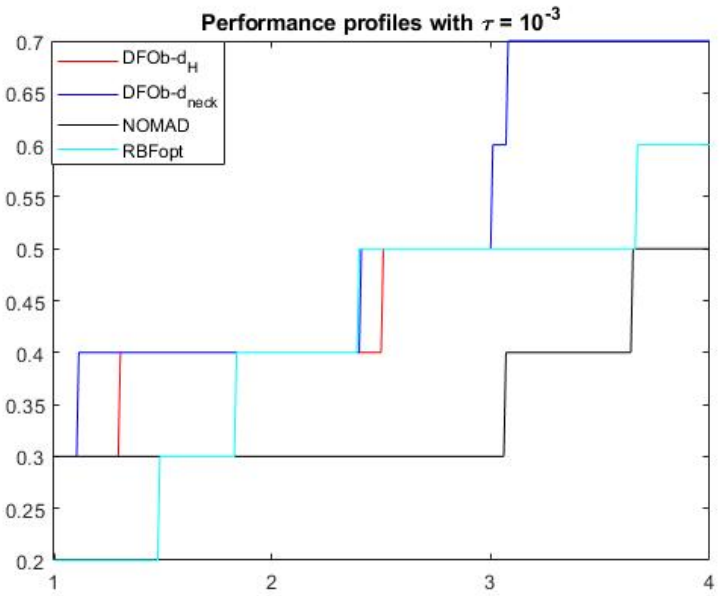

Figure 6.11: Performance profiles for Safran's simulation of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt with 10 random initial designs.

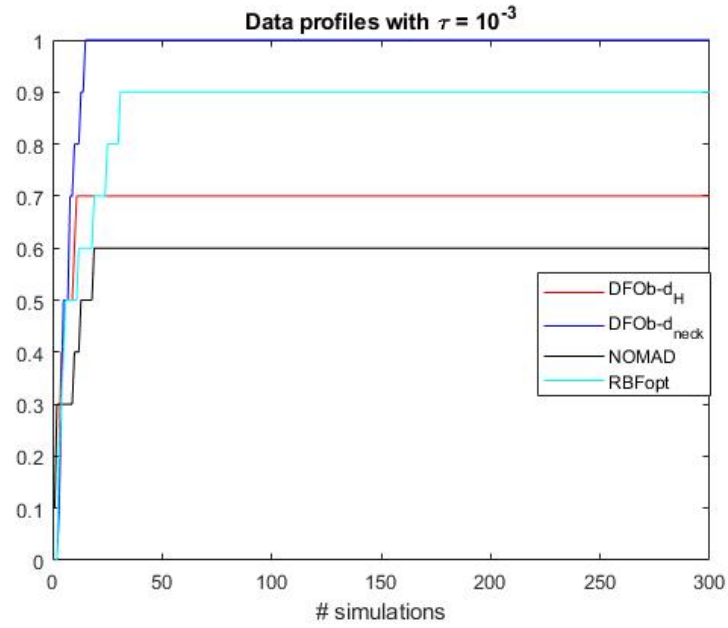

Figure 6.12: Data profiles for Safran's simulation of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt with 10 random initial designs.

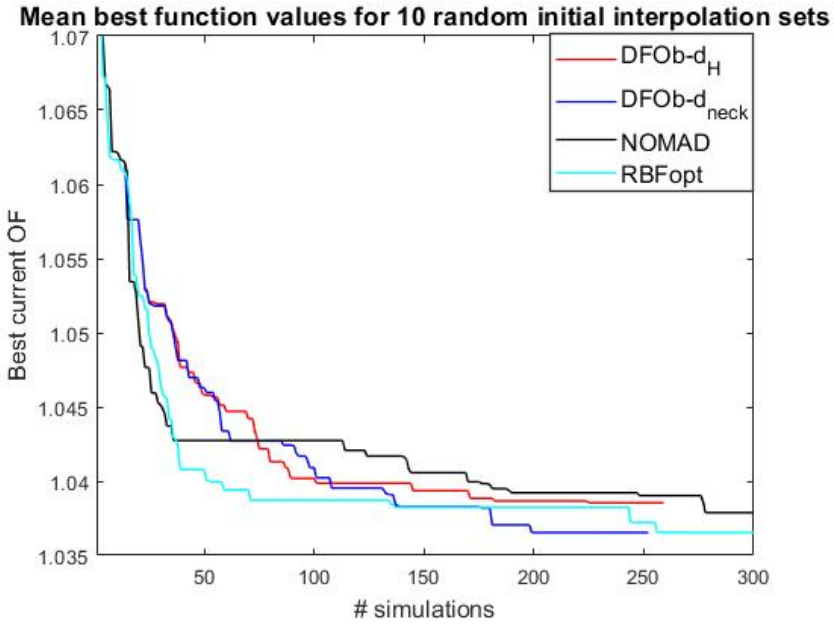

Figure 6.13: Mean relative errors on F(%) for Safran's simulation of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt with 10 random initial designs.

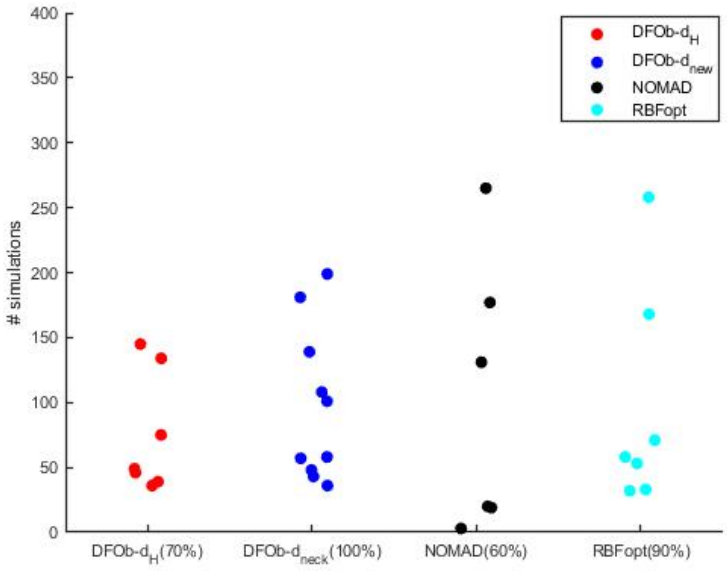

Figure 6.14: Number of simulation necessary to reach the best points for Safran's simulation of DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFopt with 10 random initial designs.

For the simplified simulation, DFOb-d_{neck} method outperforms the 3 other optimizers: NOMAD, RBFopt, DFOb-d_H. It succeeds to solve 100% of runs, whereas, DFOb-d_H, NOMAD, RBFopt succeeds to solve respectively $70\%, 60\%$ and 90% runs.

Performance profiles are reported in Figure (6.11). The results show the improvement of

DFO-d_{neck} compared to DFOb-d_H.

We report data profiles in Figure (6.12). We can see that DFOb-d_{neck} has the best performance in terms of number of simulations. Figure (6.13) shows the mean best function over 10 runs: DFOb-d_{neck} succeeds to reach the best mean values in the smaller number of simulations. Figure (6.14) displays the number of simulations necessary to reach the best values. It shows that DFO-d_{neck} succeeds to find the best points in all the runs in less than 210 simulations.

Chapter 7

Conclusion and perspectives

In this report, we have addressed derivative-free optimization problems when the cost function is computationally expensive. In Chapter 4, we showed how to extend a general framework (for continuous optimization) to mixed binary quadratic models. It is difficult to find the details of dealing with binary variables in the literature of derivative free optimization. Most of them give general comments or ideas of dealing with binary variables without special or dedicated methods are proposed. In the knowledge of the author, [15] is the only book that gives the proof about the convergence (local) of Derivative free trust region method for continuous variables.

The details of real application given by SAFRAN is presented in Chapter 3. Along with the concrete problem, the cyclic symmetry property in binary variables appear tends us to adapt our algorithm. By using the concept of Necklace, in Chapter 5, we proposed a new distance to DFO trust region method for this distance. We studied local convergence issues for the adapted DFOb trust region. We showed that the adapted model is adapted to guarantee the local convergence of the method.

In Chapter 6, we introduced collections of benchmark functions and analyzed the numerical results obtained by the proposed method and compared them to the results of state-of-the-art DFO methods: NOMAD and RBFopt. We also built a toy problem that is closed to the real application. The first results obtained on the benchmark functions and on SAFRAN's application are very encouraging. Our adapted method outperforms the other methods in term of robustness and of simulation cost.

There remains many open possibilities for future work. First, we enlighted the sensitivity of the results to the initial points. As we mentioned before, we apply a LHS design for continuous and binary variables with rounding technique. We should build an initial set designed for binary variables, especially for cyclic symmetric problems. An interesting idea to pursue is to build a suitable initial design based on a positive definite kernel. [26, 27] have shown the theoretical proof that the Hamming distance is sufficient to build a positive definite kernel. We will study the possibility to use the necklace distance which we proposed to build a positive definite kernel.

Another important issue to improve is the globalization of our method with respect to continuous variables (the globalization in sense of improving the local minima). As standard trust region methods, our method is a local optimization method. A classical approach to overcome the difficulty of local minima is the multi-start technique [9], [48]. The simulation cost and the presence of binary variables may invalid this type of method.

Another promising proposal from professor Marcel Mongeau mentions about using branch technique to work directly in necklace domain. From now we work in binary space (dimension 2^n, n number of binary variables). Thanks to necklace distance, we do not explore redundant

rotated solutions. The idea is to create a tree of all the distinct arrangements and classify the order of the leaves according to the number of 0 (1) in the arrangements. Then in the exploration phase, we can use search techniques to find a solution in Step (1.5a) on the tree.

Future work and planning

For the main schedule in my thesis, I would split into four main phases. The time schedule is arranging according to the last day in the contract with IFPEN **26/9/2021**

1. **Phase 1: Boosting Thesis (March 2020 to February 2021)** This part includes working days, meetings, attending conferences and writing papers

   - First, for the upcoming period, we would try our best to improve initial interpolation set using adapted design of experiment techniques. As explain in previous section, we would discover continuous part which attempts to globalize the objective function. We plan to submit my first paper to JoGO (Special issues in ICCSAMA conference). Presentation of my first results in Safran and also discuss with Safran experts. Presentation of my work at conference: DFO symposium (DFOS) at the University of British Columbia (Canada), from Aug 10 to 14, 2020. EUROPT continuous optimization working group of EURO at ENAC, Toulouse from 1 to 3 July, MASCOT-NUM annual conference from 4th to 7th May 2020 at Aussois.

   - We plan to collaborate with Sébastien Le Digabel (NOMAD contributor from GERAD, Canada) and Giacomo Nannicini (RBFopt contributor from IBM): some interesting ideas can be shared for DFO methods NOMAD, RBFopt and our DFO trust region method, especially, regarding globalization.

2. **Phase 2: Writing Thesis (February 2021 to June 2021)** This part is for writing final report respect to the deadline of submitting manuscript (before the defense at least 3 months)

   - Focusing on writing the final report. Respect to requirements about limitation of pages, report form, ensure the quality of the report
   - During writing part, we expect to work upon a paper also (summarize of work since midterm)

3. **Phase 3: Writing Presentation (June 2021 to 25^{th} September 2021)** : This part is for preparing

   - Defense presentation.
   - Rehearsal
   - In parallel, if possible, we can think about papers

4. **Phase 4: Defense day (26/09/2021)**: Respect to the contract of working with IFPEN, the last day of the contract is 26/09/2021. It has no doubt that we should defense on that day.

Professional project

During my internship at INRIA (Grenoble), I have realized that I am specifically interested in application fields. Then the opportunity of working at IFPEN which collaborates with Safran and ENAC en-boosted my orientation about my future work. It would be an excellent experience if I could work for companies as Safran Tech, Airbus, EDF or laboratories which combine academic and industry such as INRIA, ONERA, CEA.

Appendix A

Toy problem

Single-degree of freedom model

In the following part, we illustrate a very simple model closed to Safran's application: the problem of determining a mistuning pattern of a cyclic bladed disk that maximize the vibration amplification. Following [10–12, 32, 37, 40, 49], we use a single-degree-of-freedom (DOF) per blade disk model. To keep the notation clear throughout the section, we introduce the table of nomenclature in Table (A.1):

Table A.1: Nomenclature using for single DOF model.

N	: Number of blades	F_0	: Magnitude of exiting fore (N)
m_i	: Blade mass (kg)	E	: Order of exitation
k_c	: Coupling stiffness blade-to-blade (N/m)	A	: Tuned blades
k_b	: Stiffness of reference blade (N/m)	B	: Mistuned blades has 10% higher in k_b
c	: Damping value $(N.s/m)$	δ	: Deviation of volume (%)

In a theoretically dynamic analysis of a turbomachinery rotor, we usually assumed that the blades are identical. But, in practice, it appears differences in their mechanical properties or their geometry due to the manufacturing process. This mistuning may affect significantly the forced response and the cyclic symmetry of bladed disks. Thus, we attempt to intentionally mistune the design of the blades to analyse the impact on the cyclic property.

The consideration of intentional mistuning is not new, it is introduced and investigated in [10, 11]. Some papers [12, 49] described a general optimization approach using genertic algorithms. In this section, we focus on building the optimization of an intentional mistuning bladed disk model.

We investigate the case with two sets of blades, called A and B. This symplification is due to the complexity of modeling the manufacturing and certification process. It is highly reasonable to keep the number of different types of blades as small as possible.

Two blades A and B are different from their geometry in reality: the thickness, the length of the blades. These elements impact into frequency of blades. We consider a simple model whose continuous variable be directly frequency.

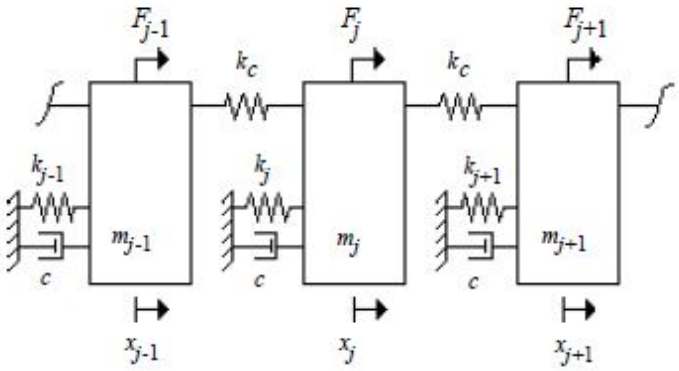

Figure A.1: Single DOF per bladed disk model

Figure (A.1) depicts the simplest model of a bladed disk: single DOF-sector model. The motion equation of the system composed of n blades is, for blade i,

$$m_i\ddot{x}_i + c\dot{x}_i + -k_c x_{i-1} + k_i x_i + 2k_c x_i - k_c x_{i+1} = F_i,\ i = 1, \ldots, N, \tag{A.1}$$

where the blade stiffness $k_i = (1+\delta_i)k_b$, the force induced by a harmonic engine order excitation E, $F_i = F_0 e^{j(\omega t+\phi_i)}$, the phase of the excitation of blade i $\phi_i = 2\pi(i-1)E/N$. δ_i is a mistuning parameter of stiffness for blade i. Let us assume the harmonic motion, i.e., $x_i = A_i e^{j\omega t}, j = \sqrt{-1}$, A_i the amplitude of displacement of blade i. We thus obtain

$$-m_i\omega^2 x_i + j\omega c x_i + -k_c x_{i-1} + (1+\delta_i)k_b x_i + 2k_c x_i - k_c x_{i+1} = F_i,\ i = 1, \ldots, N. \tag{A.2}$$

For the tuned, un-forced ($F_i = 0$), un-damped ($c = 0$) case the equation of motion is

$$m_i\ddot{x}_i - k_c x_{i-1} + k_b x_i + 2k_c x_i - k_c x_{i+1} = 0,\ i = 1, \ldots, N.$$

According to ([40]) the natural frequencies of the tuned system are given by the formula

$$\hat{\omega}_k^2 = 1 + 2R(1 - \cos\sigma_k),\ k = 1, \ldots, N, \tag{A.3}$$

where $\hat{\omega}^2 = \dfrac{m_k\omega^2}{k_b}, R = \dfrac{k_c}{k_b}, \sigma_k = \dfrac{2\pi(k-1)}{N}, \cos\sigma_k = \cos\sigma_{N+2-k}$.

We reformulate (A.2) as the form of the transform matrix. One notes that $x_{N+1} = x_1$, $x_0 = x_N$ from the cyclic property.

$$[-M\omega^2 + j\omega D + K]X = \mathcal{F}, \tag{A.4}$$

where $M = \mathrm{diag}(m_0, \ldots, m_{N-1}), D = \mathrm{diag}(c, c, \ldots, c), K = \mathrm{diag}(2k_c+(1+\delta_i)k_b)+[-k_c]_{(1,N),(N,1)}$ are respectively the mass, the damping and the stiffness matrix, $X = (x_0, \ldots, x_{N-1})^T, \mathcal{F} = (F_0, \ldots, F_{N-1})^T$ are the vector of displacements and forces.

$$TX = \begin{pmatrix} T_0 & -k_c & 0 & \ldots & 0 & -k_c \\ -k_c & T_1 & -k_c & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -k_c & 0 & 0 & \ldots & -k_c & T_{N-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{pmatrix} = \begin{pmatrix} F_0 \\ F_1 \\ F_2 \\ \vdots \\ F_{N-1} \end{pmatrix}, \tag{A.5}$$

where $T_i = -m_i\omega^2 + j\omega c + 2k_c + (1+\delta_i)k_b, \delta_i = y_i\delta, y_i = 0$ if blade A, otherwise $y_i = 1, \delta > 0$ isthe mistuning constant . Thus, (A.5) can be rewritten as

$$TA = \bar{\mathcal{F}}, \tag{A.6}$$

where $A = (A_0, A_1, \ldots, A_{N-1})^T$, $\bar{\mathcal{F}} = (\bar{F}_i)_{i=0,\ldots,N-1}, \bar{F}_i = F_0 e^{j\phi_i}$. The above equation gives us the form of the vector of amplitude magnification for forced response $A = T^{-1}\bar{\mathcal{F}}$.

Using the comments in [10,11], we build our optimization problem in order to obtain the pattern(s) that yield(s) the smallest value of the largest amplitude of response to a given excitation by varying the number of the position of the mistuned blades

$$\begin{aligned} \underset{\omega,y}{\text{minimize}} \quad & \|A\|_\infty \\ \text{subject to} \quad & \omega \in [\omega_{min}, \omega_{max}], \\ & y \in \{0,1\}^N. \end{aligned} \tag{A.7}$$

Appendix B

Fundamental theorem of calculus

Lemma B.1. *(lemma 4.1.2, [17]) Let $f : \mathbb{R}^n \to \mathbb{R} \in \mathcal{C}^1$. Then, for $x, d \in \mathbb{R}^n$, the direction derivative of x in the direction of d exists and equals $\nabla f(x)^T d$. Furthernore*

$$f(x+d) - f(x) = \int_0^1 d^T \nabla f(x+td) dt.$$

Proof. Applying the fundamental theorem of calculus of one variable to $g(t) = f(x+td)$, we have

$$g(1) = g(0) + \int_0^1 g'(t) dt.$$

Define $x(t) = x + td$, then by the chain rule, for $0 \leq \alpha \leq 1$

$$\begin{aligned} \frac{dg}{dt}(\alpha) &= \sum_{i=1}^{n} \frac{\partial f(x(t))}{\partial x(t)_i}(x(\alpha)) \frac{dx(t)_i}{dt}(\alpha) \\ &= \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x(\alpha)) p_i \\ &= \nabla f(x + \alpha d)^T d. \end{aligned} \tag{B.1}$$

By definition of g and equation (B.1) we obtain the lemma. □

Lemma B.2. *(lemma 4.1.12, [17]) Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be continuously differentiable, i.e., $f \in \mathcal{C}^1$ on $D = B(x, \Delta), x \in D$, and let ∇f be Lipchitz continuous at x in the neighborhood of D with the Lipchitz constant ν, then for any $x + d \in D$,*

$$|f(x+d) - f(x) - d^T \nabla f(x)| \leq \frac{1}{2} \nu \|d\|^2. \tag{B.2}$$

Proof. By using lemma (B.1) we have

$$\begin{aligned}
|f(x+d) - f(x) - d^T \nabla f(x)| &= |\int_0^1 d^T \nabla f(x+td)dt - d^T \nabla f(x)| \\
&= |\int_0^1 d^T (\nabla f(x+td) - \nabla f(x))dt| \\
&\leq \int_0^1 |d^T (\nabla f(x+td) - \nabla f(x))dt| \\
&\leq \int_0^1 \|d^T\| \|(\nabla f(x+td) - \nabla f(x))\| dt \\
&\leq \int_0^1 \|d^T\| \nu \|x+td-x\| dt \\
&= \nu \|d\|^2 \int_0^1 t dt = \frac{1}{2}\nu \|d\|^2.
\end{aligned} \tag{B.3}$$

□

Appendix C

Benchmark functions

C.1 Luksan and Vlcek (2000) benchmark

Problem C.1. *CB2.*

$$\begin{aligned} F(x) &= \max_{1\leq i\leq 3} f_i(x), \\ f_1(x) &= x_1^2 + x_2^4, \\ f_2(x) &= (2-x_1)^2 + (2-x_2)^2, \\ f_3(x) &= 2e^{x_2-x_1}. \end{aligned}$$

Problem C.2. *CB3.*

$$\begin{aligned} F(x) &= \max_{1\leq i\leq 3} f_i(x), \\ f_1(x) &= x_1^4 + x_2^2, \\ f_2(x) &= (2-x_1)^2 + (2-x_2)^2, \\ f_3(x) &= 2e^{x_2-x_1}. \end{aligned}$$

Problem C.3. *QL.*

$$\begin{aligned} F(x) &= \max_{1\leq i\leq 3} f_i(x), \\ f_1(x) &= x_1^2 + x_2^2, \\ f_2(x) &= x_1^2 + x_2^2 + 10(-4x_1 - x_2 + 4), \\ f_3(x) &= x_1^2 + x_2^2 + 10(-x_1 - 2x_2 + 6). \end{aligned}$$

Problem C.4. *WF.*

$$\begin{aligned} F(x) &= \max_{1\leq i\leq 3} f_i(x), \\ f_1(x) &= \frac{1}{2}(x_1 + \frac{10x_1}{x_1+0.1} + 2x_2^2), \\ f_2(x) &= \frac{1}{2}(-x_1 + \frac{10x_1}{x_1+0.1} + 2x_2^2), \\ f_3(x) &= \frac{1}{2}(x_1 - \frac{10x_1}{x_1+0.1} + 2x_2^2). \end{aligned}$$

Problem C.5. *PENTAGON.*

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 3} f_i(x),\\
f_1(x) &= -\sqrt{(x_1-x_3)^2+(x_2-x_4)^2},\\
f_2(x) &= -\sqrt{(x_3-x_5)^2+(x_2-x_6)^2},\\
f_3(x) &= -\sqrt{(x_5-x_1)^2+(x_6-x_2)^2}.
\end{aligned}
$$

Problem C.6. *ROSEN-SUZUKI.*

$$
\begin{aligned}
F(x) =& \max\{f_1(x), f_1(x)+10f_2(x), f_1(x)+10f_3(x), f_1(x)+10f_4(x)\},\\
f_1(x) =& x_1^2+x_2^2+2x_3^2+x_4^2-5x_1-5x_2-21x_3+7x_4,\\
f_2(x) =& x_1^2+x_2^2+x_3^2+x_4^2+x_1-x_2+x_3-x_4-8,\\
f_3(x) =& x_1^2+2x_2^2+x_3^2+2x_4^2-x_1-x_4-10,\\
f_4(x) =& x_1^2+x_2^2+x_3^2+2x_1-x_4-5.
\end{aligned}
$$

Problem C.7. *WONG 2.*

$$
\begin{aligned}
F(x) =& \max_{1\le i\le 6} f_i(x),\\
f_1(x) =& x_1^2+x_2^2+x_1x_2-14x_1-16x_2+(x_3-10)^2+4(x_4-5)^2+(x_5-3)^2+\\
& 2(x_6-1)^2+5x_7^2+7(x_8-11)^2+2(x_9-10)^2+(x_{10}-7)^2+45,\\
f_2(x) =& f_1(x)+10(3(x_1-2)^2+4(x_2-3)^2+2x_3^2-7x_4-120),\\
f_3(x) =& f_1(x)+10(5x_1^2+8x_2+(x_3-6)^2-2x_4-40),\\
f_4(x) =& f_1(x)+10(0.5(x_1-8)^2+2(x_2-4)^2+3x_5^2-x_6-30),\\
f_5(x) =& f_1(x)+10(x_1^2+2(x_2-2)^2-2x_1x_2+14x_5-6x_6),\\
f_6(x) =& f_1(x)+10(-3x_1+6x_2+12(x_9-8)^2-7x_{10}).
\end{aligned}
$$

Problem C.8. *WONG 3.*

$$
\begin{aligned}
F(x) =& \max_{1\le i\le 14} f_i(x),\\
f_1(x) =& x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3-10)^2 + 4(x_4-5)^2 + (x_5-3)^2 +\\
& 2(x_6-1)^2 + 5x_7^2 + 7(x_8-11)^2 + 2(x_9-10)^2 + (x_{10}-7)^2 + (x_{11}-9)^2 +\\
& 10(x_{12}-1)^2 + 5(x_{13}-7)^2 + 4(x_{14}-14)^2 + 27(x_{15}-1)^2 + x_{16}^4 + (x_{17}-2)^2 +\\
& 13(x_{18}-2)^2 + (x_{19}-3)^2 + x_{20}^2 + 95,\\
f_2(x) =& f_1(x) + 10(3(x_1-2)^2 + 4(x_2-3)^2 + 2x_3^2 - 7x_4 - 120),\\
f_3(x) =& f_1(x) + 10(5x_1^2 + 8x_2 + (x_3-6)^2 - 2x_4 - 40),\\
f_4(x) =& f_1(x) + 10(0.5(x_1-8)^2 + 2(x_2-4)^2 + 3x_5^2 - x_6 - 30),\\
f_5(x) =& f_1(x) + 10(x_1^2 + 2(x_2-2)^2 - 2x_1x_2 + 14x_5 - 6x_6),\\
f_6(x) =& f_1(x) + 10(-3x_1 + 6x_2 + 12(x_9-8)^2 - 7x_{10}),\\
f_7(x) =& f_1(x) + 10(x_1^2 + 5x_{11} - 8x_{12} - 28),\\
f_8(x) =& f_1(x) + 10(4x_1 + 9x_2 + 5x_{13}^2 - 9x_{14} - 87),\\
f_9(x) =& f_1(x) + 10(3x_1 + 4x_2 + 3(x_{13}-6)^2 - 14x_{14} - 10),\\
f_{10}(x) =& f_1(x) + 10(14x_1^2 + 35x_{15} - 79x_{16} - 92),\\
f_{11}(x) =& f_1(x) + 10(15x_2^2 + 11x_{15} - 61x_{16} - 54),\\
f_{12}(x) =& f_1(x) + 10(5x_1^2 + 2x_2 + 9x_{17}^4 - x_{18} - 68),\\
f_{13}(x) =& f_1(x) + 10(x_1^2 - x_9 + 19x_{19} - 20x_{20} + 19),\\
f_{14}(x) =& f_1(x) + 10(7x_1^2 + 5x_2^2 + x_{19}^2 - 30x_{20}).
\end{aligned}
$$

Problem C.9. *MAD1.*

$$
\begin{aligned}
F(x) =& \max_{1\le i\le 3} f_i(x),\\
f_1(x) =& x_1^2 + x_2^2 + x_1x_2 - 1,\\
f_2(x) =& \sin(x_1),\\
f_3(x) =& -\cos(x_2).
\end{aligned}
$$

Problem C.10. *MAD4.*

$$
\begin{aligned}
F(x) =& \max_{1\le i\le 3} f_i(x),\\
f_1(x) =& -e^{x_1-x_2},\\
f_2(x) =& \sinh(x_1-1) - 1,\\
f_3(x) =& -\log(x_2) - 1.
\end{aligned}
$$

C.2 Hock and Schittkowski benchmark

Problem C.11. *HS2.*

$$F(x) = 100(x_2 - x_1^2)^2 + (1-x_1)^2.$$

Problem C.12. *HS3.*

$$F(x) = x_2 + 10^{-5}(x_2 - x_1)^2.$$

Problem C.13. *HS29log.*

$$F(x) = \log_{10}(100(x_2 - x_1^2)^2 + (1 - x_1)^2).$$

C.3 Dixon–Szegö benchmark

Problem C.14. *Branin.*

$$F(x) = (x_2 - (\frac{5.1}{4\pi^2})x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10.$$

Problem C.15. *Camel.*

$$F(x) = (4 - 2.1x_1^2 + x_1^{4/3})x_1^2 + x_1x_2 + (-4 + 4x_1^2)x_1^2.$$

Problem C.16. *Goldstein-Price.*

$$F(x) = (1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2))*$$
$$(30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)).$$

Problem C.17. *Hartman3.*

```
a = [ [3.0,  0.1,  3.0,  0.1],
      [10.0, 10.0, 10.0, 10.0],
      [30.0, 35.0, 30.0, 35.0] ]
P = [ [0.36890, 0.46990, 0.10910, 0.03815],
      [0.11700, 0.43870, 0.87320, 0.57430],
      [0.26730, 0.74700, 0.55470, 0.88280]]
c = [1.0, 1.2, 3.0, 3.2]
```

$$F(x) = -\sum_{i=1}^{4} c_i e^{-\sum_{j=1}^{3} a_{ji}(x_j - P_{ji})^2}.$$

Problem C.18. *Hartman6.*

```
a = [ [10.00,  0.05,  3.00, 17.00],
      [3.00, 10.00,  3.50,  8.00],
      [17.00, 17.00,  1.70,  0.05],
      [3.50,  0.10, 10.00, 10.00],
      [1.70,  8.00, 17.00,  0.10],
      [8.00, 14.00,  8.00, 14.00] ]
p = [ [0.1312, 0.2329, 0.2348, 0.4047],
      [0.1696, 0.4135, 0.1451, 0.8828],
      [0.5569, 0.8307, 0.3522, 0.8732],
      [0.0124, 0.3736, 0.2883, 0.5743],
      [0.8283, 0.1004, 0.3047, 0.1091],
      [0.5886, 0.9991, 0.6650, 0.0381] ]
c = [1.0, 1.2, 3.0, 3.2]
```

$$F(x) = -\sum_{i=1}^{4} c_i e^{-\sum_{j=1}^{6} a_{ji}(x_j - P_{ji})^2}.$$

Problem C.19. *Shekel7.*

$$
\begin{aligned}
a &= [\ [4.0,\ 1.0,\ 8.0,\ 6.0,\ 3.0,\ 2.0,\ 5.0],\\
&\quad\ [4.0,\ 1.0,\ 8.0,\ 6.0,\ 7.0,\ 9.0,\ 5.0],\\
&\quad\ [4.0,\ 1.0,\ 8.0,\ 6.0,\ 3.0,\ 2.0,\ 3.0],\\
&\quad\ [4.0,\ 1.0,\ 8.0,\ 6.0,\ 7.0,\ 9.0,\ 3.0]\ ]\\
c &= [0.1,\ 0.2,\ 0.2,\ 0.4,\ 0.4,\ 0.6,\ 0.3]
\end{aligned}
$$

$$F(x) = -\sum_{j=1}^{7} \frac{1}{\sum\limits_{i=1}^{4}(x_i - a_{ij})^2 + c_j}.$$

Problem C.20. *Shekel10.*

$$
\begin{aligned}
a &= [\ [4.0,\ 1.0,\ 8.0,\ 6.0,\ 3.0,\ 2.0,\ 5.0,\ 8.0,\ 6.0,\ 7.0],\\
&\quad\ [4.0,\ 1.0,\ 8.0,\ 6.0,\ 7.0,\ 9.0,\ 5.0,\ 1.0,\ 2.0,\ 3.6],\\
&\quad\ [4.0,\ 1.0,\ 8.0,\ 6.0,\ 3.0,\ 2.0,\ 3.0,\ 8.0,\ 6.0,\ 7.0],\\
&\quad\ [4.0,\ 1.0,\ 8.0,\ 6.0,\ 7.0,\ 9.0,\ 3.0,\ 1.0,\ 2.0,\ 3.6]\ ]\\
c &= [0.1,\ 0.2,\ 0.2,\ 0.4,\ 0.4,\ 0.6,\ 0.3,\ 0.7,\ 0.5,\ 0.5]
\end{aligned}
$$

$$F(x) = -\sum_{j=1}^{1} 0\frac{1}{\sum\limits_{i=1}^{4}(x_i - a_{ij})^2 + c_j}.$$

C.4 GLOBALLIB benchmark

Problem C.21. `ex8_1_1`

.

$$F(x) = \cos(x_1)\sin(x_2) - \frac{x_1}{x_2^2 + 1}.$$

Problem C.22. `ex8_1_4`

.

$$F(x) = 12x_1^2 - 6.3x_1^4 + x_1^6 - 6x_1x_2 + 6x_2^2.$$

Problem C.23. *Perm6.*

$$\beta = 60,$$

$$F(x) = \sum_{k=1}^{6}(\sum_{i=1}^{6}((i+1)^k + \beta)(\frac{x_i}{(i+1)^k} - 1))^2 + 1000.$$

Problem C.24. *Perm8.*

$$\beta = 100,$$

$$F(x) = \sum_{k=1}^{8}(\sum_{i=1}^{8}((i+1) + \beta)(x_i^k - (\frac{1}{(i+1)})^k)^2 + 1000.$$

C.5 MINLPLib2 benchmark

Problem C.25. *Sporttournament.*

$$\begin{aligned} F(x) = {} & 2x_1x_3 - 2x_1 + 2x_3 + 2x_1x_7 - 2x_7 + 2x_2x_6 - 2x_2 - 2x_5 + 2x_2x_{10} - \\ & 4x_{10} - 2x_3x_4 + 2x_4 - 2x_3x_{12} - 2x_3x_{14} - 2x_4x_5 + 2x_4x_9 - 2x_9 - \\ & 2x_4x_{15} + 2x_5x_6 - 2x_6 + 2x_5x_8 - 2x_8 + 2x_6x_9 - 2x_7x_8 + 2x_7x_{12} + \\ & 2x_7x_{13} + 2x_8x_{10} + 2x_8x_{15} + 2x_9x_{11} - 2x_{11} - 2x_9x_{13} + 2x_{10}x_{11} + \\ & 2x_{10}x_{12} - 2x_{13}x_{15} + 2x_{14}x_{15}. \end{aligned}$$

Bibliography

[1] *Minlplib2*, http://www.gamsworld.org/minlp/minlplib2/html/.

[2] *Air transport action group (atag).* 2016, http://www.atag.org/facts-and-figures.html.

[3] *International air transport association, iata price analysis.* 2016, http://www.iata.org/publications/economics/fuel-monitor/Pages/price-analysis.aspx.

[4] M. Abramson, *Pattern search algorithms for mixed variable general constrained optimization problems*, PhD thesis, (2003).

[5] M. Abramson, C. Audet, G. Couture, J. Dennis, Jr., S. Le Digabel, and C. Tribes, *The NOMAD project.* Software available at https://www.gerad.ca/nomad/, https://www.gerad.ca/nomad/.

[6] C. Audet and J. Dennis, *Pattern search algorithms for mixed variable programming*, SIAM Journal on Optimization, 11 (2000), https://doi.org/10.1137/S1052623499352024.

[7] C. Audet and W. Hare, *Derivative-Free and Blackbox Optimization*, Springer Series in Operations Research and Financial Engineering, Springer International Publishing, Cham, Switzerland, 2017, https://doi.org/10.1007/978-3-319-68913-5.

[8] D. Bremner, T. M. Chan, E. D. Demaine, J. Erickson, F. Hurtado, J. Iacono, S. Langerman, M. Patrascu, and P. Taslakian, *Necklaces, convolutions, and X+Y*, CoRR, abs/1212.4771 (2012), http://arxiv.org/abs/1212.4771, https://arxiv.org/abs/1212.4771.

[9] C. Cartis, L. Roberts, and O. Sheridan-Methven, *Escaping local minima with derivative-free methods: a numerical investigation*, 2018, https://arxiv.org/abs/1812.11343.

[10] B. Choi, *Pattern optimization of intentional blade mistuning for the reduction of the forced response using genetic algorithm*, KSME International Journal, 17 (2003), pp. 966–977, https://doi.org/10.1007/BF02982981, https://doi.org/10.1007/BF02982981.

[11] B. Choi, K. H. Eun, K. H. Jung, J. Haneol, G. DongSik, and K. M. Kwan, *Optimization of intentional mistuning for bladed disk : Intentional mistuning intensity effect*, in Engineering Asset Management, J. Mathew, J. Kennedy, L. Ma, A. Tan, and D. Anderson, eds., London, 2006, Springer London, pp. 1024–1029.

[12] B. Choi, J. Lentz, A. Rivas-Guerra, and M. Mignolet, *Optimization of intentional mistuning patterns for the reduction of the forced response effects of unintentional mistuning: Formulation and assessment*, Journal of Engineering for Gas Turbines and Power, 125 (2003), pp. 131–140, https://doi.org/10.1115/1.1498270.

[13] A. L. Claudia D'Ambrosio, *Mixed integer nonlinear programming tools: an updated practical overview*, Springer Science+Business Media New York 2013, (2013), https://doi.org/10.1007/s10479-012-1272-5.

[14] R. Conn, C. D'Ambrosio, L. Liberti, and D. Sinoquet, *A trust region method for solving grey-box mixed integer nonlinear problems with industrial applications.* https://mode2016.sciencesconf.org/file/223761.

[15] R. Conn, K. Scheinberg, and L. Vicente, *Introduction to Derivative-Free Optimization*, Society for Industrial and Applied Mathematics, 2009, https://doi.org/10.1137/1.9780898718768, https://epubs.siam.org/doi/abs/10.1137/1.9780898718768, https://arxiv.org/abs/https://epubs.siam.org/doi/pdf/10.1137/1.9780898718768.

[16] A. Costa and G. Nannicini, *Rbfopt: an open-source library for black-box optimization with costly function evaluations*, Mathematical Programming Computation, 10 (2018), pp. 597–629, https://doi.org/10.1007/s12532-018-0144-7, https://doi.org/10.1007/s12532-018-0144-7.

[17] S. R. Dennis, J. E., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Classics in Applied Mathematics, Society for Industrial and Applied Mathematics, 1996, https://doi.org/10.1137/1.9781611971200, https://doi.org/10.1137/1.9781611971200, https://arxiv.org/abs/https://doi.org/10.1137/1.9781611971200.

[18] S. G. Dixon, L., *The global optimization problem: an introduction*, In: Dixon, L., Szego, G. (eds.) Towards Global Optimization, North Holland, Amsterdam (1975), pp. 1–15.

[19] E. et al., *The distance geometry of music*, (2007).

[20] M. M. et al., *Necklace swap problem for rhythmic similarity measure*, (2005).

[21] B. F.Meunier, *Computing solution of the paintshop necklace problem*, (2011).

[22] H. Fredricksen and I. J. Kessler, *An algorithm for generating necklaces of beads in two colors*, Discrete Mathematics, 61 (1986), pp. 181 – 188, https://doi.org/https://doi.org/10.1016/0012-365X(86)90089-0, http://www.sciencedirect.com/science/article/pii/0012365X86900890.

[23] D. Gabric and J. Sawada, *Constructing de bruijn sequences by concatenating smaller universal cycles*, Theoretical Computer Science, 743 (2018), pp. 12 – 22, https://doi.org/https://doi.org/10.1016/j.tcs.2018.06.039, http://www.sciencedirect.com/science/article/pii/S0304397518304559.

[24] H.-M. Gutmann, *A radial basis function method for global optimization*, Journal of Global Optimization, 19 (2001), pp. 201–227, https://doi.org/10.1023/A:1011255519438, https://doi.org/10.1023/A:1011255519438.

[25] S. K. Hock W, *Test examples for nonlinear programming codes*, Lecture Notes in Economics and Mathematical Systems, Berlin: Springer, 87 (1981).

[26] F. Hutter, L. Xu, H. H. Hoos, and K. Leyton-Brown, *Online appendix for aij article "algorithm runtime prediction: Methods & evaluation"*, Elsevier, (2013), :http://www.cs.ubc.ca/labs/beta/Projects/EPMs/EPMs-online-appendix-opt.pdf.

[27] F. Hutter, L. Xu, H. H. Hoos, and K. Leyton-Brown, *Algorithm runtime prediction: Methods & evaluation*, Artificial Intelligence, 206 (2014), pp. 79 – 111, https://doi.org/https://doi.org/10.1016/j.artint.2013.10.003, http://www.sciencedirect.com/science/article/pii/S0004370213001082.

[28] M. N. Izzat Alsmadi, *String matching evaluation methods for dna comparation*, International Journal of advanced Science and Technology, 47 (2012).

[29] M. Jiang, *On the sum of distances along a circle*, Discrete Mathematics, 308 (2008), pp. 2038 – 2045, https://doi.org/https://doi.org/10.1016/j.disc.2007.04.025, http://www.sciencedirect.com/science/article/pii/S0012365X07002555.

[30] M. Jiang., *A linear-time algorithm for hamming distance with shifts*, Theory Comput Syst, 44 (2009), p. 349–355, https://doi.org/10.1007/s00224-007-9088-4.

[31] S. Le Digabel, *Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm*, ACM Transactions on Mathematical Software, 37 (2011), pp. 1–15.

[32] H. Liao, J. Wang, J. Yao, and Q. Li, *Mistuning Forced Response Characteristics Analysis of Mistuned Bladed Disks*, Journal of Engineering for Gas Turbines and Power, 132 (2010), https://doi.org/10.1115/1.4001054, https://doi.org/10.1115/1.4001054, https://arxiv.org/abs/https://asmedigitalcollection.asme.org/gasturbinespower/article-pdf/132/12/122501/4881737/122501_1.pdf. 122501.

[33] L. S. . R. Liuzzi, G., *Derivative-free methods for bound constrained mixed-integer optimization*, Comput Optim Appl, 53 (2012), p. 505–526, https://doi.org/10.1007/s10589-011-9405-3.

[34] S. Lucidi and V. Piccialli, *An algorithm model for mixed variable programming*, SIAM Journal on Optimization, 15 (2005), https://doi.org/10.1137/S1052623403429573.

[35] L. Luksan and J. Vlček, *Test problems for nonsmooth unconstrained and linearly constrained optimization*, Institute of Computer Science, Academy of Sciences of the Czech Republic, Technical report VT798-00 (2000).

[36] W. J. C. M. D. Mckay, R. J. Beckman, *A comparison of three methods for selecting values of input variables in the analysis of output from a computer code"*, American Statistical Association and American Society for Quality, 42 (2010), pp. 55–61, :http://www.jstor.org/stable/1271432.

[37] M. Moustapha, *Conception robuste en vibration et aéroélasticité des roues aubagées de turbomachines*, PhD thesis, Université Paris-Est Marne la vallée, 2009, https://tel.archives-ouvertes.fr/tel-00529002v2/document.

[38] A. Neumaier, *Neumaier's collection of test problems for global optimization*, (Retrieved in May 2014), pp. 1–15, http://www.mat.univie.ac.at/~neum/glopt/my_problems.html.

[39] J. Nocedal and S. Wright, *Numerical optimization*, Springer, New York, NY, (2006), https://doi.org/10.1007/978-0-387-40065-5.

[40] G. Óttarsson, *Dynamic modeling and vibration analysis of mistuned bladed disks*, theses, University of Michigan, May 1994, https://tel.archives-ouvertes.fr/tel-00598068.

[41] M. Powell., *On trust region methods for unconstrained minimization without derivatives*, Math. Program., Ser, (2003), https://doi.org/10.1007/s10107-003-0430-6.

[42] C. A. S. Rao T. R. N., *Asymmetric error codes for some lsi semiconductor memories*, . 7-th Ann. Southeast Symp. Syst. Theory, Auburn-Tuskegee, (1975), pp. 170–171.

[43] A. N. L. Samuel Burer, *Non-convex mixed-integer nonlinear programming: A survey*, Elsevier, 53 (2012), pp. 97–106, https://doi.org/10.1016/j.sorms.2012.08.001.

[44] G. Toussaint, *A comparison of rhythmic similarity measures.*

[45] G. Toussaint, *A mathematical analysis of african, brazilian, and cuban clave rhythms*, in Townson University, 2002, pp. 157–168.

[46] G. Toussaint, *The geometry of musical rhythm*, in Discrete and Computational Geometry, J. Akiyama, M. Kano, and X. Tan, eds., Berlin, Heidelberg, 2005, Springer Berlin Heidelberg, pp. 198–212.

[47] G. Toussaint, *Computational geometric aspects of rhythm, melody, and voice-leading*, Computational Geometry, 43 (2010), pp. 2 – 22, https://doi.org/https://doi.org/10.1016/j.comgeo.2007.01.003, http://www.sciencedirect.com/science/article/pii/S092577210900042X. Special Issue on the 14th Annual Fall Workshop.

[48] S. M. Wild, *Derivative-Free Optimization Algorithms for Computationally Expensive Functions*, PhD thesis, USA, 2009.

[49] M. P. M. Y.Han, *Optimization of intentional mistuning patterns for the mitigation of the effects of randon mistuning*, SME Turbo Expo 2008, Berlin, Germany, (2008), https://doi.org/10.1007/978-0-387-40065-5.