Université Fédérale
Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue par

Thi Thoi TRAN

Le 25 octobre 2021

Optimisation non linéaire en variables mixtes continues et discrètes pour des simulateurs de type de boîte noire

---

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Mathématiques et Applications**

Unité de recherche :
ENAC-LAB - Laboratoire de Recherche ENAC

Thèse dirigée par
Marcel MONGEAU

Jury

M. Charles Audet, Rapporteur
M. Laurent Dumas, Rapporteur
Mme Delphine Sinoquet, Examinatrice
M. Sébastien Da Veiga, Examinateur
M. Jean-Baptiste HIRIART-URRUTY, Examinateur
Mme Ana Luísa Custódio, Examinatrice
M. Merlin Keller, Examinateur
M. Marcel MONGEAU, Directeur de thèse

Thèse en vue de l'obtention du doctorat délivré par

UNIVERSITÉ DE TOULOUSE 3 - PAUL SABATIER

École Doctorale Mathématiques, Informatuque
et Télécommunications de Toulouse

En cotutelle avec

IFP Energies Nouvelles
Departement des Mathématiques Appliques

Thèse intitulée

Optimisation non linéaire en variables mixtes continues et discrètes pour des simulateurs de type de boîte noire

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

présentée par

Thi Thoi TRAN

soutenue le 25 octobbre 2021

Thèse dirigée par

Marcel MONGEAU	ENAC
Delphine SINOQUET	IFP Energies Nouvelles
Sébastien DA VEIGA	Safran Tech

Jury

Charles AUDET	Polytechnique Montréal
Laurent DUMAS	Université de Versailles St-Quentin-en-Yvelines
Jean-Baptiste HIRIART-URRUTY	Université Toulouse III - Paul Sabatier
Ana Luísa CUSTÓDIO	Universidade NOVA de Lisboa
Merlin KELLER	EDF R&D

I would like to dedicate this thesis to my family: my parents, my siblings, my nephews and my nieces!

Acknowledgments

I would like to take this opportunity to express my sincere gratitude and appreciation to everyone who supported me during this thesis, without them, this thesis would not be completed.

First of all, I am grateful to my advisor Dr. Delphine Sinoquet for her constant guidance and support. Her passion for applied mathematics, especially optimization was an inspiration for me during my Ph.D. and fed my curiosity with state-of-the-art optimization techniques, supporting me technically and steering me in the right direction whenever she thought I needed it. I would also like to thank my director, Professor Marcel Mongeau, and my co-advisor, Dr. Sébastien Da Veiga, for their great mentorship in technique problems and research. Their enriching advice gave during the analysis of my numerical results part, or the state-of-the-art studying, or proofreading the manuscript is invaluable to my thesis.

I would like to thank the members of my thesis jury for having accepted to evaluate my work. In particular, Professor Charles Audet and Professor Laurent Dumas for their time of re-lecture my thesis in detail and their constructive comments. I also thank Professor Jean-Baptiste Hiriat-Urruty, Professor Ana Luísa Custódio, and Dr. Merlin Keller for the interest they showed by accepting to participate in this jury.

I spent most of the time in Rueil Malmaison, surrounded by my colleagues in the department of applied mathematics where I quickly felt intimated. This integration was facilitated by Benjelloun-Touimi Zakia, the head of the department, who was always benevolent, who knew how to combine work and life. I especially mention Professor Quang Huy Tran, who had always answered all the questions regarding my thesis, conferences, visa, etc. My Vietnamese friends at IFP Energies Nouvelles: Duc Thach Son Vu, Quang Duy Le, Tri Dat Ngo, Manh Quan Nguyen, Ngoc Tu Trinh, Quang Long Nguyen, Van Quan Hoang, for their experiences sharing, life caring...they are all like my brothers. I also mention other colleagues with their sincerely constant help during my Ph.D.: Miguel Munoz Zuniga, Frederic Delbos for their valuable discussions about some parts of the work. Sylvie Hoguet, department assistant, for her help to fulfill the order missions, my salary time sheets, etc. Dominique Sfez, my proprietary, and her husband Pierre Doguin, for their support. Sylvie Pegaz-Fiornet, Henry Eon, Francoise Willien, Sylvie Wolf, Guillaume Enchery... for their great integration during our lunchtime, break times... and sometimes tried to teach French to me!! And of course, I can not forget to mention the former and current Ph.D. students in my department: Bastien Hamlat, Zakaria Jorti, Karine Laurent, Julien Coulet, Alexis Cousin, Sabrina Bassetto, Dongmo Nguepi Guissel Lagnol, Joelle Ferzly, Jingang Qu, Mohamed El Maarouf Aboul Karim, with them I had lots of great after-work times together. I would like to thank DSI, Security, cleaning, restaurant staff at IFP Energies Nouvelles for their informative support, safety insurance, making a clean working environment, and providing a variety of nutritious lunches.

I had the chance to visit ENAC, where I found a friendly environment while working with others. I would like to thank Ph.D. students at ENAC: Mahfoud Herraz, Sana Ikli, Hasnae Kasmi for their enthusiastic discussions and update ENAC's information for me. In particular, I mention Professor Nicolas Couellan, and former Ph.D. student Gabriel Jarry for their ideas that contributed to the reformulation part of the thesis.

During these three years, there were a huge number of administrative procedures that I had to do. As an international student, I would not be able to finish all those tasks without the help of Virginie Lemire, HR at IFP Energies Nouvelles, Helene Weiss, Optim team's assistant, Agnes Requis, EDMITT staff.

I would also thank to Safran tech, the funding granted by them has allowed the development of this thesis.

I express my sincere gratitude to my boyfriend, Anshul Paigwar, a robotic researcher at INRIA Grenoble for his constant encouragement and his love. He is not only my boyfriend but also my teacher in life who has helped me to improve my communication skills, my fashion style, my general knowledge, and presenting skills. I would also thank my friends: Thi Thanh Yen Nguyen, Quoc Huy Vu,... for the time that we have shared.

Finally, I must express my very profound gratitude to my parents, my siblings, my nephews, my nieces for providing me with unfailing support. This accomplishment would not have been possible without them. Thank you.

Rueil Malmaison, 1st, August, 2021,

Tran Thi Thoi

Résumé

Ces dernières années, un nombre considérable d'applications industrielles réelles impliquant des variables mixtes et des simulateurs coûteux en temps ont été réalisées, par exemple, chez Safran Tech et IFPEN, des conceptions optimales de turbine de moteur d'hélicoptère, de lignes d'amarrage d'éoliennes offshore, de stators et de rotors de moteurs électriques,... Dans ces problèmes d'optimisation non linéaire, les dérivées de la fonction objectif (et, éventuellement des fonctions contraintes) ne sont pas disponibles et ne peuvent pas être directement approximées. Une autre difficulté est que ces problèmes impliquent des variables de nature hétérogène: un nombre variable de composants (variables entières), différents matériaux (variables catégorielles, généralement non ordonnées), la présence ou non de certains composants (variables binaires) et des variables continues décrivant les dimensions/caractéristiques des pièces de la structure.

Cette thèse a pour but de développer et d'adapter des méthodes d'optimisation sans dérivées (ou DFO pour *Derivative-Free Optimization*) applicables à différents types d'applications, dont la conception optimale des moteurs d'hélicoptère. Dans la première partie, nous nous concentrons sur le développement et l'adaptation d'une méthode DFO aux problèmes avec variables mixtes continues et discrètes présentant une symétrie cyclique, caractéristiques présentes dans le problème d'optimisation des pales d'une turbomachine de moteur. À cette fin, nous introduisons une distance basée sur les colliers (*necklace distance*) et adaptons une distance basée sur les colliers pour une méthode d'optimisation du type région de confiance. Avant d'appliquer notre méthode à un cas applicatif simplifié fourni par Safran, nous construisons un ensemble de fonctions tests issues de la littérature que nous adaptons pour obtainir un emsemble de problèmes mixtes à symétrie cyclique. Notre méthode est évaluée sur ces cas tests et comparée à des méthodes d'optimisation sans dérivées de l'état de l'art. Nous donnons également une preuve de convergence locale de notre méthode adaptée.

Dans la deuxième partie, nous nous consacrons à la planification d'expériences dans un espace mixte (variables continues et discrètes) en étendant à cet espace mixte les approches basées sur des méthodes à noyaux pour estimer des distributions de probabilité. Cette partie de la thèse est motivée par le besoin d'améliorer la phase d'initialisation de l'algorithme d'optimisation. Le but étant de permettre une meilleure exploration de l'espace des variables mixtes, guidée par les informations a priori disponibles (types de variables, symétrie, corrélations,...). Nous illustrons également le potentiel de l'approche proposée dans le cadre plus classique de l'approximation d'une fonction par un méta-modèle pour des variables mixtes continues et discrètes mais aussi pour des séries temporelles.

Enfin, nous donnons des pistes d'amélioration de la méthode d'optimisation proposée pour une meilleure exploration de l'espace des variables de conception pour éviter d'être piégé dans des minima locaux.

Mots-clés: Optimisation mixte en nombres entiers, optimisation sans dérivée, simulation boîte noire, symétrie cyclique distance de collier, plan d'expériences, région de confiance.

Abstract

In recent years, there has been a considerable number of industrial applications that involve mixed variables and time-consuming simulators, *e.g.*, at Safran Tech and IFPEN: optimal designs of aircraft engine turbine, 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 constraint functions) are not available and cannot be directly approximated. Another difficulty is that these problems involve heterogeneous nature 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.

This thesis aims to develop and adapt Derivative-Free Optimization (DFO) methods for different types of applications, including the optimal design of aircraft engines. In the first part, we focus on the development and adaptation of a DFO method to problems with continuous and mixed discrete variables exhibiting a cyclic-symmetry property. For that purpose, we introduce the *necklace distance* and tailor accordingly the trust-region constraints of the optimization problems. Before running our adapted method on a simplified simulation provided by Safran, we build a set of benchmark functions by transforming them into a set of cyclic-symmetry test functions. We run our method on these benchmark functions and on a Safran's simulated instance with a large number of repetitions to study the robustness of the method compared to other state-of-the-art methods. We also give a local convergence proof of our adapted method.

In the second part, we focus on the design of experiments in mixed continuous and discrete variables space by extending the kernel-embedding distribution from continuous space to mixed discrete variables case. This part of the thesis is motivated by the need to improve the initialization phase of the optimization algorithm with a better exploration of the space of mixed variables, guided by the available prior information (types of variables, symmetry, correlations, ...). We illustrate the potential of the proposed approach in the more classical framework of meta-model function approximation for continuous and discrete mixed variables, and also for time series.

Finally, we give ideas to improve the proposed optimization method for a better exploration of the space of design variables to avoid being trapped in local minima.

Keywords: Mixed integer non-linear programming, derivative free optimization, black-box simulation, cyclic-symmetry necklace distance, design of experiments, trust region.

Contents

1 Introduction **1**
1 The motivating application . . . 2
1.1 Motivation . . . 2
1.2 Turbomachines . . . 2
1.3 Optimization challenges . . . 3
2 Reduced Order Model (ROM) . . . 5
3 Thesis contributions . . . 6
4 Outline of the thesis . . . 7
5 Publications . . . 8

I Derivative-free mixed binary necklace optimization **9**

2 Introduction to blackbox optimization methods for mixed integer variables **11**

3 Derivative-free trust-region methods for continuous variables problems **17**
1 Trust-region framework basics . . . 17
2 Derivative-free trust-region methods . . . 19
2.1 Geometry of the interpolation set . . . 20
2.2 Fully-linear and fully-quadratic models . . . 22
2.3 Building the trust-region model . . . 24
2.4 Model improvement . . . 26
3 Conclusion . . . 28

4 DFOb-d_H: A derivative-free trust-region method for mixed binary problems **29**
1 The quadratic model . . . 30
2 Initial interpolation set . . . 37
3 Exploitation phase . . . 38
4 Exploration phase . . . 39
5 Local convergence results . . . 43
6 Conclusion . . . 49

5 DFOb-d_{neck}: A DFOb method for cyclic-symmetry problems **51**

1 Necklace context . . . 52
1.1 Concept of “necklace" . . . 52
1.2 Survey of necklace distances . . . 53
2 An adapted distance for cyclic-symmetry problems . . . 56
2.1 The necklace distance . . . 56
2.2 Reformulation of the QP subproblems involving the necklace distance . . . 60
2.3 Algorithm DFOb-d_{neck} . . . 65
3 Numerical results . . . 66
3.1 New mixed binary optimization test problems featuring cyclic symmetry . 69
3.2 Results obtained on the 25 analytical problems . . . 71
3.3 Design of compressor blades in a helicopter turbomachine . . . 74
4 Conclusion . . . 81

6 Further analysis on the optimization parameters 83

II Design of experiments in mixed continuous and discrete space 89

7 Design of experiments for continuous variables problems 91
1 Geometrical designs . . . 92
2 Marginal designs . . . 92
3 Low discrepancy designs . . . 93
4 Super-sample from kernel herding . . . 94

8 Design of experiments for mixed continuous and discrete variables problems 99
1 Introduction to mixed continuous and discrete variables DoEs . . . 99
2 Straightforward transformation of a continuous DoE into a mixed DoE . . . 100
3 Greedy-MDS: kernel herding and multi-dimensional scaling . . . 101
4 Adapted-Greedy: directly using a kernel defined in the mixed space . . . 103
5 Numerical experiments . . . 105
5.1 Design of experiments for a surrogate model with continuous, integer and categorical variables . . . 106
5.2 Design of experiments for cyclic-symmetry problems involving continuous and binary variables . . . 107
5.3 Design of experiments for times series . . . 112
6 Conclusions . . . 115

III Escaping from local minima 119

9 Escaping from local minima 121

1 Augmented DoE strategies for sampling a discrete variable space 122
2 Restart strategies 124
2.1 Hard restart 126
2.2 Complete soft restart 127
2.3 Partial soft restart 127
3 Numerical results 127
3.1 Results for the DS-MM function 128
3.2 Results for the Ackley function 128
4 Conclusion 130

General conclusion and perspectives **130**

A Mathematical Background **137**
1 Vector and matrix norms 137
2 Conditioning 138
3 Quadratic programming and mixed integer quadratic programming 138

B Encoding techniques and list of benchmark functions **141**
1 Categorical variables encoding techniques 141
2 Benchmark functions 142

Bibliography **149**

List of Figures

1 Illustration of a turbomachine (from [74]). 3
2 Examples of blade configurations for two different geometries (from [74]). 3
3 Illustration of a case with 23 blades with groups $AA - BB$ (from [74]). 6
4 Improvement of the interpolation set via LU factorization: X_0 and X_1 (top); X_2 and X_3 (bottom). 27
5 Performance profiles of the five solvers for the 25 analytical problems 73
6 Data profiles of the five solvers for the 25 analytical problems 73
7 Number of function evaluations to reach f^* up to $\tau = 10^{-3}$ for each of the 25 analytical problems for the five solvers (successful runs only) together with percentage of successes (in parentheses) 74
8 Performance profiles of the five solvers for the 10 highest dimension analytical problems 75
9 Data profiles of the five solvers for the 10 highest dimension analytical problems . . 75
10 Number of function evaluations to reach f^* up to $\tau = 10^{-3}$ for each of the 10 highest dimension analytical problems for the five solvers (successful runs only) together with percentage of success (in parentheses) 76
11 Performance profiles of the five solvers for the blade design application with 50 repetitions, accuracy $\tau = 10^{-5}$ 78
12 Data profiles of the five solvers for the blade design application with 50 repetitions, accuracy $\tau = 10^{-5}$ 78
13 Number of function evaluations to reach f^* up to $\tau = 10^{-5}$ for each of the 50 runs of the blade design application for the five solvers (successful runs only) together with percentage of success (in parentheses) 79
14 Performance profiles of the five solvers for the blade design application with 50 repetitions, accuracy $\tau = 10^{-3}$ 80
15 Data profiles of the five solvers for the blade design application with 50 repetitions, accuracy $\tau = 10^{-3}$ 80
16 Distribution of the binary component of the 50 solutions found by DFOb-d_{neck} for the blade design application. 81

17 The four solutions found by DFOb-d_{neck} for the blade design application (the continuous component has the same common value: $x^* = 0.03$) 82

18 Performance profiles of DFOb-d_{neck} with $nDoE = m+n+1$ and $nDoE = 2(m+n)+1$ for the 25 analytical problems 84

19 Data profiles of DFOb-d_{neck} with $nDoE = m + n + 1$ and $nDoE = 2(m + n) + 1$ for the 25 analytical problems 84

20 Number of function evaluations to reach f^* up to $\tau = 10^{-3}$ for each of the 25 analytical problems for DFOb-d_{neck} with $nDoE = m+n+1$ and $nDoE = 2(m+n)+1$ (successful runs only) together with percentage of success (in parentheses) 85

21 Accuracy of the surrogate models of the maximal torque (left) and the maximal power (right) built with designs of experiments obtained by the projected LHS method (in blue) and by the *adapted-greedy* algorithm (in red). The size of the designs, n_{DOE}, ranges from 20 to 100. 107

22 Median RMSE (in log-scale) of the estimation of benchmark function expectation obtained with 5 repetitions of the 6 methods: a random sampling and a standard LHS with rounded values for binary variables, 2 projected methods based on Sobol sequence and projected LHS, the greedy-MDS and adapted-greedy methods. 110

23 Distributions of estimation errors for each necklace of the Branin function obtained with 6 methods: a random sampling and a standard LHS with rounded values for binary variables, 2 projected methods based on Sobol sequence and projected LHS, the greedy-MDS and adapted-greedy methods. The size of the DoE is 100 points. . . 111

24 Safran's application: data profiles of the optimization runs with 100 initial DoE obtained with 3 methods: LHS with rounded values for binary variables, projected LHS, and adapted-greedy methods. 113

25 Safran's application: of the optimization runs with 100 initial DoE obtained with 3 methods: LHS with rounded values for binary variables, projected LHS, and adapted-greedy methods. 114

26 200 realizations of a Brownian motion. The thick blue curves are the 10 curves selected by the first iterations of the adapted-greedy algorithm. 115

27 200 realizations of a Max-stable process. The thick blue curves are the 10 curves selected by the first iterations of the adapted-greedy algorithm. 116

28 The eight individual one-dimensional functions for DS-MM example. The global minimum is located at $x = 0.808, y = (0, 0, 1, 0, 1)$, the objective-function value is -2.329. 124

29 Five points (blue stars) to complete the set of ten evaluated points (green circles) by the standard procedure (left) and by the augmented DoE method (right). 125

30 Ten points (blue stars) to complete the set of ten evaluated points (green circles) by the standard procedure (left) and by the augmented DoE method (right). 125

31 The best current objective-function values versus number of evaluations of DFOb-d_{neck} with standard, augmented add-point step, hard restart, complete soft restart, and partial soft restart on the DS-MM function. 129

32 Solutions obtained by DFOb-d_{neck} with standard, augmented add-point step, hard restart, complete soft restart, and partial soft restart on the DS-MM function. 130

33 Ackley function with two continuous variables and four binary variables (corresponding 6 levels: $y_1 = (0,0,0,0), y_2 = (0,0,0,1), y_3 = (0,0,1,1), y_4 = (0,1,0,1), y_5 = (0,1,1,1), y_6 = (1,1,1,1)$). 131

34 The best current objective-function values versus number of evaluations of DFOb-d_{neck} with the five options standard, augmented add-point step, hard restart, complete soft restart, and partial soft restart on the Ackley function. 132

35 Zoom on the best current objective-function values versus number of evaluations of DFOb-d_{neck} with the five options. 132

36 The two solutions obtained with DFOb-d_{neck} and the five options on the Ackley function. 133

List of Tables

2 Fuel consumption for aircraft and impact of the small reduction from 2016. 2

3 Number of distinct solutions for n blades (from [74]). 4

4 Solver parameters and options used for the benchmark. 68

5 Analytical necklace-optimization benchmark problems 70

6 List of benchmark functions. 108

7 Computational times and relative errors of the expectation estimation obtained for the function f_1 with the functional data of Brownian motion. 4 methods are evaluated: the adapted-greedy and MDS-greedy methods coupled with the global alignment kernel, the LHS method and the quantization method coupled with the FPCA dimension reduction method. The best values among the 4 methods are indicated in bold. 115

8 Computational times and relative errors of the expectation estimation obtained for the function f_1 with the functional data of Max-stable process. 4 methods are evaluated: the adapted-greedy and MDS-greedy methods coupled with the global alignment kernel, the LHS method and the quantization method coupled with the FPCA dimension reduction method. The best values among the 4 methods are indicated in bold. 117
9 Computational times and relative errors of the expectation estimation obtained for the function f_2 with the functional data of Brownian motion. 4 methods are evaluated: the adapted-greedy and MDS-greedy methods coupled with the global alignment kernel, the LHS method and the quantization method coupled with the FPCA dimension reduction method. The best values among the 4 methods are indicated in bold. 117
10 Computational times and relative errors of the expectation estimation obtained for the function f_2 with the functional data of Max-stable process. 4 methods are evaluated: the adapted-greedy and MDS-greedy methods coupled with the global alignment kernel, the LHS method and the quantization method coupled with the FPCA dimension reduction method. The best values among the 4 methods are indicated in bold. 118

List of Algorithms

1 Derivative based trust region algorithm, see [81] 19

2 Improving poisedness of X via LU factorization, see [27] 26

3 Improving poisedness of Z in the mixed trust region B, extended from [27] 36

4 Exploitation phase of DFOb-d_H (Step 1) **at iteration** k 38

5 Exploration phase of DFOb-d_H (Step 1.5) **at iteration** k 40

6 DFOb-d_H algorithm 42

7 Greedy algorithm for continuous-variable problems 97

8 Projected DoE for discrete variables 101

9 Greedy-MDS algorithm. 103

ACRONYMS

BO	: Bayesian Optimization
BONMIN	: Basic Open-source Nonlinear Mixed INteger programming
DFO	: Derivative Free Optimization
DFOb	: DFO trust-region method with mixed binary variables
DFOb-d_H	: DFOb with Hamming distance
DFOb-d_{neck}	: DFOb with necklace distance
DoEs	: Design of Experiments
EGO	: Efficient Global Optimization
GA	: Genetic Algorithm
GP	: Gaussian Process
IPOPT	: Interior Point OPTimizer
LHS	: Latin Hypercube Sampling
MADS	: Mesh Adaptive Direct Search
MINLP	: Mixed-Integer NonLinear Programming
MIQP	: Mixed Integer Quadratic Programming
NLP	: NonLinear Programming
NOMAD	: Nonlinear Optimization by Mesh-Adaptive Direct Search
QP	: Quadratic Programming
RBF	: Radial Basis Function
ROM	: Reduce Order Model
SA	: Simulated Annealing
SCIP	: Solver for Constraint Integer Programming

NOTATIONS

f	: objective function
m	: number of continuous variables
x	: continuous-variable vector
$\bar{x}$	: upper bound for x
$\underline{x}$	: lower bound for x
X	: continuous interpolation set
$\hat{X}$	: continuous normalized interpolation set
m_k	: continuous quadratic model at iteration k
y	: binary-variable vector
z	: mixed-variable vector $z = (x, y)$
n	: number of binary variables
$f(x)$	: objective function with continuous variables
$f(x, y)$	: objective function with mixed continuous and binary variables
$\nabla_x f$	: derivatives of f with respect to x
Z	: mixed variables interpolation set
$\hat{Z}$	: normalized mixed variables interpolation set
$\tilde{m}_k$	: mixed variables quadratic model at iteration k
p	: cardinality of Z
$\|.\|_F$	: Fröbenius norm
$\|.\|_\infty$	: l_∞ norm
$\|.\|_p$	: l_p norm
$\|.\|_2$	: l_2 norm
Δ_x	: trust-region radius relative to the x search subspace
Δ_y	: trust-region radius relative to the y search subspace
$\Delta_{x,0}$	: initial value of Δ_x
$\Delta_{y,0}$	: initial value of Δ_y
H_k	: Hessian matrix at iteration k
ϕ	: natural basis of polynomials

$\mathcal{M}$	: class of fully-linear (or fully-quadratic) model
$B(x_0, \Delta_x)$	: continuous l_2 ball centered at x_0 of radius Δ_x
$\mathcal{B}(y_0, \Delta_y)$	: binary l_2 ball centered at y_0 of radius Δ_y
$B_{y_0}(x_0, \Delta_x)$	: continuous l_2 ball with fixed binary part $y = y_0$ of radius Δ_x
d_H	: Hamming distance
d_{neck}	: necklace distance
$Rot^r(y)$	: rotation of y by r positions
$nDoE$	: number of points in the interpolation set

1 Introduction

> *It is through science that we prove, but through intuition that we discover.*
>
> **— Henri Poincaré**

Contents

1 The motivating application **2**
1.1 Motivation 2
1.2 Turbomachines 2
1.3 Optimization challenges 3
2 Reduced Order Model (ROM) **5**
3 Thesis contributions **6**
4 Outline of the thesis **7**
5 Publications **8**

This thesis stems from the collaboration between IFP Energies Nouvelles, Safran Tech and École Nationale de l'Aviation Civile (ENAC). The aim is to study and provide an algorithm to solve mixed continuous and discrete non-linear problems arising from industrial applications.

One of the motivating applications of the thesis is the optimal design of the turbine blades of a helicopter engine. The objective is to maximize the compressor efficiency under some stability constraints (minimizing vibrations). This involves very expensive mechanical simulations (solid and fluid mechanics), typically involving computer runs of 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. The lack of analytical form of the objective function, the appearance of discrete variables as well as the symmetry property provides challenging but also interesting problems.

This chapter is structured as follows: Section 1 introduces Safran's application context. In Section 2, we describe the idea of the reduced-order model which is currently used in practice. Section

3 presents the main contributions of the work. Finally, the outline of the thesis is presented followed by the list of the publications.

1 The motivating application

In the context of turbomachine design, manufacturing operations possibly affect the structural cyclic symmetry and can increase the forced response amplitudes. As a result, vibration appears in the system and destroys the engine. Therefore, minimizing vibration to obtain a stable engine is important.

1.1 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 [3, 53], and also with a huge budget for the maintenance and manufacturing of the new engines.

Table 2 shows the fuel consumption for aircraft: if we can reduce by 0.5% the amount of fuel bust, 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.

Tab. 2.: Fuel consumption for aircraft and impact of the small reduction from 2016.

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

1.2 Turbomachines

There are several ways to optimize cost in aviation: through optimizing trajectories, arrangements of passengers (through the 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 Figure 1 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. 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 [74].

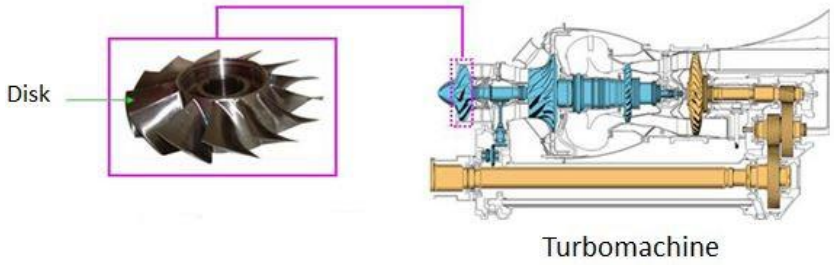

Fig. 1.: Illustration of a turbomachine (from [74]).

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

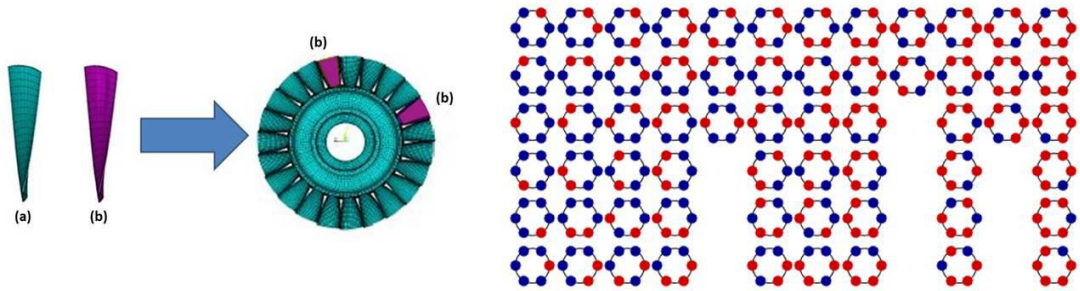

Fig. 2.: Examples of blade configurations for two different geometries (from [74]).

1.3 Optimization challenges

Due to the complexity of fluid dynamic systems, we are not able to obtain an analytical form of the objective function, as well as physical-constraints functions. Therefore, the optimization

problem that we are dealing with is a derivative-free optimization or blackbox simulator problem. Derivative-free optimization is a field of applied mathematics that concerns optimization problems without any information of derivatives of the objective function. We list in the following some mathematical challenges that must be addressed:

- No information of derivatives of the objective function and, possibly of some physical-constraint functions.
- There is a combinatorial problem linked to the presence of discrete variables.
- The cyclic symmetry property of the problem yields a huge number of redundant arrangements.

Regarding this cyclic-symmetry challenge, remark that: two blade disks that differ only by a rotation of the pattern around the disk not only lead to a same value of the objective function, but also correspond to identical compressors. The number of equivalent solutions also rapidly increases with the number of blades, n, as illustrated in Table 3. Particularly, in this type of application, one simulation can cost several hours or more. Therefore, we aim to avoid recomputing during the optimization process, the costly objective-function value at equivalent configurations.

Tab. 3.: Number of distinct solutions for n blades (from [74]).

Total number of blades on the disk	Number of distinct arrangements	Total number of arrangements
2	3	4
3	5	8
6	14	64
12	352	4096
20	52488	1048576

The next section presents the Reduced Order Model (ROM) to reduce the computational cost of optimization for such highly combinatorial problems.

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 identical solutions. Recall that when the number of blades is increasing, the number of such equivalent solutions rapidly increases (Table 3).

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* (changes from A to B of from B to A) or if there exists a series of groupings of two consecutive blades of the form AA or BB (e.g., $7B5A$: 7 consecutive blades B and 5 consecutive blades A), 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.

If the optimal distribution of the blade geometry presents few alternations of the two different shapes, it can roughly be described by a distribution of patterns of the types AA or BB. On the other hand, if the optimal distribution presents many alternations of the two different shapes, one should rather 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 that deals with $\frac{n}{2}$ groups instead of n blades. The idea of ROM is to consider $n/2$ groups of two blades with two possibilities for each group. We group either patterns AA and BB if there are few switches (*high coupling*), or AB and BA in the case where there are many switches (*low coupling*). For illustration, a disk with 10 blades consists of 108 distinct distributions of two types of blades, whereas a ROM approach leads to a problem involving 5 groups and only 8 distinct distributions.

A 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 groups of 2 blades in order to provide good initial guesses for the orginal n blades problem.Then, a local search limited

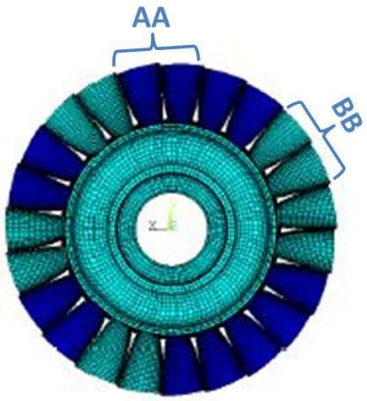

Fig. 3.: Illustration of a case with 23 blades with groups $AA - BB$ (from [74]).

to, for instance performing one *flip* changing one A geometry to a B geometry or vice-versa from these initial guesses provides what are may call a "local" solutions to the original problem.

ROM reduces the dimension of the discrete space, which in turn tends to reduce the computation time required for each sub-problem. A disadvantage however is that it is likely to remove a large number of possibilities which will not be explored, leading thereby to a sub-optimal solution. For instance, for the problem involving $n = 12$ blades, ROM removes 324 disk configurations. Out of 352 distinct configurations as each of the two subproblem considers only 14 distinct configurations. This motivates our study to look for another method to address the above-mentioned cyclic symmetry property to ensure that all the distinct arrangements are considered. In Chapter 5 of Part I, we will propose an adapted mathematical optimization method for this type of problem.

3 Thesis contributions

The three main contributions of the thesis are:

- Development of a new mixed continuous and discrete variables derivative-free optimization method based on trust-region method, [27] for cyclic-symmetry problems, called DFOb-d_{neck}. The originality of our approach is the introduction of the *necklace distance* which avoids recomputing expensive simulations corresponding to the equivalent configurations.

- A theoretical convergence proof of the mixed-integer DFO method (evoked in [25] but never published) is provided. This proof is then further extended into a convergence proof of DFOb-d_{neck}.

- A new design of experiments method based on a kernel-embedding probability distribution for mixed discrete variables problems.

A secondary contribution includes the proposition of benchmark cyclic-symmetry test functions and numeric comparison of our optimization approach with state-of-the-art methods such as NOMAD, RBFOpt, and DFLBOX.

4 Outline of the thesis

We begin by recalling in Chapter A the notation and some mathematical background necessary for the readers to follow the thesis. The rest of the thesis is composed of three parts that are detailed as follows:

Derivative-free mixed binary necklace optimization. In this first part, we focus on optimization methods for solving derivative-free problems:

- Chapter 2 gives an overview of blackbox optimization methods for mixed discrete variables problems.

- Chapter 3 carries out a review of the continuous DFO trust-region method.

- Chapter 4 focuses on the derivative-free trust-region methods for mixed binary problems. Our contribution is the convergence proof of the original method proposed in [25].

- Chapter 5 presents our two main contributions. First, we introduce a new optimization method for cyclic-symmetry problems, named DFOb-d_{neck} which is based on DFOb-d_H. Our second contribution, is the proposal of a set of benchmark functions adapted to cyclic-symmetry problems. These functions were used as tests for our method and for state-of-the-art methods. A simplified simulation provided by Safran is also tested.

- Chapter 6 shows an analysis result of methods about the impact of number of points in the interpolation set.

Design of experiments in mixed continuous and discrete variables space. Blackbox model-based methods can be significantly impacted by the *initial design* (the points at which the objective function is evaluated before optimization). The choice of the initial design has a strong influence on how well the search domain will be explored.

- Chapter 7 reviews about design of experiments methods in continuous space and in mixed discrete space.

- Chapter 8 presents one of the main contributions of the thesis: the introduction of two extended kernel-embedding probability distribution DoEs to mixed discrete variables space. We also give examples with a large range of application types.

Escaping from local minima. This part is dedicated to escaping from local solutions by applying restart techniques.

5 Publications

The work of this thesis yielded the following publications:

- T. T. Tran, D. Sinoquet, S. D. Veiga, and M. Mongeau. (2020) An adapted derivative-free optimization method for an optimal design application with mixed binary and continuous variables. In: Le Thi H., Le H., Pham Dinh T., Nguyen N. (eds) Advanced Computational Methods for Knowledge Engineering. ICCSAMA 2019. Advances in Intelligent Systems and Computing, vol 1121. Springer. `https://doi.org/10.1007/978-3-030-38364-0_8`.

- T. T. Tran, D. Sinoquet, S. D. Veiga, and M. Mongeau (2021) Derivative-free mixed binary necklace optimization for cyclic-symmetry optimal design problems. Accepted (on Aug 27, 2021) to be published in *Optimization & Engineering*.

- T. T. Tran, D. Sinoquet, S. D. Veiga, and M. Mongeau (2021) Design of Experiments for mixed continuous and discrete variables problems. (In preparation, to be submitted).

Part I

Derivative-free mixed binary necklace optimization

2

Introduction to blackbox optimization methods for mixed integer variables

This part corresponds to the article entitled “Derivative-free mixed binary necklace optimization for cyclic-symmetry optimal design problems”, that is accepted for publication in Optimization and Engineering (submitted on 15.03.2021, revision submitted on 18.06.2021, accepted on 27.08.2021), and a technical report entitled “Nonlinear optimization of mixed continuous and discrete variables for black-box simulators", `https://hal-ifp.archives-ouvertes.fr/hal-02511841/document`.

In this chapter, we give an overview of derivative-free optimization for mixed integer variables. We consider the following bound constrained optimization problem:

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

where $x \in \mathbb{R}^m,\ y \in \mathbb{I}^n$ are the continuous and discrete variables, respectively, and $\mathbb{I}$ denotes the discrete space (*e.g.*, integer, binary or categorical variables). In this thesis, we focus on solving problem (1) with mixed continuous and binary variables, *i.e.*, $\mathbb{I} = \{0, 1\}$. For integer or categorical variables problems, encoding can be applied to reformulate the problem so as to deal with continuous and binary variables in many cases. There is a wide study in encoding categorical variables. To give the readers a summary, we make a list of encoding methods in Appendix B.

The objective function f is the output of a blackbox numerical simulator with two features:

1. The derivative, ∇f, is not available, we only have access to function values, and

2. It is computationally expensive to evaluate the function f.

There is a large body of works in the operation research community that regards Mixed-Integer NonLinear Programming problems (MINLP), see for instance [16]. Most deterministic algorithms for solving MINLP are based on branch-and-bound methods. Briefly, the branch-and-bound algorithm is based on recursively sub-dividing the set of possible solutions during the branching step, and estimating bounds on the optimal objective-function value in each branch (the “cut”

or “bound” operation) to find a solution (see *e.g.*, [33]). In [93], the authors do not create a search tree but relax the integrality constraints via a sine function that penalizes the variables for not being integers. Remark that relaxing the binary variables is not possible in our application context. While convex MINLPs can be tackled by several available software, for instance BONMIN [17] or SCIP [2], nonconvex problems are more difficult and usually require convexification and reformulation strategies. Such strategies are either impossible (reformulation) or still need to be developed when dealing with black-box optimization. This is a real challenge, especially in our context of objective functions that are computationally expensive to evaluate (several hours or even days for a single evaluation). Therefore, we choose in this thesis not to focus on (meta) heuristic methods (*e.g.*, evolutionary algorithms [44] and simulated annealing) due to the large number of objective function evaluations that such approaches require.

Blackbox optimization algorithms only use the values of the objective function at evaluation points to perform the optimization. An overview introduction can be found in [6]. A review and comparison of algorithms and software for derivative-free optimization is presented in [85]. Generally, derivative-free optimization methods can be classified in two main categories which are *direct search methods* and *model-based methods*. Direct Search Methods use systematic rules to explore the domain. Among direct search methods, we can count:

- **Pattern search methods** [97]. They consider the behavior of the objective function on a pattern of points over a rational lattice. A search exploratory strategy moves points in the vicinity of the current iterate followed by a so-called local poll step. A convergence proof for nonsmooth functions approaches can be found in [12]. Extension work to mixed variables problems is introduced in [11].

- **Heuristic methods**. They include the popular genetic algorithms (GA) [54] which are based on the evolution of a population of points. More precisely, by sorting the individuals for reproduction according to an adhoc so called *fitness* function, the best individuals (with highest fitness values) are selected to generate new candidate points. For example, *crossover phase* is operated to generate *offsprings*, i.e., new generations of solutions. Then possibly a mutation step is added to have complete new individuals. The algorithm terminates when the set of offsprings is not significantly different from the parents (no more improvement). This method generally requires a large number of evaluations to find a good solution. Some works extend genetic algorithm to mixed variables problem ([111] is one example). This extension is usually straightforward by choosing adapted crossover and mutation operators.

- **Simplex search methods**. These heuristic methods (not to be confused with Dantzig’s Simplex method for linear programming) include the famous Nelder-Mead algorithm [79] which is based on the idea of replacing the worst vertex in the set of evaluated points

with a new one that gives a better value for the objective function by using geometrical transformations such as expansion, reflection, contraction.

A widely cited DFO algorithm in the family of pattern search methods, NOMAD [1, 61], implements the Mesh Adaptive Direct Search (MADS) algorithm [8] for black-box optimization under general nonlinear constraints. MADS is an extension of Torczon's generalized pattern search algorithms [97, 7]. MADS principally relies on two main steps. The *search* step is flexible enough to allow local and global explorations with generic strategies such diverse as Latin-Hypercube Sampling (LHS), variable neighborhood search [9] or a Nelder Mead method [14]. The *poll* step is critical to the local convergence proof. It involves evaluating the objective function on a discrete grid that is dynamically updated. More recently, [26] introduced a search strategy that automatically constructs quadratic models to try and find promising trial points. An extension of MADS to handle integer variables problems is presented in [13]. The main idea is to harmonize the minimal granularity of variables with the finest mesh containing all trial points. A different approach for direct search methods for bound constrained problems, named DFLBOX is introduced in [63]: instead of searching in all the components of the variable, DFLBOX searches the solution component-wise. The algorithm consists of a continuous and a discrete search procedure that calculates the step sizes. It removes the sufficient decrease condition on the discrete variables and updates the iterate by choosing the coordinate that yields the largest objective function reduction. DFLBOX is extended to a more general constrained problems using the sequential quadratic penalty approach in [62].

Among model-based methods, ones can identify two main categories in literature:

- **Derivative-free trust-region methods**. These include for instance the DFO algorithm of [27], and NEWUOA [86]. These methods are based on local quadratic models that interpolate the objective function at some evaluated points. The models need to satisfy a so-called *fully-linear* or *fully-quadratic* property to ensure the local convergence [27]. An exploitation phase minimizes the quadratic approximation within a *trust region*, while an exploration step explores the domain in view of diversification. An extension of this work was first presented in [25] at the SMAI conference.

- **Response surface methods** which include Radial Basis Function (RBF) based optimization methods and the Efficient Global Optimization (EGO) method or Bayesian Optimization based on Gaussian Processes (GP). The surrogate models used in these methods are *global* models, *i.e.*, they use a single substitute of the objective function that aims to be sufficiently predictive in the whole search domain to detect areas of interest with good values of the objective function (*exploration*), and that can be refined in these areas (*exploitation*). Note that these exploitation and exploration objectives are similar to the goals of the poll

and search steps of MADS. The RBF method for global optimization was introduced by Gutmann [48], several variations followed [90, 89, 30], and its extension to mixed discrete variables problems [77, 51]. EGO [56] is based on a Gaussian process surrogate and an adaptive strategy to propose new evaluation points based on the so-called expected improvement criterion, which balances between exploration and exploitation. These types of methods tackle two main disadvantages, first, the *scale of dimension*, and the difficulty of giving an explicit convergence proof. In [76], Gaussian process kernels that are products of continuous and discrete kernels are integrated into an EGO method framework; the resulting mixed *categorical* (involving integer variables not related to effective quantities) optimization problem is then solved by NOMAD. The strengths and weaknesses of various types of kernels for Gaussian processes are discussed in [84].

In recent years, there have been considerable studies dedicated to improving derivative-free optimization algorithms. Some of them focused on overcoming the limitation of Bayesian Optimization for high dimensional problems. The EGO-CMAS algorithm proposed in [72] is one example. In this algorithm, an iteration starts with EGO search, then an interruption is set, to be followed by a CMA-ES search over whole starting points with the initial size (the Covariance Matrix is computed from simulated points). A Random Embedding BO (REMBO) method was proposed in [107] where instead of optimizing in the original high-dimension space, the function is optimized in a reduced-space variable defined by a random projection based on an assumption of a reduced effective dimensionality of f. In the same year, [58] proposes the Add-GP-UCB approach which decomposes f as a sum of functions defined in lower-dimensional disjoint spaces and then use GP for each disjoint domain. SEGOKPLS, SEGOKPLS+K [18] use PCA (Principal Component Analysis) to find the most informative components of the variable set, and then apply EGO with an adapted user-defined kernel. A very recent work [95] proposes a method named CobBO (Coordinate Backoff Bayesian Optimization). If backoff stopping rules are met, the farthest point (from the current trust region center) of a random sampling is chosen with an exploration goal, otherwise a Bayesian optimization based on Radial Basis Functions is applied within the current trust region. There are other methods based on EGO to deal with the problem of dimensionality. For instance, SEGOMOE (Super Efficient Global Optimization with Mixture of Experts) [15] constructs different local approximations (*experts*) for different domains in the design space. These local approximations can be tailored to deal with disparate local trends in the function, including flat regions, discontinuities, and regions where there are strong nonlinearities. Another interesting track combines the response surface algorithms and trust-region methods. A first stage involves the combination of RBF and trust-region method, for instance, ORBIT (Optimization by radial basis function interpolation in trust region), [108], and BOOSTERS, [82]. The idea is to replace the quadratic model with more flexible radial basis functions. The second stage combines EGO and trust-region methods, for instance, TRIKE (Trust-Region Implementation in Kriging-based optimization with Expected improvement) [91], CYCLONE (CYClic Local search

in OptimizatioN using Expected improvement) [91], TURBO (TrUst-Region BO solver) [39], TREGO (Trust-Region Framework for Efficient Global Optimization) [35].

This part gave an overview of derivative-free optimization algorithms, a class of methods built to tackle expensive black-box optimization problems. In the next chapter, we present in more detail derivative-free optimization trust-region methods.

3 Derivative-free trust-region methods for continuous variables problems

Contents

1 Trust-region framework basics **17**

2 Derivative-free trust-region methods **19**

2.1 Geometry of the interpolation set 20

2.2 Fully-linear and fully-quadratic models 22

2.3 Building the trust-region model 24

2.4 Model improvement 26

3 Conclusion **28**

In this chapter, we address the bounded constrained problem of the form

$$\begin{cases} \min\limits_{x} f(x) \\ x \in [\underline{x}, \bar{x}], \end{cases} \tag{2}$$

where $x \in \mathbb{R}^m$ is a vector of continuous variables. The objective function f is the output of a black-box numerical simulator. We further assume that f is smooth with respect to x and bounded from below to guarantee convergence of the algorithm (see [27, 6]). We also assume that derivatives of f exist but unavailable.

We now begin with the basic background required for the trust-region method framework. We first outline the basic framework of trust-region method, and then describe the derivative-free trust-region methods.

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 [81] for details. In trust-region methods, at each iteration k, we build a quadratic model $\widetilde{m}_k$ around the current iterate x_k. Remark that this model is a tangent model with exact gradient at the center of the trust region.

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, Δ_k), where $\Delta_k > 0$:

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

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

$$\min_{s \in B(0,\Delta_k)} \widetilde{m}_k(x_k + s), \tag{3}$$

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

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

where $\kappa_d \in (0, 1]$ is a constant. Taylor's theorem guarantees the existence of a trust region that guarantees condition, but it does not give its precise radius. Therefore, updating and adjusting the trust-region radius after each iteration is necessary. Given an approximate solution s_k of (3), the trust-region radius is updated depending on the ratio of actual improvement and predicted improvement

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{\widetilde{m}_k(x_k) - \widetilde{m}_k(x_k + s_k)}.$$

If the model reduction matches well the actual reduction of the objective function (when $\rho_k > 0$ and near 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 [81]

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

0. Initialization. Compute $f(x_0), \nabla f(x_0)$. Set $k = 0$

1. Model definition. Build the quadratic model $\widetilde{m}_k$ in $B(x_k, \Delta_k)$

2. Solve sub-problem (3)

$$s_k \in \operatorname{argmin}_{s \in B(x_k, \Delta_k)} \widetilde{m}_k(x_k + s).$$

3. Center update. Compute

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{\widetilde{m}_k(x_k) - \widetilde{m}_k(x_k + s_k)}.$$

Set:

$$x_{k+1} = \begin{cases} x_k + s_k & \text{if } \rho_k \geqslant \eta_0, \\ x_k & \text{if } \rho_k < \eta_0. \end{cases}$$

4. Trust region update

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{max}\} & \text{if } \rho_k \geqslant \eta_1, \\ \Delta_k & if\ \eta_0 \leqslant \rho_k < \eta_1, \\ \gamma_0 \Delta_k & \text{if } \rho_k < \eta_0. \end{cases}$$

When the derivatives are not available, we build an 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 the interpolation set to improve the accuracy of the model. This is detailed in the next section.

2 Derivative-free trust-region methods

The basic idea of DFO trust region methods [27] 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 methods. At each iteration, we solve the sub-problem to find the possible candidate for the next iteration. Similarly to derivative-based trust region methods, if the model is qualified as valid (according to the ratio between the actual improvement and the predicted improvement), we decide to increase, decrease or keep the current trust-region radius and choose the solution of the sub-problem as the center of the new trust-region, or not.

We keep running the algorithm until stopping criteria are met, typically when the trust region and the norm of the gradient of the model are small enough.

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

- 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 focus on the model construction and the necessary properties of the geometry of the interpolation set to control the error bounds of the models.

2.1 Geometry of the interpolation set

Consider a set of sample points given by

$$X = \{x^0, x^1, \dots, x^p\}, \text{where } x^i \in \mathbb{R}^m, i = 1, 2, \dots, p. \tag{4}$$

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

We would like to build a quadratic model $\widetilde{m}(x)$ which interpolates the points in X. If $p_1 = (m+1)(m+2)/2$, the problem (2) is a quadratic interpolation problem. If $p_1 > (m+1)(m+2)/2$, the problem is over-determined, whereas $p_1 < (m+1)(m+2)/2$, it is under-determined. In typical applications, the problem is under-determined since the simulations are computationally expensive.

Let $\widetilde{m}(x)$ denote the polynomial of degree d (e.g., $d = 1$ with $1 < p_1 = m + 1$ *or* with $d = 2, m + 1 < p_1 \leqslant (m+1)(m+2)/2$) that interpolates $f(x)$ at the points in X that:

$$\widetilde{m}(x^i) = f(x^i), \;\; i = 0, \dots, p. \tag{5}$$

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

$$\widetilde{m}(x) = \sum_{i=0}^{p} \alpha_i \phi_i(x), \tag{6}$$

for some coefficients $\alpha_i, i = 0, \ldots, p,$. It is clear that $\widetilde{m}(x)$ is determined if the values of $\alpha_0, \ldots, \alpha_p$ are determined. From (5) and (6), the coefficients α are found by solving the following linear system

$$\begin{pmatrix} \phi_0(x^0) & \phi_1(x^0) & \ldots \phi_p(x^0) \\ \phi_0(x^1) & \phi_1(x^1) & \ldots \phi_p(x^1) \\ \vdots & \vdots & \vdots \\ \phi_0(x^p) & \phi_1(x^p) & \ldots \phi_p(x^p) \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_p \end{pmatrix} = \begin{pmatrix} f(x^0) \\ f(x^1) \\ \vdots \\ f(x^p) \end{pmatrix}, \tag{7}$$

or, equivalently, in matrix form:

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

where $M(\phi, X)$ and $f(X)$ represent respectively the matrix and right-hand side vector of the system 7. If $M(\phi, X)$ is non-singular (*i.e.*, has full rank) then the system has a 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 $\widetilde{m}(x)$ exists and is unique.

In [27] 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 [27])

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

with $\Delta = \Delta(X) = \max\limits_{1 \leqslant i \leqslant p} \|x^i - x^0\|$.
In practice, we choose for ϕ the natural basis

$$\{1, x_1, x_2, \ldots, x_m, \frac{1}{2}x_1^2, x_1x_2, \ldots, \frac{1}{(d-1)!}x_{m-1}^{d-1}x_m, \frac{1}{d!}x_m^d\}, \tag{8}$$

Definition 3.1. (**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

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

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

Definition 3.2. ($\Lambda-$ **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, one has

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

   or, equivalently,

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

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

   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 always hold.

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, the class of fully-linear and fully-quadratic models is introduced and detailed in [27].

We suppose that x_0 is given as the initial iterate. We define the level set

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

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

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

Assumption 3.1. *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 3.3. (Class of fully-linear models) Let a function $f : \mathbb{R}^m \to \mathbb{R}$, that satisfies Assumption (3.1), be given. A set of model functions $\mathcal{M} = \{\widetilde{m}(x) : \mathbb{R}^m \to \mathbb{R}, \widetilde{m}(x) \in \mathcal{C}^1\}$ 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 $\widetilde{m}(x+s) \in \mathcal{M}$, 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

   $$\|\nabla f(x+s) - \nabla \widetilde{m}(x+s)\| \leqslant \kappa_{eg}\Delta, \ \forall s \in B(0,\Delta), \tag{9}$$

   - the error between the model and the function satisfies

   $$|f(x+s) - \widetilde{m}(x+s)| \leqslant \kappa_{ef}\Delta^2, \ \forall s \in B(0,\Delta). \tag{10}$$

   Such a model $\widetilde{m}$ 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 $\widetilde{m} \in \mathcal{M}$ is fully linear on $B(x;\Delta)$
   - or find a model $\bar{m} \in \mathcal{M}$ 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 3.2. *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 3.4. (Class of fully-quadratic models) Let a function $f : \mathbb{R}^m \to \mathbb{R}$, that satisfies Assumption (3.2) , be given. A set of model functions $\mathcal{M} = \{\widetilde{m}(x) : \mathbb{R}^m \to \mathbb{R}, \widetilde{m}(x) \in \mathcal{C}^2\}$ 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 $\widetilde{m}(x+s) \in \mathcal{M}$, with Lipschitz continuous Hessian 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

   $$\|\nabla^2 f(x+s) - \nabla^2 \widetilde{m}(x+s)\| \leqslant \kappa_{eh}\Delta, \ \forall s \in B(0,\Delta), \tag{11}$$

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

   $$\|\nabla f(x+s) - \nabla \widetilde{m}(x+s)\| \leqslant \kappa_{eg}\Delta^2, \ \forall s \in B(0,\Delta), \tag{12}$$

   - the error between the model and the function satisfies

   $$|f(x+s) - \widetilde{m}(x+s)| \leqslant \kappa_{ef}\Delta^3, \ \forall s \in B(0,\Delta). \tag{13}$$

   Such a model $\widetilde{m}$ 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 $\widetilde{m} \in \mathcal{M}$ is fully quadratic on $B(x;\Delta)$

   - or finds a model $\bar{m} \in \mathcal{M}$ that is fully quadratic 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 functions evaluations to build a fully quadratic model.

2.3 Building the trust-region model

Next, we indicate how to construct a fully-linear (or fully-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.

Case where the number of interpolation points is exactly equal to $m+1$.

We build a linear interpolation function $L_X(x) := \alpha_0 + \alpha_1^T x$ where (α_0, α_1) is the solution of

$$[1 \ X] \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} = f(X). \tag{14}$$

If the interpolation set X is poised, then there exists only one linear interpolation model whose coefficients satisfy (14).

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

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

$$\min_{\alpha_0, \alpha_1} \left[\sum_{i=0}^{p} (\alpha_0 + \alpha_1^T x^i - f(x^i))^2 \right]. \tag{15}$$

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

$$\begin{array}{l} \min\limits_{(\alpha_0, \alpha, H)} \dfrac{1}{2} \sum\limits_{i=1}^{m} \sum\limits_{j=1}^{m} h_{i,j}^2 \\ \alpha_0 + \alpha^T x^i + \frac{1}{2} (x^i)^T H x^i = f(x^i), \ \ i = 0, \ldots, p, \\ H = H^T. \end{array} . \tag{16}$$

Case where the number of interpolation points is exactly $\frac{(m+1)(m+2)}{2}$.

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

$$\alpha_0 + \alpha^T x^i + \frac{1}{2} (x^i)^T H x^i = f(x^i), \ \ i = 0, \ldots, p. \tag{17}$$

One notes that the quadratic interpolation is fully linear if X is poised. Comparing with the linear interpolation, the error bounds of the quadratic interpolation are tighter, and the function M_X provides both approximations of the gradient and the Hessian. Therefore, in terms of iterations, quadratic interpolation can converge much faster, but it requires more points in the interpolation set.

Case where 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.

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 [27], the authors give two main ways to improve the sample set: one based on the Lagrange polynomials and one based on the LU factorization. In our study, we focus on LU factorization to improve the model. The following model improvement algorithm is applied before each model update.

Algorithm 2: Improving poisedness of X via LU factorization, see [27]

0. Initialization

Initialize the pivot polynomial basis with some basis, *e.g.*, the monomial basis (8), $u_i(x) = \phi_i(x), i = 0, \ldots, p$. Select pivot threshold $\xi > 0$.

For $i = 0, \ldots, p$

1. **Point selection:** If j_i is found such that $|u_i(x^{j_i})| \geqslant \xi$, 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 $j = i+1, \ldots, p$
$$u_j(x) \leftarrow u_j(x) - \frac{u_j(x^i)}{u_i(x^i)} u_i(x).$$

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):

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

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

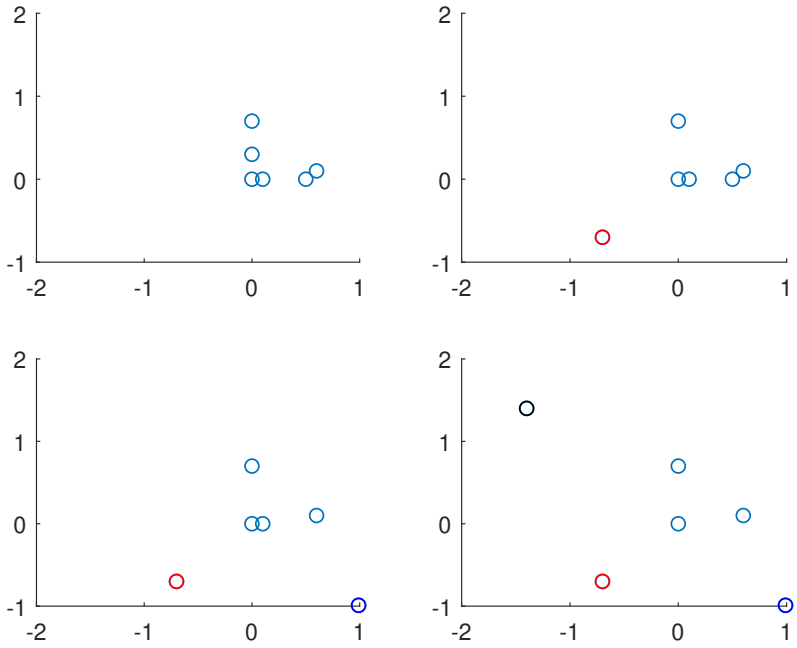

Fig. 4.: Improvement of the interpolation set via LU factorization: X_0 and X_1 (top); X_2 and X_3 (bottom).

Another way to improve the model is to use Lagrange polynomials. In [86] the author uses the definition of $\Lambda = \max_{0 \leqslant i \leqslant p} \max_{x \in B} |l_i(x)|$ and maintains 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 the 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 [27]. The validity of the convergence of our model is based on the fully-linear or fully-quadratic property in definition (3.3) and (3.4),

which is guaranteed by the poisedness of the interpolation set. Besides, the model must satisfy inequality (10.10) in [27]

$$\forall k,\ \widetilde{m}_k(x_k) - \widetilde{m}_k(x_k + s_k) \geqslant \frac{\kappa}{2}\|\nabla \widetilde{m}_k(x_k)\| \min(\frac{\|\nabla \widetilde{m}_k(x_k)\|}{\|H_k\|}, \Delta_k),$$

for some constant $\kappa \in (0, 1)$. The first-order critical-point convergence $\lim_{k \to +\infty} \|\nabla f(x_k)\| = 0$ is shown in Theorem (10.13) in [27].

3 Conclusion

In this chapter, we presented a basic framework of trust-region methods as well as DFO trust-region methods for continuous problems. We emphasized that the local convergence is guaranteed based on the so-called *fully-linear* and *fully-quadratic* properties. In the next chapter, we present an extension of DFO trust-region methods from continuous variables problems to mixed continuous and binary problems that was initially proposed in [25].

4

DFOb-d_H: A derivative-free trust-region method for mixed binary problems

Contents

1	The quadratic model	**30**
2	Initial interpolation set	**37**
3	Exploitation phase	**38**
4	Exploration phase	**39**
5	Local convergence results	**43**
6	Conclusion	**49**

This chapter discusses the main ingredients of the DFOb-d_H algorithm, an extension of DFO trust-region methods to mixed continuous and binary variables proposed in [25]. It will serve as the key building block when proposing our new algorithm in Chapter 5. After summarizing the algorithm, we focus on the theoretical proof of its local convergence, which was not explicitly provided in the original presentation [25].

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

$$d_H(\bar{y},\tilde{y}) = \sum_{j:\tilde{y}_j=0} \bar{y}_j + \sum_{j:\tilde{y}_j=1} (1-\bar{y}_j), \tag{18}$$

for $\bar{y},\tilde{y} \in \{0,1\}^n$. Roughly speaking, this distance simply computes the minimal number of flips (from 0 to 1, or from 1 to 0) required to transform $\bar{y}$ into $\tilde{y}$. Then, an exploration phase is added to help the optimization explore wider the binary domain. The convergence result we are about to present in this section is in fact driven totally by the exploitation phase, which solves a continuous

quadratic optimization subproblem by temporarily fixing the value of the discrete variables y, and by building *fully-linear* models of the continuous objective function $f(\cdot, y)$.

The outline of the chapter is as follows: Section 1 presents a mixed quadratic model construction based on the mixed interpolation set. Section 2 provides a review of the state-of-the-art initial design methods that are used in DFO. Sections 3 and 4 present the exploitation and exploration steps in DFOb-d_H respectively. We end up with the complete DFOb-d_H algorithm, some conclusions and discussion for this chapter in Section 4.

1 The quadratic model

This subsection details how the trust-region quadratic sub-problem model at iteration k, $\widetilde{m}_k$, is built. We consider the vector of optimization variables $z = (x, y)$, where $x \in \mathbb{R}^m, y \in \{0,1\}^n$. By adding binary variables, the notation of the interpolation set becomes

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

The classification of problems based on the value of p is unchanged, but with dimension $m+n$ (for mixed continuous and binary) instead of m (for continuous). We keep all the notations and definitions from Chapter 2, but with mixed binary $z = (x, y)$ variables.

Suppose that one is given a set of points $z^i = (x^i, y^i), x^i \in \mathbb{R}^m, y^i \in \{0,1\}^n, i = 0, 1, \ldots, p$, at which the objective function is evaluated with values $f^i := f(x^i, y^i), i = 0, 1, \ldots, p$, where $p > m + n$. The derivative-free trust-region algorithm for mixed binary variables is based on the local quadratic model

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

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

$$\begin{cases} \min\limits_{\alpha,g,H=H^T} \dfrac{1}{2}\|H\|_F^2 \\ \widetilde{m}_{\alpha,g,H}(z^i) = f^i, i = 0, 1, \ldots, p, \end{cases} \tag{19}$$

where $\|.\|_F$ is the Frobenius norm. From a computational perspective, subproblem (19) can be addressed by NLP solvers such as IPOPT (see details in [106]). For the sake of notational simplicity, in the sequel, the model $\widetilde{m}_{\alpha,g,H}$ will simply be denoted by $\widetilde{m}$.

The interpolation set needs to satisfy some conditions to ensure the uniqueness of the solution of the fitting problem (19). Let $d = 1$ or 2, and let $\{\phi_i\}_{i=0}^{h-1}$ be the natural basis of the space

of polynomials of degree $\leqslant d$ in $\mathbb{R}^{m+n}$ (h is then simply the dimension of this space). In our context where $y \in \{0,1\}^n$, and when $d = 2$, the ϕ_i elements of this basis are the components of the vector:

$$\begin{aligned}\phi(z) = (1, x_1, \ldots, x_m, y_1, \ldots, y_n, \frac{1}{2}x_1^2, \ldots, \frac{1}{2}x_m^2, \ldots, x_i x_j, \ldots, x_i y_j, \ldots, \\ x_m y_n, y_1 y_2, \ldots, y_i y_j, \ldots, y_{n-1} y_n),\end{aligned} \tag{20}$$

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

The proof of maintaining a $\Lambda-$poised sample set in a finite number of steps is evoked in [25]. We give the proof in the following lemma for the case of mixed continuous and binary variables, and $d = 1$ or 2. Remark that we only consider a proper mixed continuous and binary case, the pure continuous proof can be found in [27], and the pure binary case can be asset easily.

Lemma 4.1. *Let $v^T\phi(z)$ be a quadratic polynomial of degree at most d ($d = 1$ or 2), where $\phi(z)$ is defined above and $\|v\|_\infty = 1$. Then, there exists a constant $\sigma_\infty > 0$, independent of v, such that*

$$\max_{x \in B(0,1), y \in \mathbb{B}} |v^T\phi(z)| \geqslant \sigma_\infty. \tag{21}$$

For $d = 1, \sigma_\infty > 0$ and for $d = 2, \sigma_\infty \geqslant \frac{1}{4}$, where $B(x_k, \Delta_k)$ is the ball centered at x_k with radius Δ_k, and where $\mathbb{B}$ indicates the binary set $\{0,1\}^n$.

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

$$\psi(v) = \max_{x \in B(0,1), y \in \mathbb{B}} |v^T\phi(z)|. \tag{22}$$

We see that $\psi(v)$ is a norm in the space of vector v. Since the ratio of any two norms in finite dimensional spaces can be uniformly bounded by a constant, there exists a $\sigma_\infty > 0$ such that $\psi(v) \geqslant \sigma_\infty \|v\|_\infty$. The constant σ_∞ is defined as:

$$\sigma_\infty = \min_{\|v\|_\infty = 1} \psi(v). \tag{23}$$

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

We demonstrate for the two cases: for $d = 1$, there exists $\sigma_\infty > 0$ and $d = 2, \sigma_\infty \geqslant \frac{1}{4}$.
For $d = 1$, 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 $w = [v_{m+2}, v_{m+3}, \ldots, v_{m+n+1}]^T$. Remark that we consider the mixed continuous and binary case, it means $\|u\|_\infty \neq 0$ and $\|v\|_\infty \neq 0$. Therefore, $\psi(v)$ becomes $\max_{x \in B(0,1), y \in \mathbb{B}} |v_1 + u^T x + w^T y|$ which reaches an optimal value when $x = \frac{u}{\|u\|}$ and when y is chosen as

$$y_i = \begin{cases} 1, \text{if } w_i > 0, \\ 0, \text{otherwise}, \end{cases}$$

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

$$y_i = \begin{cases} 1, \text{if } w_i < 0, \\ 0, \text{otherwise}. \end{cases}$$

Thus, the optimal value of $\psi(v)$ is $|v_1| + \|u\| + \sum_{y_i = 1} |w_i| > 0$. This ends the proof of the first case.
For the case where $d = 2$: the natural polynomial basis of degree 2 is given as

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

Since $\|v\|_\infty = 1$, by the definition of $l_\infty-$norm, there exits $i : |v_i| = 1$. Thus, one of the coefficients of the polynomial $q(z) = v^T \phi(z)$ is equal to either 1 or -1 (corresponding to linear part in the basis $1, x_1, \ldots, x_m, y_1, \ldots, y_n$) or $\frac{1}{2}$, or $-\frac{1}{2}$ (corresponding to the quadratic part in the basis $\frac{1}{2}x_1^2, \ldots, \frac{1}{2}x_m^2, \ldots, x_i x_j, \ldots, x_i y_j, \ldots, x_m y_n$).
Let us consider only the cases where one of the coefficients of $q(z)$ is 1 or $\frac{1}{2}$ (the cases -1 or $-\frac{1}{2}$ are analyzed similarly).
The largest coefficient in absolute value in v corresponds to a term which is either a constant term, a linear term x_i or y_i, a quadratic term $\frac{1}{2}x_i^2$ or $x_i x_j$ or $x_i y_j$. 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 six 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 one has $|q(z)| \geqslant \frac{1}{4}$:

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

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

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

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

- $q(z) = y_i + \delta$: if $|\delta| \geqslant \frac{1}{4}$, then $|q(x, 0)| = |\delta| \geqslant \frac{1}{4}$. Otherwise, $\frac{-1}{4} < \delta < \frac{1}{4}$ and 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 $|q(1)| \geqslant \frac{1}{4}$ or $|q(-1)| \geqslant \frac{1}{4}$ holds. In both cases $|q(1)| < \frac{1}{4}$ and $|q(-1)| < \frac{1}{4}$, by adding these inequalities, we have $\frac{-1}{2} < 1 + 2\delta < \frac{1}{2}$ implies that $\delta < \frac{-1}{4}$, and also we have $q(0) = \delta < \frac{-1}{4}$. Thus, we have $|q(0)| > \frac{1}{4}$.

- $q(z) = x_i x_j + \frac{1}{2}\alpha x_i^2 + \frac{1}{2}\beta x_j^2 + \gamma x_i + +\delta x_j + \epsilon$: 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

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

As a result, we obtain $q(p_1) - q(p_2) = 1 + \sqrt{2}\delta$ and $q(p_3) - q(p_4) = -1 + \sqrt{2}\delta$. If $\delta \leqslant 0$, we have $q(p_1) - q(p_2) \leqslant 1$. If $|q(p_1)| < \frac{1}{2}$, then $|q(p_2)| \geqslant \frac{1}{2}$. Otherwise, $\delta > 0$, $q(p_3) - q(p_4) \geqslant -1$. Thus, if $|q(p_3)| < \frac{1}{2}$, then $q(p_4) \leqslant \frac{-1}{2}$.

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

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

From the last two equations, we have

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

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

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

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

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

In this case, if $|q(-1,0)| < \frac{1}{2}$, then $q(1,0) < \frac{-1}{2}$, which implies that $|q(1,0)| > \frac{1}{2}$. Otherwise, if $|q(-1,0)| \geqslant \frac{1}{2}$, the result is trivial.

□

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

$$M := \begin{pmatrix} \phi_0(z^0) & \phi_1(z^0) & \dots \phi_h(z^0) \\ \phi_0(z^1) & \phi_1(z^1) & \dots \phi_h(z^1) \\ \vdots & \vdots & \vdots \\ \phi_0(z^p) & \phi_1(z^p) & \dots \phi_h(z^p) \end{pmatrix}. \tag{24}$$

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

- $h = p + 1$ (*determined case*): Following [27], the interpolation set Z is said to be *poised* if the determinant of M is non-zero.

- $p+1 < h$ (*under-determined case*): Again, as in [27], Z is *poised* if M is full column rank ($\text{rank}(M) = \min(p, h) = p$).

- $h < p+1$ (*over-determined case*): In this case we propose to remove $p-h$ points from the interpolation set (we shall define precisely in the algorithm which points should be eliminated), so that one falls into one of the two previous cases.

A so-called *ill-geometry* situation leading to a non-poised interpolation set occurs when for instance at some iteration, two or more interpolation points collapse or are affinely dependent. This results in non-uniqueness of solutions of the fitting problem (19). There is also an ill-geometry problem in the case of a near-singular interpolation matrix (when two interpolation points are too close to each other for example). To prevent this scenario, an improvement step based on LU factorization is set up in the mixed space $\mathbb{R}^m \times \{0,1\}^n$, inspired from the continuous version of Section 2.2.4 from [27], and detailed in Algorithm 3. It involves solving a mixed-integer quadratic programming problem (MIQP) to be defined below.

Algorithm 3 provides a poised interpolation set such that when Gaussian elimination is applied to the interpolation matrix M, the absolute value of all pivots are not smaller than the chosen threshold ξ.

Algorithm 3: Improving poisedness of Z in the mixed trust region B, extended from [27]

0. Initialization

Choose an initial pivot polynomial basis with some basis $u_i(z), i = 0, 1, \ldots, h$, *e.g.*, the monomial basis $\phi(z)$ given by (20).

Select a pivot threshold $\xi > 0$.

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

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

$$z^i \in \operatorname{argmax}_{z \in B} |u_i(z)|, \tag{25}$$

   where B is the trust region we are considering.

   Stop if $|u_i(z^i)| < \xi$.

2. **Gaussian elimination:** For $j = i+1, i+2, \ldots, p$

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

From a computational perspective, subproblem (25) can be addressed by a MIQP solver such as CPLEX or BONMIN.

The introduction of binary variables requires an adapted trust-region definition. In the presentation [25], the authors introduce an $\|.\|_\infty$ (l_∞-norm) trust region for the continuous variables, and a Hamming-distance trust region for the binary variables.

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

$$B(x_0, \Delta_x) \times \mathcal{B}(y_0, \Delta_y), \tag{26}$$

where

$$B(x_0, \Delta_x) = \{x \in \mathbb{R}^m : \|x - x_0\|_\infty \leqslant \Delta_x\}, \tag{27}$$

and

$$\mathcal{B}(y_0, \Delta_y) = \{y \in \{0,1\}^n : d_H(y, y_0) \leqslant \Delta_y\}, \tag{28}$$

for some given trust-region radii Δ_x and Δ_y.

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

2 Initial interpolation set

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

In order to construct a first quadratic model, one requires an interpolation set that contains a sufficient number of points together with the corresponding objective function values. As indicated in [27], for DFOb-d_H this number is often taken equal to $m + n + 1$. Further, these points need to satisfy strict geometry conditions for the interpolation problem to be well posed. As remarked in [105], a "good" design of experiments (DoE) not only needs to be *affinely independent*, but should additionally satisfy *space-filling, non-collapsing* properties [41].

There are several methods to choose a given number of sample points in a continuous space, such as factorial designs, Latin Hypercube Sample (LHS), and Optimal LHS designs (see *e.g.* [105]). However, here we deal with mixed continuous and binary variables problems: we need to provide a DoE in the mixed space $\mathbb{R}^m \times \{0, 1\}^n$. When the dimension is small, one way to proceed is to sample among 2^{n+m} corner points of the boundary box: for example [48] proposes a strategy that chooses $m + n + 1$ corner points plus the central point of the box. For larger dimensions, a popular strategy is the Latin Hypercube Sample (LHS) [66, 109], originally used for generating samples for continuous variables in a bounded subset. However, points sampled by this strategy will surely not satisfy our binary constraints.

In our implementation, we therefore proceed as in [30] for the RBFOpt algorithm: we first construct a Latin Hypercube Design with maximin distance criterion of $m + n + 1$ points in the considered bounded subset of $\mathbb{R}^{m+n}$, then we round the n components associated with the binary variables to zero or one. Remark that rounding recovers the binary domain but may destroy the desirable properties of LHS or, even worse, it may generate identical points. A future track of our research should therefore be dedicated to improving the method for generating the initial design of experiments.

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

3 Exploitation phase

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

$$\begin{cases} \min\limits_{x} \widetilde{m}_k(x, y_k) \\ \text{s.t.} \quad \|x - x_k\|_\infty \leqslant \Delta_{x,k}, \end{cases} \tag{29}$$

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

The following step, Step 1b, tests whether the solution, x^*, of (29) should be accepted, based on the ratio, ρ, of the true improvement in f brought by x^*, over the improvement predicted by the model:

$$\rho = \frac{f(x_k, y_k) - f(x^*, y_k)}{\widetilde{m}_k(x_k, y_k) - \widetilde{m}_k(x^*, y_k)}, \tag{30}$$

where one remarks that the denominator is always negative since x^* is solution of (29). We introduce η_{good}, η_{ok}, and η_{tol}, some pre-defined acceptance threshold values such that $\eta_{good} > \eta_{ok} > \eta_{tol} > 0$.

If $\rho \geqslant \eta_{tol}$, the new iterate is accepted (*successful iteration*), If $\rho < \eta_{tol}$, the solution is rejected (*unsuccessful iteration*).

This exploitation phase (referred to as Step 1 in the sequel) is summarized in Algorithm 4.

Algorithm 4: Exploitation phase of DFOb-d_H (Step 1) **at iteration** k

---

Step 1a (TR QP)

- Solve (29) **for fixed** $\boldsymbol{y} = y_k$ in $B_{y_k}(x_k, \Delta_{x,k})$ to get x^*
- Evaluate $f(x^*, y_k)$; if $n_{simu} = \overline{n_{simu}} \rightarrow$ **STOP**
- Add $((x^*, y_k), f(x^*, y_k))$ to Z_k; $p \leftarrow p + 1$

Step 1b (Validation)
Compute the acceptance ratio ρ via (30)

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

---

4 Exploration phase

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

$$\begin{cases} \min\limits_{x\in\mathbb{R}^m, y\in\{0,1\}^n} \widetilde{m}_k(x,y) \\ \text{s.t.} \quad \|x - x_k\|_\infty \leqslant \Delta_{x,k}, \\ \qquad d_H(y, y_k) \leqslant \Delta_{y,k}. \end{cases} \tag{31}$$

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

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

In case of an unsuccessful Step 1.5a (*i.e.*, no improvement in the minimization of f or failure in solving (31)), we continue with the same y_k, with a trust-region management with respect to the x component, and a new Step 1 to improve the current solution with y fixed to the value y_k:

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

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

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

If the solution of (31) does not yield improvement with respect to the current center (x^*, y^*), and the pre-defined minimal value of the trust-region size, $\underline{\Delta_x}$, is reached, then (x^*, y^*) is considered to be a locally-optimal solution. In this case, the algorithm explores in a Step 3 the binary search

space using *no-good cuts*, analogous to those introduced in [32] for general mixed optimization problems. The aim of no-good cuts were to remove a point that was integer, but not feasible for all the constraints which we can informally call the "no-good" solutions. This leads in our case to relaxing the trust-region constraint:

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

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

$$\sum_{j:y_j^*=0} y_j + \sum_{j:y_j^*=1} (1 - y_j) \geqslant K^*, \tag{32}$$

where $K^* \in \mathbb{N}^*$ is some user-defined discrepancy value strictly greater than 1. Note that for a given x_k, several such no-good cut constraints are likely to cumulate, as there will be one constraint of the form (32) corresponding to each of the different y^* values obtained. The set of no-good cut constraints at iteration k is denoted by Ω_k^{NGC}. If the exploration does not provide us a better solution, an *add-points* step is added. In this particular step, we allow to add an user-defined number of points generated by DoEs method. The exploration step (Step 1.5) is summarized in Algorithm 5.

Algorithm 5: Exploration phase of DFOb-d_H (Step 1.5) **at iteration** k

---

Step 1.5a (MIQP subproblem)

- $is_new_NGC = 0$
- Solve MIQP (31) in $B(x_k, \Delta_{x,k}) \times (\mathcal{B}(y_k, \Delta_{y,k}) \cap \Omega_k^{NGC})$ to get (x^*, y^*)
- Evaluate $f(x^*, y^*)$ if $n_{simu} = \overline{n_{simu}} \rightarrow$ **STOP**
- Add $((x^*, y^*), f(x^*, y^*))$ to Z_k; $p \leftarrow p + 1$

Step 1.5b (Validation)

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

  else (unsuccessful step 1.5)

  If $y^* \neq y_k$: $\Delta_{y,k} \leftarrow \Delta_{y,k} - 1$

The algorithm finally ends when the maximal budget of objective-function evaluations or the maximal number of no-good cuts is reached. The maximal number of possible no-good cuts is theoretically equal to $2^n - 1$ (for n binary variables). We shall see later that in the context of

cyclic symmetry, it is in fact approximately $2^n/n$. But more importantly, with this type of property we will see that the definition of no-good cuts in (32) is not sufficient to discriminate equivalent configurations.

The complete DFOb-d_H algorithm is outlined in Algorithm 6.

Algorithm 6: DFOb-d_H algorithm

Initialization

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

Iteration k:

Step 0 (Model update and improvement)

- Build quadratic model $\widetilde{m}_k(x, y)$ (cf. Section 1)
- Improve poisedness of Z_k by solving a TR MIQP in $B(x_k, \Delta_{x,k}) \times \mathcal{B}(y_k, \Delta_{y,k})$ (Algorithm 3)[a]
- **If** $n_{simu} = \overline{n_{simu}} \rightarrow$ **STOP**

if $is_new_NGC = 1$: go to **Step 1.5**

Step 1 (Exploitation phase): Algorithm 4 (cf. Section 3)

- **If** unsuccessful Step 1: **go to Step 2**

Step 1.5 (Exploration phase): Algorithm 5 (cf. Section 4)

- **If** successful Step 1.5: $\boldsymbol{k \leftarrow k+1}$ and **go to Step 0**

Step 2 (TR update and local convergence check)

- **If** $\rho \leqslant \eta_{ok}$: $\Delta_{x,k} \leftarrow \Delta_{x,k}/2$
- **If** $\rho \geqslant \eta_{good}$: $\Delta_{x,k} \leftarrow 2\Delta_{x,k}$
- **If** $\Delta_{x,k} > \underline{\Delta_x}$: $\boldsymbol{k \leftarrow k+1}$ and go to **Step 0**

Step 3 (Exploration after local convergence)

- Adding a new no-good cut : $\Omega_k^{NGC} \leftarrow \Omega_k^{NGC} \cap \{y \in \{0,1\}^n : d_H(y, y_k) \geqslant K^*\}$, $\boldsymbol{n_{NGC} \leftarrow n_{NGC} + 1}$; $\boldsymbol{is_new_NGC = 1}$
- Reinitialize TR radii: $\Delta_{x,k} \leftarrow \Delta_{x,0}$, $\Delta_{y,k} \leftarrow \overline{\Delta_y}$
- **If** $n_{NGC} \leqslant \overline{n_{NGC}}$: $\boldsymbol{k \leftarrow k+1}$ and go to **Step 0**

  else: $\rightarrow$ **STOP**

[a] Algorithm 3 adds possibly in Z_k new points to be simulated.

5 Local convergence results

In order to avoid ill-conditioning and ensure the local convergence of the algorithm, we rely on a class of so-called *fully-linear* models [27] within the chosen trust region. As shown in [27] for continuous problems, if an interpolation set is poised, then the model obtained by solving the minimal Frobenius fitting problem is fully linear in the trust region of size $\Delta_x = \max_{i=1,2,\dots,p}(\|x^i - x_0\|_\infty)$ defined by the interpolation points, which ensures a control of the model error by controlling the size of the trust region and the interpolation set poisedness with the model improvement step.

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

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

Note that, in our implementation, the model improvement step (Algorithm 3) is performed in $B(x_0, \Delta_x) \times \{0,1\}^n$, where $B(x_0, \Delta_x)$ is defined by (27). This allows a larger exploration with respect to binary variables than an improvement step in the mixed trust region $B(x_0, \Delta_x) \times B(y_0, \Delta_y)$ while still fulfilling the required assumptions for Lemma 4.2 below. In the following, we give the proof of fully-linear models for fixed $y = y_0$.

Assumption 4.1.

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

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

Assumption 4.2.

At every iteration k of the Algorithm (6), the Frobenius norm of the model Hessian evaluated at iterate (x_k, y_k), H_k, is bounded.

Lemma 4.2. *Let (x_0, y_0) be the initial iterate. Under Assumption 4.1 and Assumption 4.2, the model $\widetilde{m}(\cdot, y_0)$ which is constructed from $\widetilde{m}(x, y)$ by fixing $y = y_0$ is **fully linear** in $B_{y_0}(x_0, \Delta_x)$. In other words, for all $x \in B_{y_0}(x_0, \Delta_x)$, there exist $\kappa_f^*, \kappa_g^* > 0$ such that:*

$$|f(x, y_0) - \widetilde{m}(x, y_0)| \leqslant \kappa_f^* \Delta_x^2, \tag{33}$$

and

$$\|\nabla_x f(x, y_0) - \nabla_x \widetilde{m}(x, y_0)\|_2 \leqslant \kappa_g^* \Delta_x. \tag{34}$$

Proof. The model constructed in mixed space is given as:

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

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

$$\widetilde{m}(x, y_0) = \bar{c}_x + \bar{g}_x x + \frac{1}{2} x^T \bar{H}_x x, \tag{35}$$

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

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

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

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

We define

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

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

$$\begin{aligned}(x^i - x)^T err_0^g(x) &= (x^i - x)^T(\bar{H}_x x + \bar{g}_x - \nabla f_0(x)) \\ &= (x^i - x)^T \bar{H}_x x + (x^i - x)^T \bar{g}_x - f_0(x^i) + f_0(x) \\ &+ [f_0(x^i) - f_0(x) - (x^i - x)^T \nabla f_0(x)] \\ &= m_0(x^i) - m_0(x) - \frac{1}{2}(x^i - x)^T \bar{H}_x (x^i - x) - f_0(x^i) + f_0(x) \\ &+ [f_0(x^i) - f_0(x) - (x^i - x)^T \nabla f_0(x)] \\ &= err_0^f(x^i) - err_0^f(x) - \frac{1}{2}(x^i - x)^T \bar{H}_x (x^i - x) \\ &+ [f_0(x^i) - f_0(x) - (x^i - x)^T \nabla f_0(x)].\end{aligned} \tag{36}$$

Since f_0 is continuously differentiable, we have:

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

which implies that

$$\begin{aligned}(x^i - x)^T err_0^g(x) = &\int_0^1 (x^i - x)^T (\nabla f_0(x + t(x^i - x)) - \nabla f_0(x)) dt \\ &+ err_0^f(x^i) - err_0^f(x) - \frac{1}{2}(x^i - x)^T \bar{H}_x (x^i - x).\end{aligned} \tag{37}$$

Using (37) with $x = x_0$, we obtain:

$$\begin{aligned}(x^i - x_0)^T err_0^g(x) &= (x^i - x)^T err_0^g(x) - (x_0 - x)^T err_0^g(x) \\ &= \int_0^1 (x^i - x)^T (\nabla f_0(x + t(x^i - x)) - \nabla f_0(x)) dt + err_0^f(x^i) \\ &- \frac{1}{2}(x^i - x)^T \bar{H}_x (x^i - x) \\ &- \int_0^1 (x_0 - x)^T (\nabla f_0(x + t(x_0 - x)) - \nabla f(x)) dt - err_0^f(x_0) \\ &+ \frac{1}{2}(x_0 - x)^T \bar{H}_x (x_0 - x).\end{aligned} \tag{38}$$

First, note that $err_0^f(x_0) = 0$. Then, for each terms of (38), we obtain the following upper bounds:

- From the Lipschitz property of $f_0(x)$ (Assumption 4.1), one has:

$$\begin{aligned} \left| \int_0^1 (x^i - x)^T (\nabla f(x + t(x^i - x)) - \nabla f(x)) dt \right| &\leqslant \frac{1}{2}\nu \|x^i - x\|^2 \\ &\leqslant \frac{1}{2}\nu (2\Delta_x)^2 \\ &\leqslant 2\nu\Delta_x^2. \end{aligned} \tag{39}$$

- In the same way, we have:

$$\left| \int_0^1 (x_0 - x)^T (\nabla f(x + t(x_0 - x)) - \nabla f(x)) dt \right| \leqslant \frac{1}{2}\nu \|x_0 - x\|^2 \leqslant \frac{1}{2}\nu\Delta_x^2. \tag{40}$$

- In the following two inequalities, note that $\|\bar{H}_x\|_F$ is bounded from Assumption 4.2:

$$\left| \frac{1}{2}(x^i - x)^T \bar{H}_x (x^i - x) \right| \leqslant \frac{1}{2}\|\bar{H}_x\|_F \|x^i - x\|^2 \leqslant \frac{1}{2}\|\bar{H}_x\|_F (2\Delta_x)^2 \leqslant 2\|\bar{H}_x\|_F \Delta_x^2. \tag{41}$$

$$\left| \frac{1}{2}(x_0 - x)^T \bar{H}_x (x_0 - x) \right| \leqslant \frac{1}{2}\|\bar{H}_x\|_F \|x_0 - x\|^2 \leqslant \frac{1}{2}\|\bar{H}_x\|_F \Delta_x^2. \tag{42}$$

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

$$|err_0^f(x^i)| \leqslant \epsilon' \Delta_x^2, \tag{43}$$

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

$$|err_0^f(x^i)| > \epsilon' \Delta_x^2 \quad \forall \epsilon' > 0. \tag{44}$$

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

$$\begin{aligned} |err_0^f(x^i)| &= |f_0(x^i) - \widetilde{m}_0(y_0)| \\ &= |f_0(x^i) - f_0(x_0) + \widetilde{m}_0(x_0) - \widetilde{m}_0(x^i)| \\ &\leqslant |f_0(x^i) - f_0(x_0)| + |\widetilde{m}_0(x_0) - \widetilde{m}_0(x^i)| \\ &\leqslant (\epsilon_1 + \epsilon_2)\Delta_x. \end{aligned} \tag{45}$$

  Thus, setting $\epsilon' = \dfrac{\epsilon_1 + \epsilon_2}{\Delta_{x,min}} \geqslant \dfrac{\epsilon_1 + \epsilon_2}{\Delta_x}$ in (44) contradicts (45).

Thus, we find from (38) and the inequalities (39 – 43):

$$|(x^i - x_0)^T err_0^g(x)| \leqslant \frac{5}{2}\Delta_x^2\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big), \tag{46}$$

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

$$\|err_0^g(x)\|_2 \leqslant \frac{5}{2}\Delta_x\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big). \tag{47}$$

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

We recall that the interpolation set Z is defined as

$$Z = \begin{pmatrix} x_0 & y_0 \\ x^1 & y^1 \\ \vdots & \vdots \\ x^p & y^p \end{pmatrix}. \tag{48}$$

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

We have

$$X^T err_0^g(x) = \frac{1}{\Delta_x}\begin{bmatrix} (x^1 - x_0)^T \\ \vdots \\ (x^p - x_0)^T \end{bmatrix} err_0^g(x). \tag{49}$$

Then, we obtain from inequality (46):

$$\begin{aligned} \|X^T err_0^g(x)\|_\infty &= \frac{1}{\Delta_x}\max_{i=1,\ldots,p}|(x^i - x_0)^T err_0^g(x)| \\ &\leqslant \frac{1}{\Delta_x}\left[\frac{5}{2}\Delta_x^2\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big)\right] \\ &= \Delta_x\left[\frac{5}{2}\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big)\right]. \end{aligned} \tag{50}$$

Moreover, we have:

$$\begin{aligned} \|err_0^g(x)\|_2 &= \|X^{-T}X^T err_0^g(x)\|_2 \\ &\leqslant \|X^{-T}\|_2 \, \|X^T err_0^g(x)\|_2. \end{aligned} \tag{51}$$

Thus, we obtain:

$$\|err_0^g(x)\|_2 \leqslant \sqrt{m}\|X^{-T}\|_2 \ \|X^T err_0^g(x)\|_\infty \leqslant \sqrt{m}\|X^{-T}\|_2 \ \Delta_x \left[\frac{5}{2}\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big)\right]. \tag{52}$$

Recovering $err_0^f(x)$ by equation (37), we have:

$$\begin{aligned} |err_0^f(x)| &\leqslant \|err_0^g(x)\|\Delta_x + 2\nu\Delta_x^2 + 2\|\bar{H}_x\|_F\Delta_x^2 + |err_0^f(x^i)| \\ &\leqslant \left[\sqrt{m}\|X^{-T}\|_2\frac{5}{2}\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big) + 2(\nu + \|\bar{H}_x\|_F) + \epsilon'\right]\Delta_x^2. \end{aligned} \tag{53}$$

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

$$\kappa_g^* = \frac{5}{2}\sqrt{m}\|X^{-T}\|_2\Big(\nu + \|\bar{H}_x\|_F + \epsilon\Big), \tag{54}$$

and

$$\kappa_f^* = \Big(\nu + \|\bar{H}_x\|_F)\Big)\Big(\frac{5}{2}\sqrt{m}\|X^{-T}\|_2 + 2\Big) + \frac{5}{2}\epsilon\Big(\sqrt{m}\|X^{-T}\|_2 + 1\Big). \tag{55}$$

□

We can now state the local convergence of the algorithm.

Theorem 4.1. *Let Assumptions 4.1 and 4.2 hold. Then,*

$$\lim_{k\to\infty} \nabla_x f(x_k, y_{k-1}) = 0, \tag{56}$$

or all the limit points of the sequence of iterates in Algorithm 6 are first-order critical points.

The proof is obtained by following the same process as in [27] (Theorem 10.13): from the results of Lemma 4.2 above, we can prove the local convergence (convergence for fixed y) of the algorithm with the additional assumption that f is bounded from below for all $(x, y) \in \Omega$, a closed subset of $\mathbb{R}^m \times \{0, 1\}^n$.

REMARK 4.1. As explained in [28], an interpolation point outside $B_{y_0}(x_0, \Delta_x)$ has to be replaced in order to ensure a fully-linear model. However, in practice, in order to save expensive objective-function evaluations, we allow to go on with a model that is not certified to be fully linear when it yields effective progress in the minimization of the function.

6 Conclusion

This chapter provides the main ingredient of DFOb-d_H, a derivative-free optimization method for mixed continuous and binary variables problems. Unfortunately, based on the fact that the Hamming distance cannot detect the equivalent configurations in the binary part of the variables, the method is not satisfying to address cyclic-symmetry problems. In the next chapter, we give our efforts to adapt DFOb-d_H to this type of problem.

5

DFOb-d_{neck}: A DFOb method for cyclic-symmetry problems

Contents

1 Necklace context **52**
1.1 Concept of “necklace" 52
1.2 Survey of necklace distances 53
2 An adapted distance for cyclic-symmetry problems **56**
2.1 The necklace distance 56
2.2 Reformulation of the QP subproblems involving the necklace distance . 60
2.3 Algorithm DFOb-d_{neck} 65
3 Numerical results **66**
3.1 New mixed binary optimization test problems featuring cyclic symmetry 69
3.2 Results obtained on the 25 analytical problems 71
3.3 Design of compressor blades in a helicopter turbomachine 74
4 Conclusion **81**

We begin the chapter by recalling our target application: the optimal design of the blade shape of a turbine for aircraft's engine in order to minimize the vibrations. 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.

The reduced order model (ROM) is commonly used for addressing this problem. However, ROM removes a large number of possibilities for optimization, in the sense that ROM considers two reduced subproblems but the cyclic symmetry property remains and is not taken into account explicitly in the standard optimization workflow. For instance, in the case of $n = 12$ blades, ROM considers two subproblems 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, and the removed configurations could be "useful" candidates for the optimization.

Therefore, we define a new distance (to be used within the DFOb algorithm) to model the cyclic symmetry and 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 [42, 43]. In addition, the adaptation of the optimization method to this new distance is discussed.

1 Necklace context

To avoid useless costly evaluations of the numerous equivalent solutions for cyclic-symmetry problems, engineers typically resort to simplifications or adapted strategies (such as the reduced-order model methodology [23, 24]) to reduce the optimization problem dimension. However, such simplifications are likely to discard interesting or even optimal configurations.

1.1 Concept of “necklace"

In order to avoid re-evaluating costly objective-function evaluations at equivalent blade arrangements, we propose to use the concept of *necklace* [42, 43].

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**) Let k and n be strictly positive integers. 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 with 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 $\frac{2^n}{n}$ as mentioned in the turbine application [49, 74]. By using the result from [42], we obtain that 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 numerous applications of the concept of necklace, many of them are based on necklace distances such as similarity of different types of music [98, 99, 100], the dissimilarity of DNA in biology [4], calculating the leap years and design calendars [34], painting car in manufacturing [68]. In the the next section, we review some usual necklace distances.

1.2 Survey of necklace distances

The study of distances in continuous space is well known and widely studied 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 a challenge of our problem as it can save a large number of simulations in the optimization procedure. There exists several 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

$$d_H(y, y') = \sum_{i=1}^{n} |y_i - y'_i|. \tag{57}$$

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

$$d_V(y, y') = \max(N_{01}(y, y'), N_{10}(y, y')), \tag{58}$$

where $N_{01}(y, y') = \#\{(y_i, y'_i) : y_i = 0, y'_i = 1\}$ and $N_{10}(y, y') = \#\{(y_i, y'_i) : y_i = 1, y'_i = 0\}$, where $\#$ denotes the cardinality of the set.

One has:

$$N_{01}(y, y') = \sum_{i=1}^{n}(1 - y_i)y'_i = \sum_{i=1}^{n} y'_i - \sum_{i=1}^{n} y_i y'_i,$$

and

$$N_{10}(y, y') = \sum_{i=1}^{n}(1 - y'_i)y_i = \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} y'_i y_i.$$

Therefore, we have:

$$d_V(y, y') = \max\left(\|y\|_1, \|y'\|_1\right) - y^T y'.$$

For more details, see [29]. Varshamov distance can detect cases where the number of 1s in string y is larger than the number of 1s in string y'.

Definition 5.4. (**The swap distance**) y and y' are as above. Assume that y and y' have the same number of 1, $U = (u_1, \ldots, u_k), V = (v_1, \ldots, v_k)$ 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

$$d_{swap}(y, y') = d_{swap}(U, V) = \sum_{i=1}^{k} |u_i - v_i|. \tag{59}$$

In [5] the authors use the swap distance to measure the similarity of two rhythms (application in classification of types of music) or in bioinformatics where the two strings to be compared are chain polymers. Computing U, V from y, y' costs $\mathcal{O}(n)$ operations and Toussaint [101] states that the swap distance can be computed in $\mathcal{O}(n^2)$. However, in [5], the authors introduce algorithms computing the swap distance in $\mathcal{O}(k^2)$ and $\mathcal{O}(k^3)$ operations.

Definition 5.5. (**The Hamming distance with shifts**) Given y, y' two binary strings, 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 [70]. It measures not only the number of mismatches but also how far apart the mismatches occur.

We note that with the values of $c_{ins} = c_{del} = 1, c_{sh} = \infty$, the Hamming distance with shifts reduces to the Hamming distance, and when $c_{ins} = c_{del} = \infty, c_{sh} = 1$, it reduces to the swap distance.

In [69] the author 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 1s values (taking into account the rotations). 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 1s is given by

$$d_E(y, y') = \sqrt{\sum_{i=1}^{k} (u'_i - v'_i)^2}. \tag{60}$$

The interval vectors can be computed from the given binary sequences in $\mathcal{O}(n)$ operations, and $d_E(y, y')$ can be computed in $\mathcal{O}(k)$ operations, where k is the number of 1s 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 $Z^1 = (z^1_1, \ldots, z^1_k)$, with $z^1_i = v'_{i+1}/v'_i, i = 1, \ldots, k-1, z^1_k = y'_k/v'_1$. The **interval difference vector** distance is

$$d_{ID}(y, y') = \Big(\sum_{i=1}^{k} \frac{\max(z_i, z^1_i)}{\min(z_i, z^1_i)} \Big) - k. \tag{61}$$

In [98] $d_{ID}(y, y')$ is computed in $\mathcal{O}(n)$ operations.

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

$$d_{MDPA}(y, y') = \min_{Y,Y'} \Big(\sum_{i,j=1}^{n} |y_i - y'_j| \Big). \tag{62}$$

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* [101, 98]. We listed above the main formulations to see how the cyclic symmetry could possibly be taken into account. In the next section, we present a new necklace distance tailored to our application.

2 An adapted distance for cyclic-symmetry problems

To avoid useless costly evaluations of the numerous equivalent solutions for cyclic-symmetry problems (as illustrated in Table 3), scientists and engineers typically resort to simplifications or adapted strategies (such as the reduced-order model methodology [23, 24]) to reduce the optimization problem dimension. However, such simplifications are likely to discard interesting or optimal configurations.

This section first defines a new distance to be used in algorithm DFOb so as to avoid to evaluate the costly blackbox objective function at configurations that were previously evaluated (equivalent solutions), without arbitrary removal of (potentially good) candidate configurations. The new distance should lead to constraints easily manageable in efficient optimization methods, just like the Hamming distance which leads to linear constraints (cf. (18) and (32)). Then, in Subsection 2.2 we propose a reformulation of the algorithm optimization subproblems with this new distance for both the exploitation and the exploration phases. Subsection 2.3 summarizes the adapted algorithm, named DFOb-d_{neck}, and provides the local convergence statement.

2.1 The necklace distance

Our blade design application can be seen as a 2-color (or *binary*) necklace optimization problem involving a fixed number, n, of beads (the number of reference blade shapes). The number of distinct arrangements in our applicative context is therefore given by the number of n-bead

necklaces: $\frac{1}{n}\sum_{d|n}\phi(d)2^{n/d}$, where ϕ is Euler's totient function (the number of positive integers between 1 and n that are relatively prime to n) and the summation is taken over all divisors d of n.

The new distance we shall use is inspired from the particular l_p *necklace alignment distance* (denoted l_p NAD) where $p = 1$ [19]. Given two vectors of n real numbers, $v = (v_1, v_2, \ldots, v_n), v' = (v'_1, v'_2, \ldots, v'_n), v_i, v'_i \in [0, 1)$, the l_p NAD is defined as:

$$d_{l_pNAD}(v, v') = \min_{c,s} \sum_{i=1}^{n} (d^0((v_i + c) \bmod 1, v'_{(i+s) \bmod n}))^p, \tag{63}$$

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

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

(see [19] for more detail).

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

Definition 5.9. Given two k-ary necklaces of length n: $u = (u_1, u_2, \ldots, u_n)$ and $u' = (u'_1, u'_2, \ldots, u'_n)$, where $u_i, u'_i \in \{a_1, a_2, \ldots, a_k\}$, $i = 1, 2, \ldots, n$, the **discrete necklace distance** between u and u' is:

$$d^*_{neck}(u, u') = \min_{s=1,2,\ldots,n} \sum_{i=1}^{n} |u_i - u'_{i+s}|. \tag{64}$$

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

Definition 5.10. Given $y, y' \in \{0, 1\}^n$, the **binary necklace distance** between y and y' is:

$$d_{neck}(y, y') = \min_{i=1,2,\ldots,n} d_H(y, Rot^i(y')), \tag{65}$$

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

For example, considering two binary strings: $y = (0, 1, 0)$ and $y' = (1, 1, 0)$. One has: $d_H(y, y') = 1, d_H(y, Rot^1(y') = d_H(y, (0, 1, 1)) = 1, d_H(y, Rot^2(y')) = d_H(y, (1, 0, 1)) = 2$. Using Definition (65): $d_{neck}(y, y') = 1$.

Metric properties of the necklace distance d_{neck}.

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

Proposition 5.1. ***(Non-negativity property)*** *d_{neck} has the non-negativity property*

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

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

Proposition 5.2. ***(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 5.3. ***(Commutativity property)*** *d_{neck} has the commutativity property*

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

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

Proposition 5.4. ***(Triangle inequality property)*** *d_{neck} satisfies the triangle inequality property:*

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

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

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

Definition 5.11. (**Cyclic symmetry property in distance**) We say that a distance d has the **cyclic symmetry property** if

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

for any given two distinct vectors y, y'.

Proposition 5.5. ***(Cyclic symmetry property)*** *d_{neck} has the cyclic symmetry property. In other words*

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

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

1. Proof of the first implication " $\implies$ ": "$d_{neck}(y, y') = 0 \implies y \in Rot(y')$
   Assume $d_{neck}(y, y') = 0$ then $\min_r(d_H(y, Rot^r(y'))) = 0$, which implies that there exists $i \in \{1, 2, \ldots, n\}$ such that: $d_H(y, Rot^i(x)) = 0$, by the metric property of Hamming distance $d_H(y, Rot^i(y')) = 0 \iff y = Rot^i(y')$.

2. Proof of the second implication " $\impliedby$ ": $y \in Rot(y') \implies d_{neck}(y, y') = 0$.
   If $y \in Rot(y')$ then there exists $i \in \{1, 2, \ldots, n\}$ such that $y = Rot^i(y')$, which implies $d_H(y, Rot^i(y')) = 0$. Then, $\min_r d_h(y, Rot^r(y')) = 0$, hence $d_{neck}(y, y') = 0$.

□

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

Unfortunately, contrary to the Hamming distance (see Equations (18) or (32) for instance), a constraint involving the binary necklace distance cannot be straightforwardly expressed as linear constraints (due to the "min" operator involved in the definition of d_{neck}). The next section proposes a way to address this critical issue for adapting Algorithm 6 to the new distance d_{neck} (which will replace the Hamming distance) so as to obtain an algorithm, that we shall name DFOb-d_{neck}, that deals only with linear constraints.

2.2 Reformulation of the QP subproblems involving the necklace distance

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

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

$$\min_{i=1,2,\ldots,n} \{a_i\} \geqslant K^* \Longleftrightarrow a_i \geqslant K^*, i = 1, 2, \ldots, n. \tag{66}$$

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

$$\min_{i=1,2,\ldots,n} \{d_H(y, Rot^i(y_0))\} \geqslant K^* \Longleftrightarrow d_H(y, Rot^i(y_0)) \geqslant K^*, i = 1, 2, \ldots, n, \tag{67}$$

or

$$d_{neck}(y, y_0) \geqslant K^* \Longleftrightarrow d_H(y, Rot^i(y_0)) \geqslant K^*, i = 1, 2, \ldots, n. \tag{68}$$

To summarize, one can formulate a no-good cut that avoids useless costly evaluations by using n linear constraints since (68) involves n Hamming-distance inequalities, each of which can be written under the form of a linear inequality following (18).

Necklace-distance based trust regions. The way we replace Hamming distances by binary necklace distances in the exploration phase (more precisely, in the trust-region mixed binary quadratic sub-problem (31) of Step 1.5 of Algorithm 6) is less straightforward.

We consider the mixed binary optimization problem:

$$\begin{cases} \min\limits_{x,y} \widetilde{m}_k(x, y) \\ \text{s.t. } \|x - x_k\|_\infty \leqslant \Delta_{x,k}, \\ \quad\ d_{neck}(y, y_k) \leqslant \Delta_{y,k}, \\ \quad\ y \in \{0,1\}^n, \end{cases} \tag{69}$$

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

We propose to replace (69) by the following perturbed problem which involves an auxiliary variable t:

$$\begin{cases} \min\limits_{x,y,t} & \widetilde{m}_k(x,y) + \mu t \\ \text{s.t.} & \|x - x_k\|_\infty \leqslant \Delta_{x,k}, \\ & t = \min\limits_{i=1,2,\ldots,n} d_H(y, Rot^i(y_k)), \\ & t \leqslant \Delta_{y,k}, \\ & y \in \{0,1\}^n, \end{cases} \tag{70}$$

where $\mu > 0$ is a weighting parameter. We shall see in Subsection 2.3 that setting μ to a small-enough value conserves the fully-linear property of the perturbed model, thus ensuring the local convergence (Lemma 5.1 in next section) of the algorithm DFOb-d_{neck} to be presented.

Consider now the related mixed binary quadratic problem:

$$\begin{cases} \min\limits_{x,y,\tilde{y},t} & \widetilde{m}_k(x,y) + \mu t \\ \text{s.t.} & \|x - x_k\|_\infty \leqslant \Delta_{x,k}, \\ & t \geqslant d_H(y, Rot^i(y_k)) - M\tilde{y}_i, \quad i = 1,2,\ldots,n, \\ & t \leqslant \Delta_{y,k}, \\ & \sum\limits_{i=1}^{n} \tilde{y}_i = n - 1, \\ & y, \tilde{y} \in \{0,1\}^n, \end{cases} \tag{71}$$

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

In the sequel we shall write that two optimization problems are **equivalent** if an optimal solution of one problem straightforwardly provides an optimal solution of the other problem, and vice versa. Proposition 5.6 below is introduced in order to show that the new problem (71) (involving only linear constraints) is equivalent to problem (70). Since the essential difficulty resides in the "min" constraint of (70) and in the Hamming-distance constraints of (71), the proposition statement disregards the trust-region constraint on x and the constraint $t \leqslant \Delta_{y,k}$ (both of which are straightforwardly modeled in an MIQP). Corollary 5.1 below will establish the equivalence of problems (70) and (71), as a special case of Proposition 5.6.

Proposition 5.6. *(Mini-min reformulation) Let $\mu > 0$ be a given constant, N, n be positive integers, and let $f : \Omega \subseteq \mathbb{R}^N \to \mathbb{R}$ be a quadratic function, $g_i : \Omega \to \mathbb{R}, i = 1,2,\ldots,n$, be real-valued*

functions satisfying $0 \leqslant g_i(z) \leqslant M$*, for all* $z \in \Omega$*, for some* $M > 0$*. Then, the two following optimization problems are equivalent:*

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

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

Proof. We prove the proposition in two steps:

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

Let us consider the first assertion: (P_2) is a relaxation of (P_1).

Let $(\bar{z}, \bar{t})$ be a feasible solution of (P_1).
Consider now the point $(\bar{z}, \bar{y}, \bar{t})$ where, for $i = 1, 2, \dots, n$:

$$\bar{y}_i := \begin{cases} 0, \text{ if } i \text{ is the smallest index such that } \bar{t} = \min\limits_{i=1,2,\dots,n} \{g_i(\bar{z})\}, \\ 1, \text{otherwise.} \end{cases} \tag{72}$$

Let I be the unique index i such that $\bar{y}_i = 0$.

From the definition of $\bar{y}$, we note that:

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

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

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

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

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

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

For the second step, let us now show that: if (z^*, y^*, t^*) is an optimal solution of (P_2), then (z^*, t^*) is feasible for (P_1), *i.e.*, we want to prove that $t^* = \min_{i=1,2,\ldots,n}\{g_i(z^*)\}$.

By contradiction, we shall suppose that this optimal solution of (P_2) is such that

$$t^* \neq \min_{i=1,2,\ldots,n}\{g_i(z^*)\}.$$

Let us consider two cases:

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

Let I_{y^*} denote the unique index i such that $y_i^* = 0$.

Then, $y_{I_{y^*}} = 0$ and $y_i^* = 1$, for all $i \neq I_{y^*}$.

Using the fact that (z^*, y^*, t^*) is a feasible solution for (P_2), we have

$$t^* \geqslant g_{I_{y^*}}(z^*). \tag{73}$$

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

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

which contradicts (73).

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

$$\begin{aligned} \bar{z} &= z^*, \\ \bar{t} &= \min_{i=1,2,\ldots,n}\{g_i(z^*)\}, \end{aligned} \tag{74}$$

and

$$\bar{y}_i := \begin{cases} 0, & \text{if } i \text{ is the smallest index satisfying } g_i(z^*) = \min_{i=1,2,\ldots,n}\{g_i(z^*)\}, \\ 1, & \text{otherwise,} \end{cases} \tag{75}$$

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

We have:

- This new solution $(\bar{z}, \bar{y}, \bar{t})$ is feasible for (P_2). Indeed, for $i \neq I^*$, the ith constraint, $t \geqslant g_i(z) - My_i$, is satisfied for $(\bar{z}, \bar{y}, \bar{t})$, since M is an upper bound for the function $g_i(z)$ and $\bar{y}_i = 1$.

  If $i = I^*$, then on the one hand $\min_{i=1,2,\ldots,n}\{g_i(z^*)\} = \bar{t}$, and on the other hand $g_{I^*}(\bar{z}) = g_{I^*}(z^*) = \min_{i=1,2,\ldots,n}\{g_i(z^*)\}$ by definition of I^*. Therefore, the I^*th constraint of (P_2) is satisfied for $(\bar{z}, \bar{y}, \bar{t})$.

- In terms of objective-function values, it is clear that

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

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

□

Corollary 5.1. *The two problems (70) and (71) are equivalent.*

Proof. Consider the special case of Proposition 5.6 where $\Omega = \mathbb{R}^m \times \{0,1\}^n$, $z = (x, y)$, and $N = m + n$, and restrict both feasible sets of (P_1) and (P_2) by adding the two constraints: $\|x - x_k\|_\infty \leqslant \Delta_{x,k}$ and $t \leqslant \Delta_{y,k}$. □

2.3 Algorithm DFOb-d_{neck}

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

The new algorithm is named **Derivative-Free trust-region method for mixed binary necklace optimization**, and is denoted **DFOb-d_{neck}**. It follows exactly the steps of DFOb-d_H (Algorithm 6, given at the end of Chapter 3), except for the following specific changes:

- In **Step 1.5a**:
  Solve MIQP subproblem (71), instead of MIQP subproblem (31).

- In **Step 3**:
  Replace, in the new no-good cut, the Hamming-distance inequality:

  $$d_H(y, y_k) \geqslant K^*,$$

  by the n inequalities:

  $$d_H(y, Rot^i(y_k)) \geqslant K^*, i = 1, 2, \ldots, n,$$

  which are linear constraints equivalent to $d_{neck}(y, y_k) \geqslant K^*$ by (68).

Let us now derive a result of local convergence for the new algorithm DFOb-d_{neck}, analogous to that established in Lemma 4.2 and Theorem 4.1 for DFOb-d_H. First, let us consider the perturbed model:

$$\widetilde{m}^\epsilon(x, y_0) = \widetilde{m}(x, y_0) + \epsilon, \tag{76}$$

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

Lemma 5.1. *Under Assumption 4.1 and Assumption 4.2, the perturbed model $\widetilde{m}^\epsilon(\cdot, y_0)$ (defined with fixed $y = y^0$) is **fully linear** in $B_{y_0}(x_0, \Delta_x)$. In other words, for all $x \in B_{y_0}(x_0, \Delta_x)$, there exist positive constants $\kappa_f^\epsilon, \kappa_g^\epsilon$ such that:*

$$|f(x, y_0) - \widetilde{m}^\epsilon(x, y_0)| \leqslant \kappa_f^\epsilon \Delta_x^2, \tag{77}$$

and

$$\|\nabla_x f(x, y_0) - \nabla_x \widetilde{m}^\epsilon(x, y_0)\|_2 \leqslant \kappa_g^\epsilon \Delta_x, \tag{78}$$

with $\kappa_f^\epsilon = \kappa_f^* + \epsilon'$*, and* $\kappa_g^\epsilon = \kappa_g^*$*, and where* κ_f^* *and* κ_g^* *are the constants of Lemma 4.2.*

The proof is a straightforward adaptation of the proof of Lemma 4.2.

As for the convergence proof (detailed in Lemma 4.2) for the DFOb-d_H algorithm, from Lemma 5.1 and with the additional assumption that f is bounded from below, we can prove that the algorithm DFOb-d_{neck} is locally convergent, following the lines of the proof of convergence of the (continuous) DFO algorithm in [27] (Theorem 10.13).

3 Numerical results

This section presents comparative numerical results. After briefly presenting the comparison methodology, we propose in Subsection 3.1 a set of benchmark mixed binary optimization problems that features cyclic symmetry. It consists of a set of 25 instances constructed by transforming existing analytical test problems from the literature, plus one completely original problem related to the design of compressor blades in a helicopter turbomachine. Subsection 3.2 reports numerical results on the 25-instance set, while Subsection 3.3 presents comparative results on the helicopter application problem.

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

- The mesh adaptive direct search algorithm implemented in NOMAD software [1, 61],
- The surrogate-based optimization method implemented in RBFOpt [30] (based on radial basis functions),
- The derivative-free line-search bound constrained method, DFLBOX [63].

Following the methodology proposed in [73], we compare the solvers' performances in terms of number of evaluations of the objective function. This is a classical indicator for applications involving expensive objective-function evaluations where a solver is evaluated through its capacity to achieve a given function reduction within a limited budget of *simulations* (evaluations of the objective function). Remark that we did not integrate techniques to overcomes cyclic symmetry

property of other DFO methods. One technique can be use to avoid recomputing equivalent solutions is to check if one equivalent necklace is evaluated, if so we neglect this solution and go to other candidates.

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

$$f(x_0) - f(\bar{x}) \geqslant (1-\tau)(f(x_0) - f^*), \tag{79}$$

where, in the sequel, f^* denotes the best function value found by any solver (or the global-minimum value, if known), x_0 is the starting point for each solver (or the best point of the initial interpolation points), and τ is the desired accuracy, a user-defined tolerance value (in our tests, $\tau = 10^{-3}$ or 10^{-5}). If a solver does not provide a solution that satisfies (79), we consider that it fails.

Performance and data profiles (see [73]) are complementary tools to compare solvers on a collection of problems.

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

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

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

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

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 $\rho_s(\eta), \eta > 1$.

The performance profile of a solver depends therefore on the other solvers tested. For instance, the value of the performance profile of a given solver for a performance ratio of 2 is the number of problems solved by this solver within less than twice the number of evaluations required by the most efficient solver for each problem. However, this does not give an accurate information

on the number of evaluations required by a solver to solve a whole collection of problems. This is the reason why data profiles are widely used to compare DFO methods. Data profiles give the fraction of problems that can be solved within a given number of objective-function evaluations. Data profiles therefore provide the performance of the solvers for any given simulation budget.

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

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

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.

Table 4 summarizes the options and the main parameter values used in our comparison for the five solvers under study: DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFOpt and DFLBOX.

Tab. 4.: Solver parameters and options used for the benchmark.

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

3.1 New mixed binary optimization test problems featuring cyclic symmetry

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

The 25 analytical benchmark problems. To our knowledge, no instance of cyclic-symmetry mixed binary optimization problems are proposed in the literature. The last two columns of Table 5 list the name and the source of 25 optimization problems that we selected to build our 25-instance benchmark problem set, more details in Appendix 2. Originally these benchmark problems involve continuous optimization variables only. Some of these optimization problems also include constraints. The present study considers mixed binary optimization problems featuring cyclic symmetry and involving only bound constraints. Thus, we describe now how we transform these instances from the literature into new mixed binary bound-constrained optimization problems featuring cyclic symmetry. A first step in this transformation process is to propose intermediate mixed *categorical* (involving integer variables not related to effective quantities) optimization problems.

The 10 instances from [64] are associated with continuous-optimization *minimax problems* of the form:

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

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

$$\min_{x\in[\underline{x},\bar{x}],w\in\{1,2,\ldots,l\}} \tilde{F}(x,w) := \begin{cases} f_1(x), & \text{if } w = 1, \\ f_2(x), & \text{if } w = 2, \\ \quad\vdots & \\ f_l(x), & \text{if } w = l, \end{cases} \tag{80}$$

for which the integer w is the category variable.

For each of the 15 remaining benchmark problems (the 12 problems from [50, 36, 80] and the three problems from [71]), the transformation into a mixed categorical problem goes as follows.

Tab. 5.: Analytical necklace-optimization benchmark problems

Test problem	m	n	**source instance**	**reference**
Branin-nl	1	3	Branin	[36]
Camel-nl	1	3	Camel	[36]
Goldstein-Price-nl	1	3	Goldstein-Price	[36]
Hartman3-nl	2	3	Hartman3	[36]
Hartman6-nl	5	3	Hartman6	[36]
Shekel7-nl	3	3	Shekel7	[36]
Shekel10-nl	3	3	Shekel10	[36]
HS2-nl	1	3	HS2	[50]
HS29log-nl	1	3	HS29log	[50]
HS3-nl	1	3	HS3	[50]
CB1-nl	2	2	CB1	[64]
CB2-nl	2	2	CB2	[64]
MAD1-nl	2	2	MAD1	[64]
MAD2-nl	2	2	MAD2	[64]
QL-nl	2	2	QL	[64]
Pentagon-nl	6	3	Pentagon	[64]
RosenSuzuki-nl	4	3	RosenSuzuki	[64]
WF-nl	2	2	WF	[64]
Wong2-nl	10	4	Wong2	[64]
Wong3-nl	20	6	Wong3	[64]
Perm6-nl	5	3	Perm6	[80]
Perm8-nl	7	3	Perm8	[80]
Ex8-1-1-nl	1	3	Ex8-1-1	[71]
Ex8-1-4-nl	1	3	Ex8-1-4	[71]
Sporttournament06-nl	14	3	Sporttournament06	[71]

We first restrict the last continuous variable, say x^{end}, to take only a finite number of values in the discretized-interval set:

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

where $\underline{x}^{end}$ and $\bar{x}^{end}$ are respectively the lower and upper bounds of the variable x^{end} in the original problem, and where the number, l, of categories is to be set by the user. One thereby obtains a mixed categorical optimization problem involving an objective function of the form $\tilde{F}(x, w)$, where w is a category variable ($w \in \{1, 2, \ldots, l\}$). In our numerical tests, we set the number of categories to $l = 4$ for these 15 benchmark problems (*i.e.*, other than those that originate from minimax problems, for which l is given by the original instance).

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

$$\min_{x \in [\underline{x}, \bar{x}], y \in \{0,1\}^2} f(x, y) := \begin{cases} \tilde{F}(x, 1), & \text{if } y = (0, 0), \\ \tilde{F}(x, 2), & \text{if } y = (0, 1), (1, 0), \\ \tilde{F}(x, 3), & \text{if } y = (1, 1). \end{cases} \tag{81}$$

The resulting new mixed binary necklace-optimization instances are listed in Table 5 together with the number of continuous variables (m), the number of binary variables (n), the name of the original instance from which it was constructed together with the literature reference. The particular choice of problems is motivated by the sake of comparing methods on a diverse range of problem dimensions and difficulties, including the presence of multiple local minima.

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

3.2 Results obtained on the 25 analytical problems

The results obtained with DFOb-d_H, DFOb-d_{neck}, NOMAD, RBFOptand DFLBOX over the 25 analytical problems are presented under the form of performance profiles (Figure 5) and data

profiles (Figure 6). The required accuracy is set to $\tau = 10^{-3}$. The number of objective-function evaluations necessary to reach the best known solution for each problem is also displayed in Figure 7. If the solution is not reached by a solver, no point is displayed but the percentages of successes are indicated for each solver. For all the numerical results, DFOb-d_H results are displayed in blue, DFOb-d_{neck} results in red, NOMAD results in black, RBFOpt in magenta, and DFLBOX in cyan.

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

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

The performance profiles of Figure 5 reveal that for a (number-of-simulation) performance ratio equal to 1 (to indicate the percentage of problems for which the solver does as well as the best solver), DFLBOX, DFOb-d_{neck}, and RBFOpt reaches the best value: 36%.

Figure 5 also shows that the new necklace distance improves the efficiency of the DFOb method when addressing cyclic-symmetry optimization problems, as DFOb-d_H solves only 28% of the problems with the minimal number of objective-function evaluations.

In addition, the data profiles of Figure 6 illustrate that DFOb-d_{neck} solves most problems with a relatively small number of objective-function evaluations. For instance, 62% of the problems are solved within a number of simulations less than 15 times the number of variables.

Finally, Figure 7 shows that: DFLBOX requires a small number of simulations (less than 200) for the 60% of the problems it succeeds to solve, the two DFOb methods need less than 200 function evaluations for all successful problems but one (only one successful problem requires 270 function evaluations for each solver), whereas RBFOpt and NOMAD require more than 200 objective-function evaluations for 29.4% and 66.7% of their successful problems, respectively.

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

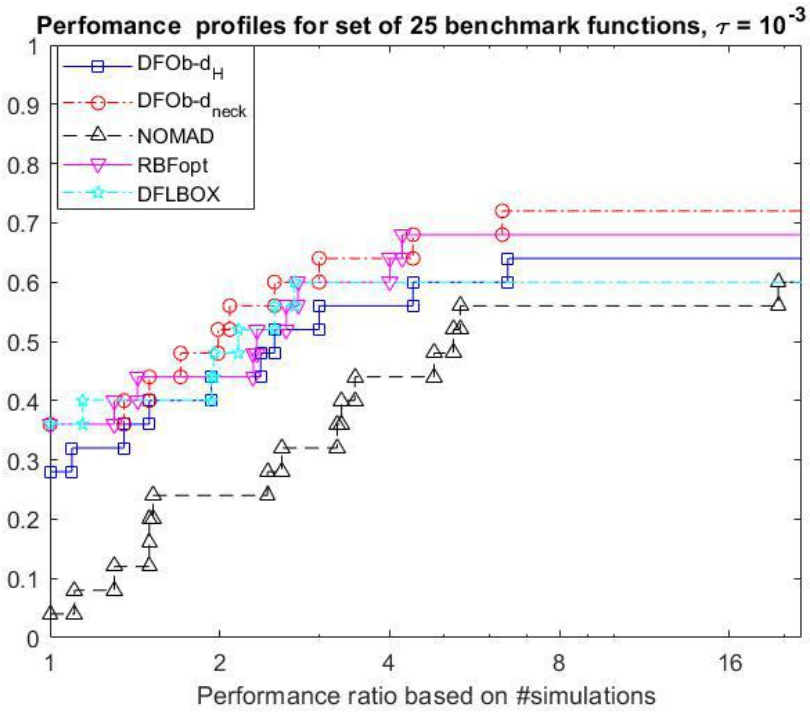

Fig. 5.: Performance profiles of the five solvers for the 25 analytical problems

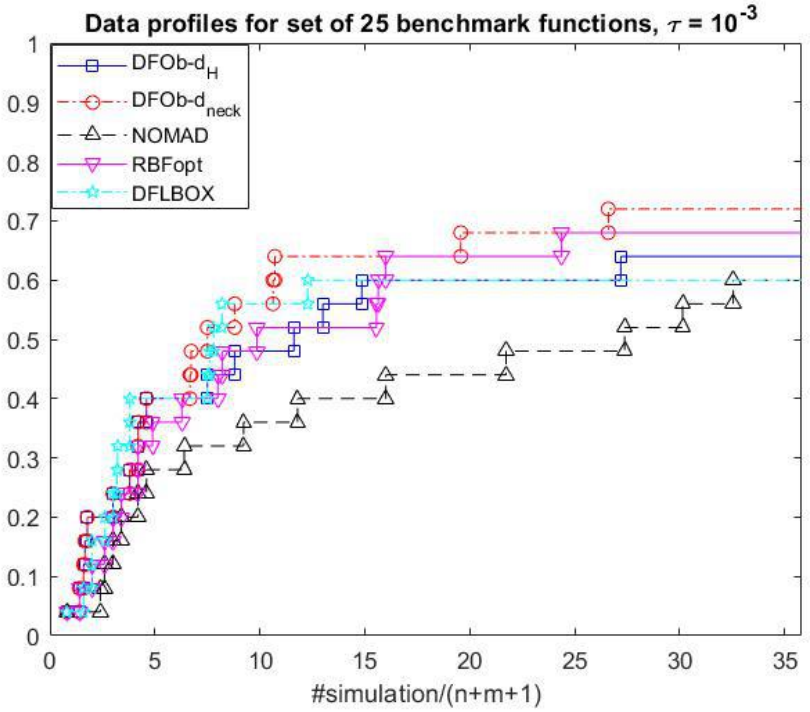

Fig. 6.: Data profiles of the five solvers for the 25 analytical problems

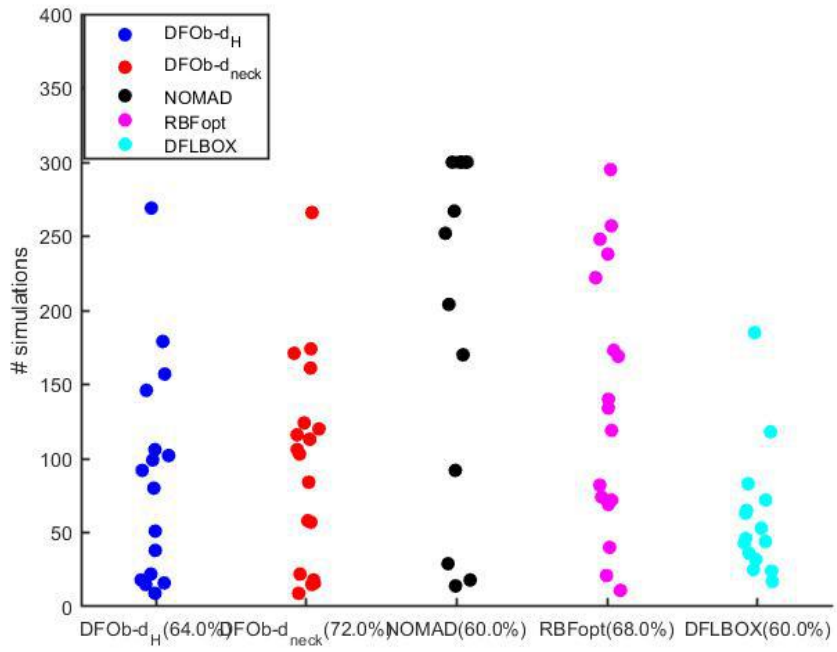

Fig. 7.: Number of function evaluations to reach f^* up to $\tau = 10^{-3}$ for each of the 25 analytical problems for the five solvers (successful runs only) together with percentage of successes (in parentheses)

In order to investigate the impact of problem dimension on the performance of the different methods, we analyze separately the subset of the 10 benchmark problems that involve 3 or more continuous variables (Figures 8, 9, and 10). RBFOpt and DFOb-d_{neck} succeed to solve 7 problems over 10, whereas DFOb-d_H and NOMAD only solve 5 problems, and DFLBOX only 4. These results indicate that, for higher-dimensional problems, RBFOpt and DFOb-d_{neck} reach the best performances, and DFLBOX the worst ones. This is not completely surprising as DFOb-d_H, DFLBOX and NOMAD are local optimization methods (with respect to the continuous search space, $\mathbb{R}^m$). The new method DFOb-d_{neck}, is also a local solver but with the help of the new distance, it succeeds to find better solutions within the budget of simulations. RBFOpt is designed to address global optimization problems. Current research work is dedicated to this global-optimization issue via the proposition of a diversification search strategy for DFOb-d_{neck} in the continuous search space, $\mathbb{R}^m$.

3.3 Design of compressor blades in a helicopter turbomachine

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

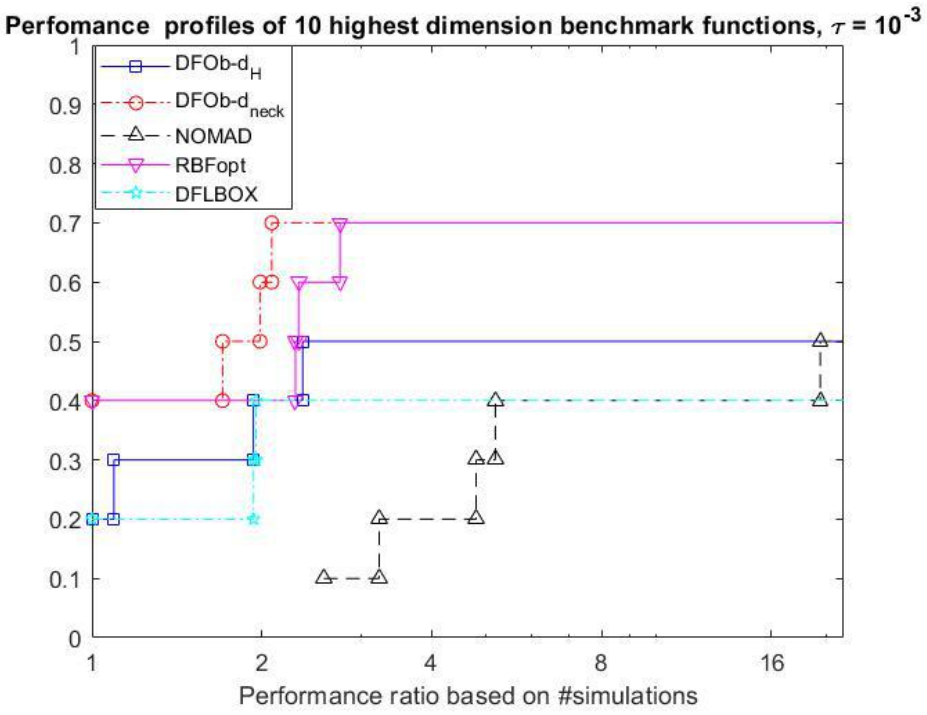

Fig. 8.: Performance profiles of the five solvers for the 10 highest dimension analytical problems

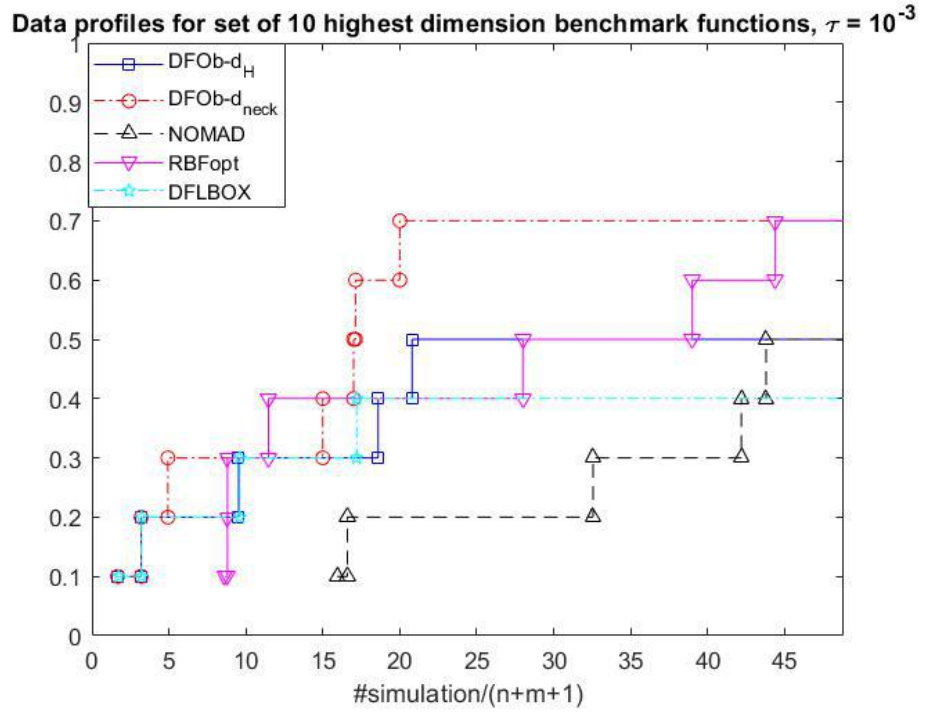

Fig. 9.: Data profiles of the five solvers for the 10 highest dimension analytical problems

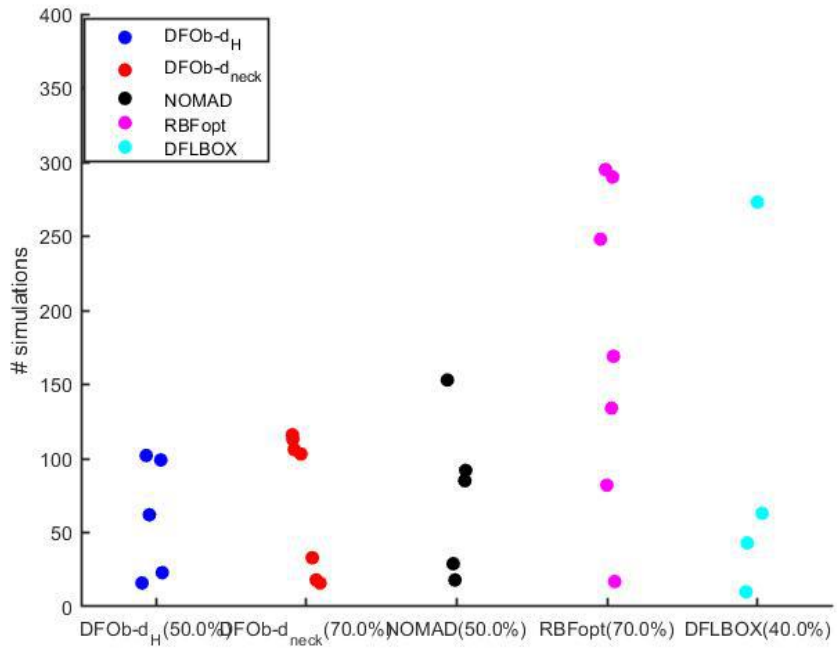

Fig. 10.: Number of function evaluations to reach f^* up to $\tau = 10^{-3}$ for each of the 10 highest dimension analytical problems for the five solvers (successful runs only) together with percentage of success (in parentheses)

For the sake of fair comparison, we repeat 50 runs of each of the four solvers. Each run starts with a different design of experiments (of cardinality $n + m + 1 = 14$), following the construction process described in Section 2. Again, we report performance (Figure 11) and data profiles (Figure 12), as well as the number of iterations (Figure 13) to reach, this time, a reduced accuracy of $\tau = 10^{-5}$ is mainly considered (due to the scale of the objective function) on the function reduction (79) associated with these 50 runs, we also show the results with the accuracy of $\tau = 10^{-3}$ after.

Considering first the results with the accuracy $\tau = 10^{-5}$. DFLBOX shows very good performances on this test case: it can solve all of the 50 repetitions within a very small number of simulations. As already noticed on the benchmark function results, DFLBOX is more efficient on small-dimensional problems than on large ones. For this application, which involves only one continuous variable, it clearly outperforms the four other methods. DFLBOX algorithm is based on line searches in each coordinate direction for both continuous and discrete variables.The analysis of the results of this blackbox optimization problem revealed its particular structure: for all feasible combinations of values of the binary variables tested (a few hundreds), the minimum of the function corresponds to the same value of the continuous variable. In other words, this simplified optimal design application problem appears to be separable. Indeed, we observed that for all tested combinations

of the $n = 12$ binary variables, the resulting univariate continuous optimization problem yields the same optimal value for the continuous variables. As a consequence, one could therefore separately minimize the univariate continuous problem for one combination of values of the binary variables, then fix the continuous variables to the corresponding value, and then finally address the resulting combinational combination problem (in 12 binary variables) The relaxation used in DFLBOX works very well on such a problem: one continuous variable and 12 binary variables; this methodology is indeed quite efficient as shown in the presented results.

Concerning the four other methods, the results show the very good performance of DFOb-d_{neck}, as it reaches the value f^* (up to τ accuracy) for 88% of the runs, compared with $84\%, 56\%$ and 78% for DFOb-d_H, NOMAD and RBFOpt, respectively. One therefore observes that, depending on the initial design of experiments, some runs do not achieve to reach the required reduction of the objective function. However, the two versions of DFOb are robust methods with regard to the initial design, with more than 80% of success within the budget of simulations. Data profiles in Figure 12 show the robustness of the DFOb methods and their good performances in terms of number of simulations compared to RBFOpt and NOMAD methods. Figure 13 illustrates the efficiency of DFOb-d_{neck}, with only one successful run that requires more than 150 simulations. Finally, Figure 16 displays the distribution of the 50 solutions found by DFOb-d_{neck} for each of the 50 runs. For 88% of the runs, DFOb-d_{neck} converged to a point, denoted (x^*, y_1^*), corresponding to the minimal objective-function value f^* (probably a global minimum); the remaining runs terminated with three other (locally-optimal) solutions, denoted (x^*, y_2^*), (x^*, y_3^*) and (x^*, y_4^*). All the runs converged to the same value of the continuous-variable component: $x^* = 0.03$ (corresponding to some geometry feature of the blades). The discrete-variable components of these four different solutions found by DFOb-d_{neck} are displayed in Figure 17, where the 0 and 1 values correspond to different pre-defined types of blades.

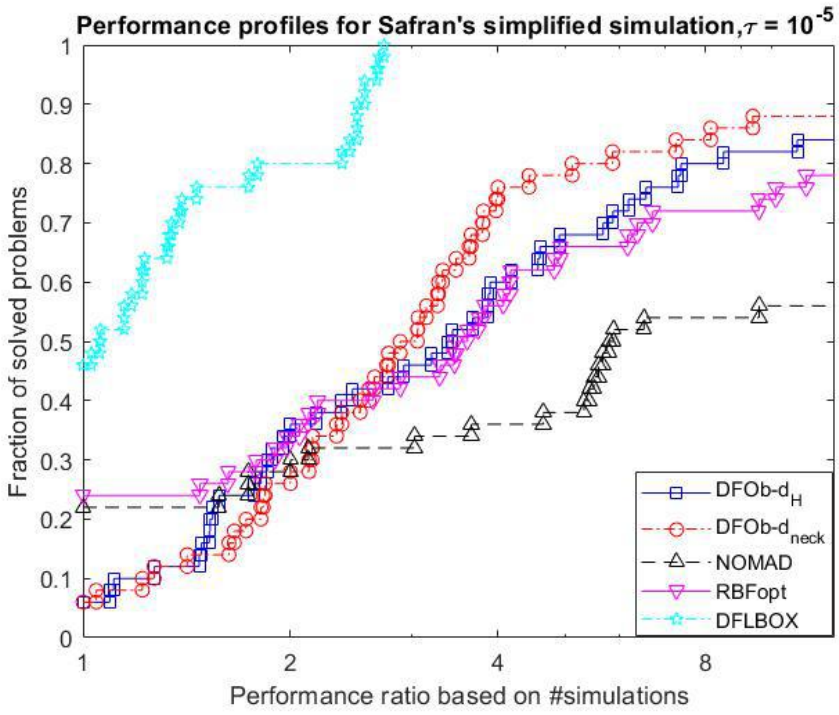

Fig. 11.: Performance profiles of the five solvers for the blade design application with 50 repetitions, accuracy $\tau = 10^{-5}$

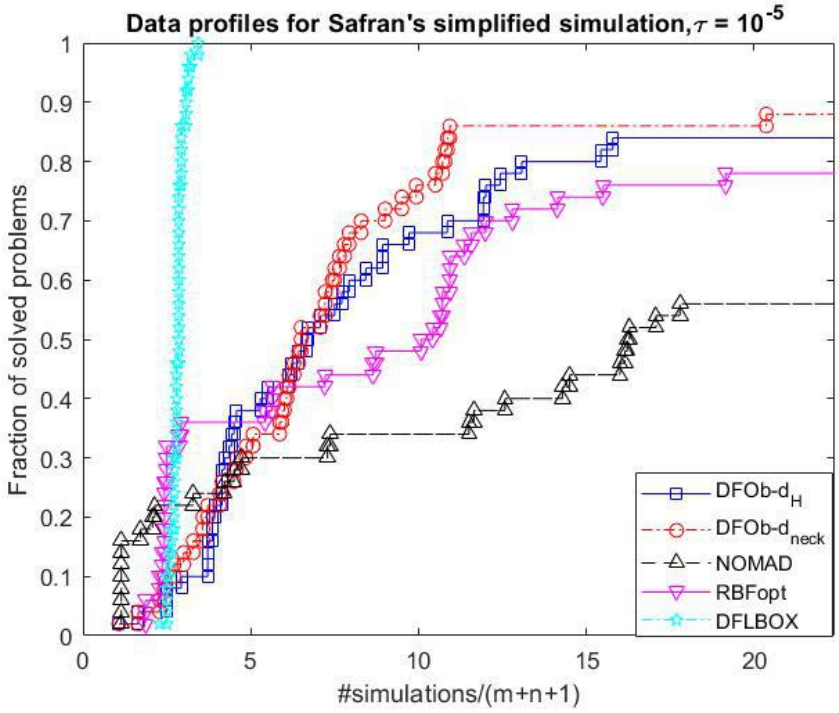

Fig. 12.: Data profiles of the five solvers for the blade design application with 50 repetitions, accuracy $\tau = 10^{-5}$

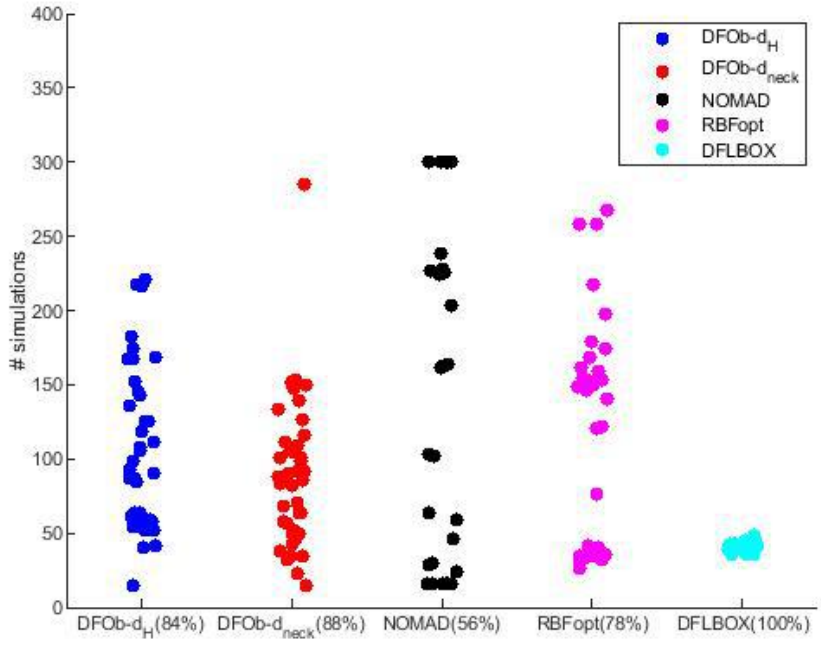

Fig. 13.: Number of function evaluations to reach f^* up to $\tau = 10^{-5}$ for each of the 50 runs of the blade design application for the five solvers (successful runs only) together with percentage of success (in parentheses)

The results with the accuracy $\tau = 10^{-3}$ are shown in the Figures (15,14). The results are slightly different compare to the above results with the accuracy of 10^{-5}: RBFOpt shows a better performance with 90% of successes runs. The results stay the same for four other methods: DFOb-d_H, DFOb-d_{neck}, NOMAD, and DFLBOX.

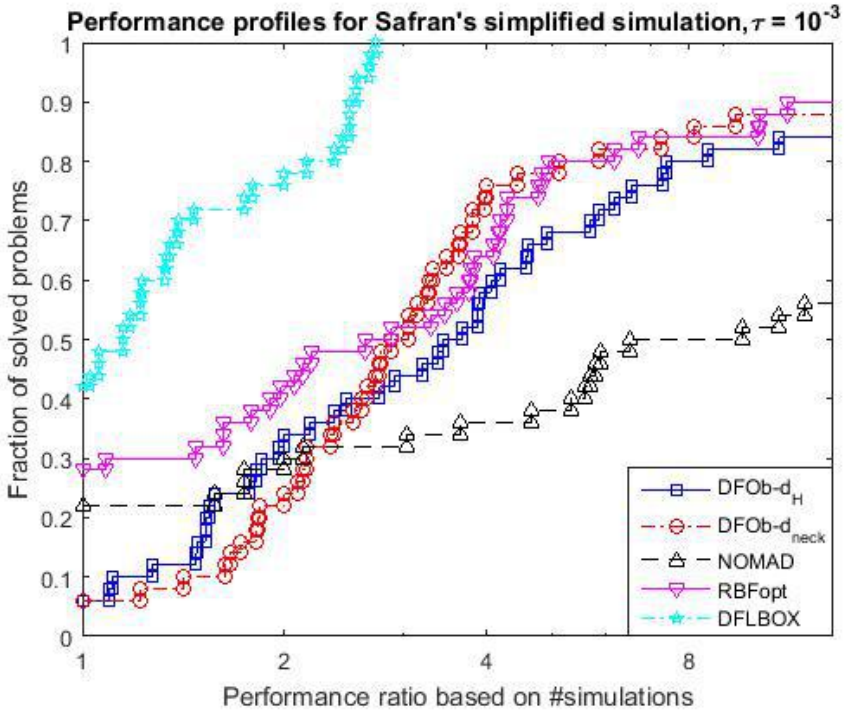

Fig. 14.: Performance profiles of the five solvers for the blade design application with 50 repetitions, accuracy $\tau = 10^{-3}$

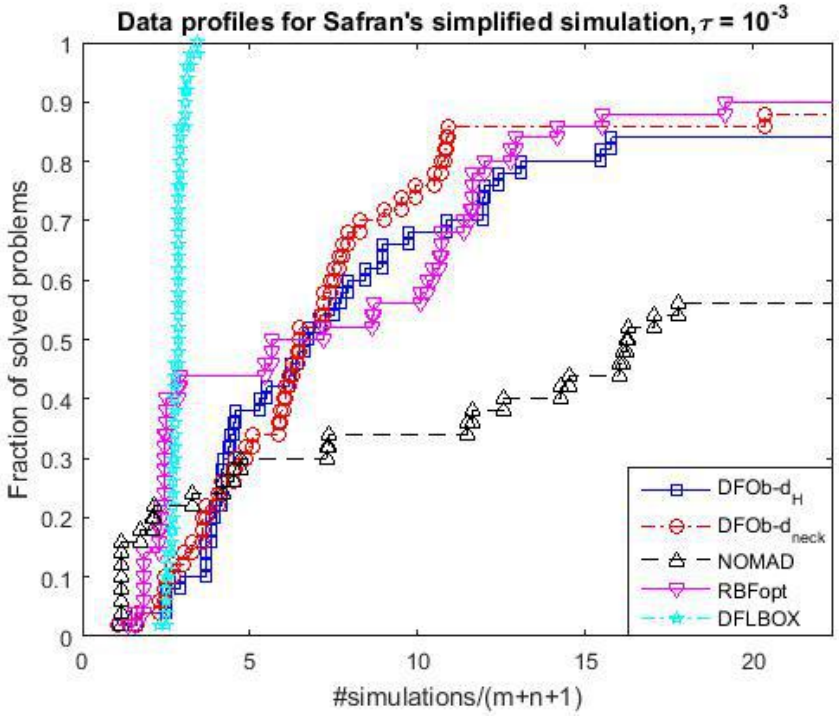

Fig. 15.: Data profiles of the five solvers for the blade design application with 50 repetitions, accuracy $\tau = 10^{-3}$

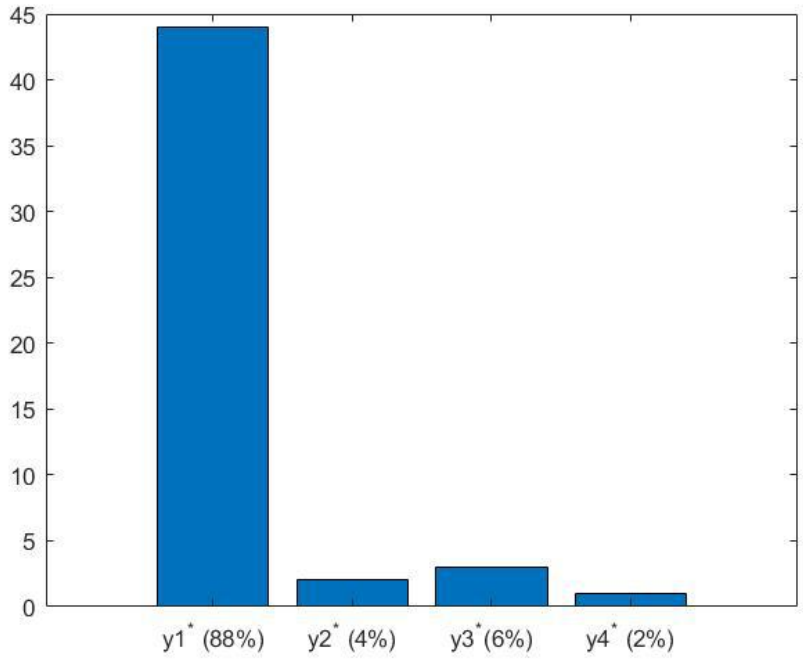

Fig. 16.: Distribution of the binary component of the 50 solutions found by DFOb-d_{neck} for the blade design application.

4 Conclusion

In this chapter we addressed derivative-free mixed binary optimization problems involving a cyclic symmetric property, by proposing an adapted distance, d_{neck} (necklace distance), for the binary search space. We presented theoretical results related to the linear formulation of constraints involving this necklace distance that allowed us to integrate d_{neck} in the trust-region derivative-free method DFOb-d_H proposed by [25] for mixed binary problems, in place of the Hamming distance. The convergence of both DFOb-d_H and that of the adapted algorithm, named DFOb-d_{neck}, to a locally-optimal solution was proved.

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

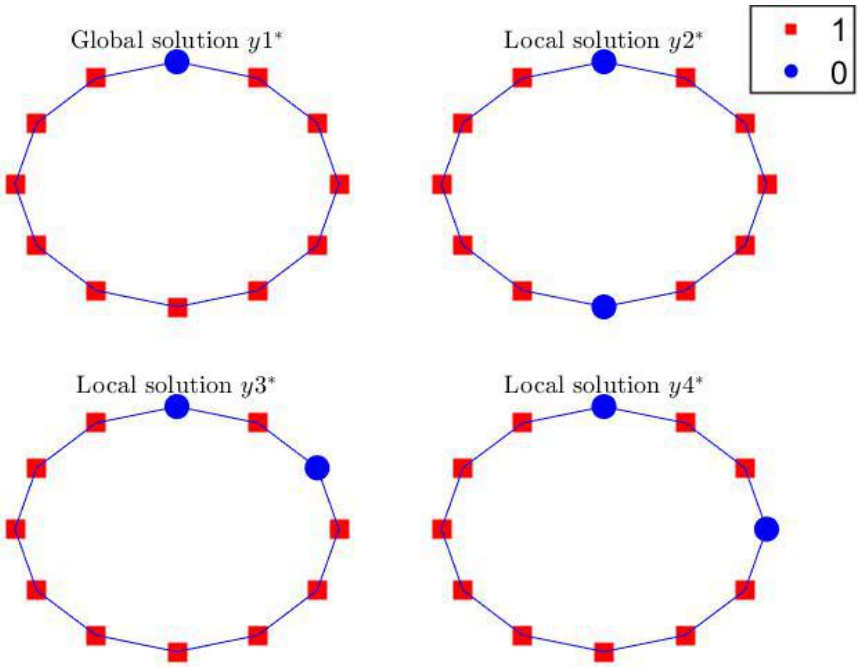

Fig. 17.: The four solutions found by DFOb-d_{neck} for the blade design application (the continuous component has the same common value: $x^* = 0.03$)

6

Further analysis on the optimization parameters

This chapter provides an analysis of the impact of the number of points, $nDoE$, in the interpolation set on the performance of the DFOb-d_{neck} algorithm in Chapter 5. Two options are studied: $nDoE = n + m + 1$ and $nDoE = 2(n + m) + 1$ over the 25 analytical problems detailed in Table 6. The initial designs are generated by the adapted greedy algorithm which will be described in detail in Part II.

We repeat 20 runs for each problem, so in total we have 500 runs. We report the results in performance profiles (Figure 18), data profiles (Figure 19), and the number of iterations (Figure 20). Results for DFO-d_{neck} with $nDoE = n + m + 1$ are displayed in blue, whereas the results for DFO-d_{neck} with $nDoE = 2(m + n) + 1$ are in red.

These three figures show that, for this 25 analytical problems, the results are similar for the two sizes of DoE: $nDoE = 2(m+n)+1$ and $nDoE = m+n+1$: DFOb-d_{neck} with $nDoE = 2(m+n)+1$ succeeds to solve 86.5% of the problems, whereas DFOb-d_{neck} with $nDoE = m + n + 1$ succeeds to solve 86.3%. The performance profiles in Figure 18 show that for a ratio equal to 1, DFOb-d_{neck} with $nDoE = m+n+1$ reaches the best value: 43.9%, while DFOb-d_{neck} with $nDoE = 2(m+n)+1$ yields a value of 58.3%. The data profiles of Figure 19 and the number of simulations to reach the best solutions in Figure 20 show very similar results for $nDoE = m + n + 1$ and for $nDoE = 2(m + n) + 1$.

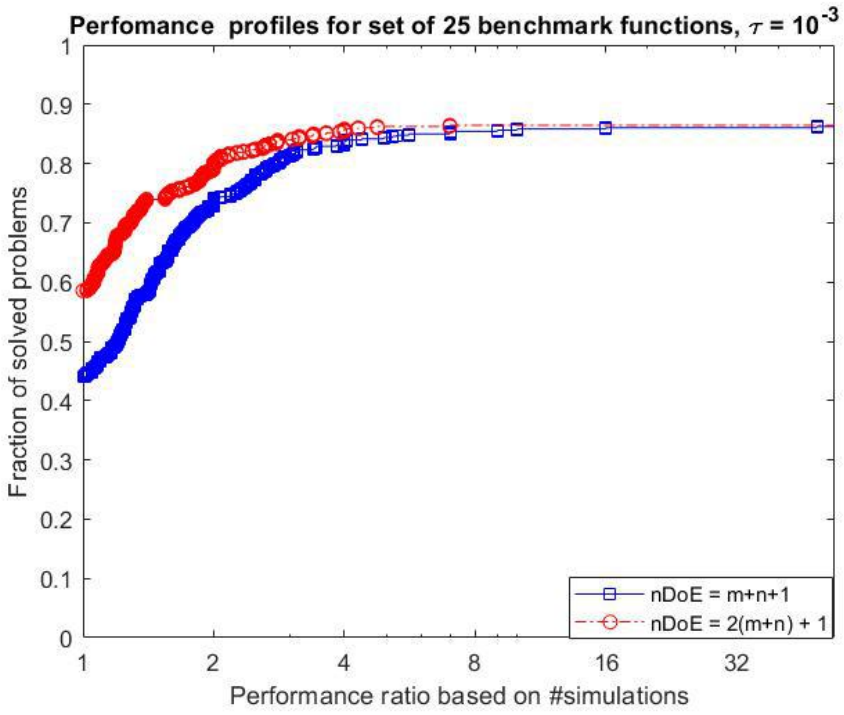

Fig. 18.: Performance profiles of DFOb-d_{neck} with $nDoE = m + n + 1$ and $nDoE = 2(m + n) + 1$ for the 25 analytical problems

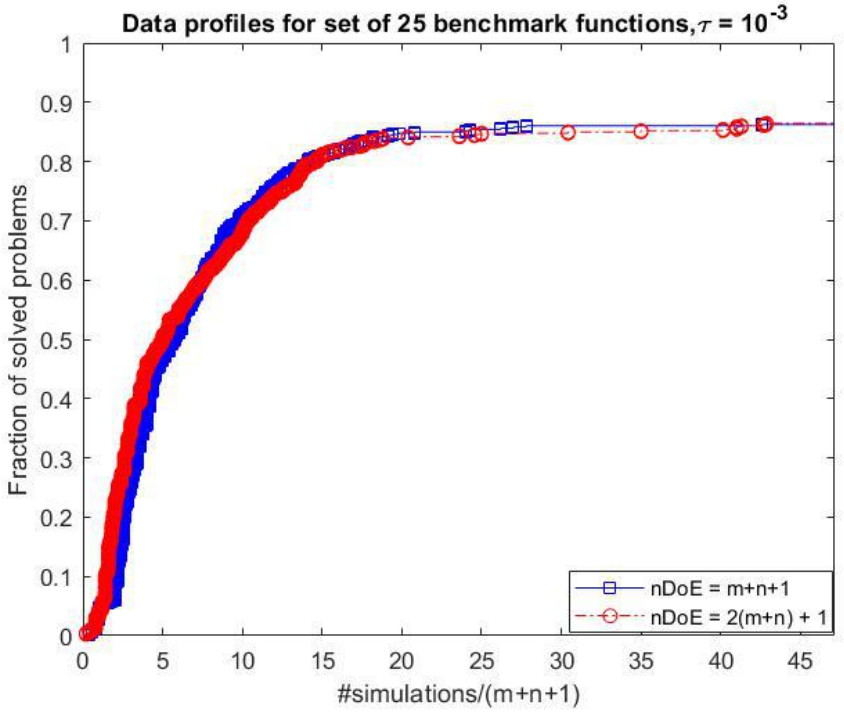

Fig. 19.: Data profiles of DFOb-d_{neck} with $nDoE = m + n + 1$ and $nDoE = 2(m + n) + 1$ for the 25 analytical problems

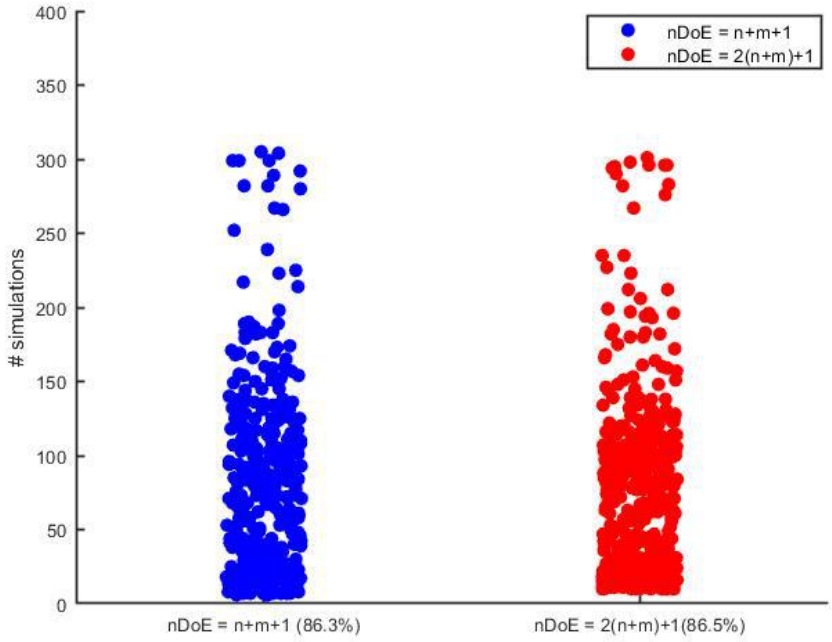

Fig. 20.: Number of function evaluations to reach f^* up to $\tau = 10^{-3}$ for each of the 25 analytical problems for DFOb-d_{neck} with $nDoE = m + n + 1$ and $nDoE = 2(m + n) + 1$ (successful runs only) together with percentage of success (in parentheses)

The results show that the number of points in the interpolation set has a small impact in the optimization procedure (when this number is relatively small, since we are considering an expensive-simulation context). It is likely that, the distribution of the points itself is more impactful. We suspect that an initial set of well-distributed points, especially for the set of discrete variables, provides useful information to build good approximation models for the optimization and should help to avoid local solutions. Therefore, the second part of the thesis is dedicated to the design of experiments in which we can generate a "good" initial design for the optimization.

Conclusion

Part I presents the main materials for the first subject of the thesis which is dedicated to derivative-free optimization trust-region methods. The work is an extension of [25], adapted to cyclic symmetry problems. The contribution of the author in this part is, first, the convergence proof of the original work in [25], a new method, DFOb-d_{neck}, and its convergence proof. This method is an extension of the original method DFOb-d_H for cyclic symmetric problems with mixed continuous and binary variables. The introduction of the proposed necklace distance allows to deal with the symmetry and to define new sub-problems for the trust region method so that the problem can be addressed by standard MINLP solvers. The preliminary comparative results obtained on a benchmark of test functions and on the application provided by Safran are very encouraging in terms of both robustness and of simulation costs when compared to state-of-the-art methods.

Part II

Design of experiments in mixed continuous and discrete space

7

Design of experiments for continuous variables problems

Contents

1	Geometrical designs	**92**
2	Marginal designs	**92**
3	Low discrepancy designs	**93**
4	Super-sample from kernel herding	**94**

Design of experiments (DoE) are used in various contexts such as optimization or uncertainty quantification based on a time-consuming numerical simulator. It aims to select a limited number of values to assign to the simulator input variables that give a maximal knowledge on the simulator outputs of interest.

In recent years, the need for efficient design of experiments has emerged as a key research area for analyzing complex physical numerical models. Indeed, such models are usually expensive to evaluate, sometimes several hours or even days are required to run one single simulation, which means that using them to conduct optimization studies or uncertainty quantification and sensitivity-analysis investigations is in general too computationally demanding. In the computer experiments community, the standard practice thus consists in building a surrogate model of the numerical code and use it as a proxy for all further intensive computations required by optimization or uncertainty propagation. However, the final surrogate model accuracy heavily depends on the available samples of the computer code inputs/outputs relationship. For this task, it is critical to build a Design of Experiments (DoE) which provides information about all portions of the experimental region. For example, two appealing concepts are the so-called *space filling*, [105, 92] and the Latin Hypercube Sampling (LHS) [67]. If designs of experiments for continuous variables have been extensively studied with several space-filing criteria minimax and maximin [87], discrepancy [96], maximum projection [57], the case where models involve both continuous and discrete input variables has been much less investigated and tested.

This chapter focuses on design of experiments in the continuous space $\mathcal{D} = \mathbb{R}^m$. Remark that as mentioned in [105], the two most important requirements for a good DoE are:

- Space-filling property: The design points should be uniformly spread over the entire design space

- Noncollapsing property: Two design points should not share any coordinate values if we not know a priori which dimensions are important

This chapter is organized as follows. In Sections 1, 2, 3, and 4 we present the geometrical, marginal, low discrepancy and super-sample from kernel herding designs for continuous variables problems respectively.

1 Geometrical designs

In this class of DOEs, one uses a deterministic procedure to create initial designs. We highlight some space-filling designs:

Full factorial designs [40]. It is the most straightforward method to create initial sets uniformly. The main idea is to divide each dimension i into n_i intervals, called cells, of equal length, then select the centers of these cells to constitute the DoE $n_1 \dots n_m$ cells. It is clear that the sample set obtained by full factorial designs are highly uniform, but the noncollapsing property is not satisfied. Moreover, a full-factorial design can only be used if the number of sampling points is of the form $n_1 \dots n_m$.

Fractional factorial designs [40]. It is a variation of the full factorial designs in which one only uses a fraction of the sample set created by the full factorial designs.

2 Marginal designs

Latin Hypercube Sample (LHS) designs [67]. LHS divides each axis into n intervals of equal length so that one considers n^m cells, then selects n points in each of these n^m cells so that there are no two points with the same coordinate in any dimension. To do that, one can for instance simply perturb the coordinates of the points as needed, starting from the beginning.

Optimal LHS designs [55]. This type of design aims at improving the space-filling property of LHS by using either the *minimax distance*:

$$\min_{|S|=p} \max_{x \in D} d(x, S),$$

where $d(x, S) = \min_{x' \in S} d(x, x')$, p is the number of required points; or by using the *maximin distance*:

$$\max_{|S|=p} \min_{x,x' \in S} d(x, x').$$

3 Low discrepancy designs

Discrepancy is a concept for measuring the "difference" between uniform point distribution and a given point distribution. For instance, in the case of dimension 1, given the empirical distribution F_N of a set X of N points $x^1, x^2, \ldots, x^N$, and F_U a uniform distribution on $[0, 1]$, we define the discrepancy, $D_N(X)$, by:

$$D_N(X) = \sup_{x \in [0,1]} |F_N(X) - F_U(X)|.$$

From the above description, if a sample set has lower discrepancy, it approximates the empirical distribution better. The low discrepancy design can be based on Sobol sequences, Halton sequences, Hammersly sequences or Faure sequences (details in [41]). However, in the implementing scope of this study, we focus on adapting the low discrepancy design based on Sobol sequences to a mixed binary and continuous space, which will be presented in the next subsection.

Low discrepancy designs based on Sobol sequences

Low discrepancy designs based on Sobol sequences are defined from primitive polynomials over the finite field $\mathbb{Z}_2 = \{0, 1\}$ (*i.e.*, with coefficients either 0 or 1):

Definition 7.1. A polynomial, $p(t)$, of degree s of the form $t^s + u_1 t^{s-1} + \ldots + u_{s-1} t + u_s$ is *primitive* over $\mathbb{Z}_2$ if it is irreducible and it has order $2^s - 1$.

The term of irreducible signifies polynomials which cannot be factored into nontrivial polynomials over the field. The order of polynomial is the smallest positive number i such that $p(t)$ divides $t^i - 1$ (or $t^i + 1$).

Definition 7.2. A *Sobol sequence* $S = \{x^0, x^1, \ldots, x^{N-1}\}$ in dimension 1 is defined as:

$$x^i = \frac{1}{2^h} (\oplus_{k=1}^{h_i} p_k l_k),$$

where $(p_1, p_2, \ldots, p_h)$ is the binary representative of i, $h_i = \begin{cases} 1 + \lfloor \log_2 \rfloor & \text{if } i \neq 0, \\ 1 & \text{if } i = 0, \end{cases}$

and $\oplus$ refers to the addition in $\mathbb{Z}_2$, and each $l_k, k > s$ is obtained using the following recurrence relationship:

$$l_k = 2u_1 l_{k-1} \oplus 2^2 u_2 l_{k-2} \oplus \ldots \oplus 2^{s-1} u_{s-1} l_{i-s+1} \oplus (2^s l_{i-s} \oplus l_{i-s}),$$

where u_k are the coefficients of a primitive polynomial degree s over $\mathbb{Z}_2$, and $l_1, \ldots, l_s$ must be odd integers such that $1 \leqslant l_k \leqslant 2^k, k = 1, 2, \ldots, s$. In order to construct a Sobol sequence in dimension d, we choose d distinct primitive polynomials.

A low discrepancy design based on Sobol sequences is rapid to build given the binary's representative. However, due to the recurrence relation, the new points constructed tend to be in the vicinity of the previous iteration.

4 Super-sample from kernel herding

Another type of method in this class DoEs is based on Maximum Mean Discrepancy (MMD) designs, called kernel herding [22]. MMD (or kernel distance) between two probability distributions $\mathbb{P}_{X_1}$ and $\mathbb{P}_{X_2}$ is defined as follows:

$$MMD^2(\mathbb{P}_{X_1}, \mathbb{P}_{X_2}) = \|\mu_{\mathbb{P}_{X_1}} - \mu_{\mathbb{P}_{X_2}}\|^2_{\mathcal{H}_k}, \tag{82}$$

where $\mathcal{H}_k$ is a Reproducing Kernel Hilbert Space (RKHS) with kernel k and $\mu_{\mathbb{P}} = \int k(x, .) d\mathbb{P}(x)$ is the representative of the probability distribution $\mathbb{P}$. Recall that, a RKHS is a Hilbert space in real-value functions in a closed bounded subset X of $\mathbb{R}^m$ with the property that for each $t \in X$ the evaluation functional is a bounded linear functional.

We start with two important definitions to follow up this part:

Definition 7.3. (Positive definite kernel, [52]) Consider the domain $\mathcal{I}$. A function $k : \mathcal{I} \times \mathcal{I} \to \mathbb{R}$, is a positive definite (p.d) if it satisfies the two following criterion

1. *symmetric*: for any pair of inputs $x_i, x_j \in \mathcal{I}$, k satisfies

$$k(x_i, x_j) = k(x_j, x_i),$$

2. *positive definite*: for any p inputs $x_1, x_2, \ldots, x_p \in \mathcal{I}$ and any p constants $c_1, c_2, \ldots, c_p \in \mathbb{R}$, k satisfies

$$\sum_{i=1}^{p} \sum_{j=1}^{p} c_i c_j k(x_i, x_j) \geqslant 0.$$

Definition 7.4. (Stationary kernel (or translation invariant kernel), [52]) A stationary kernel k can be expressed in terms of a multivariate function $\tilde{k} : \mathbb{R}^m \to \mathbb{R}$ of a single $m-$dimensional variable, *i.e.,*

$$k(x_1, x_2) = \tilde{k}(x_1 - x_2), \;\; x_1, x_2 \in \mathbb{R}^m.$$

A standard example is given by the anisotropic Gaussian kernel:

$$k(x_1, x_2) = \prod_{l=1}^{m} e^{-\lambda_l^2 (x_l^1 - x_l^2)^2}.$$

If $X_1 \sim \mathbb{P}_{X_1}$ and $X_2 \sim \mathbb{P}_{X_2}$ we can expand MMD^2 to get an expression involving $k(\cdot, \cdot)$ explicitly:

$$MMD^2(\mathbb{P}_{X_1}, \mathbb{P}_{X_2}) = \mathbb{E}_{X_1, X'}(k(X_1, X')) + \mathbb{E}_{X_2, X''}(k(X_2, X'')) - 2\mathbb{E}_{X_1, X_2}(k(X_1, X_2)), \tag{83}$$

where $X' \sim \mathbb{P}_{X_1}$ and $X'' \sim \mathbb{P}_{X_2}$ are independent copies of X_1 and X_2 respectively.

To build DoEs based on MMD, suppose that we have a target probability distribution $\mathbb{P}_X$ given as a sample, *i.e.*,

$$\mathbb{P}_X = \frac{1}{N} \sum_{i=1}^{N} \delta_{x_i}, \tag{84}$$

where $(x_i)_{i=1,2,\ldots,N}$ is the given sample of size N, δ_x is the mass function at x. Typically N will be large.

We want to find a probability distribution $\mathbb{P}_{X_1}$ which approximates $\mathbb{P}_X$, in the sense of the MMD, with the following form:

$$\mathbb{P}_{X_1} = \frac{1}{n_{DoE}} \sum_{j=1}^{n_{DoE}} \delta_{x_j^1}, \;\; x_j^1 \in X_1, \tag{85}$$

where n_{DoE} is the number of required sample points and is strictly smaller than N. One can choose the $x_j^1, j = 1, 2, \ldots, nDoE$ so that $\mathbb{P}_{X_1}$ represents the target probability distribution $\mathbb{P}_X$.

To find $(x_j^1)_{j=1,2,\dots,n_{DOE}}$ we have to solve the following optimization problem:

$$\min_{x_1^1,x_2^1,\dots,x_{n_{DOE}}^1} MMD^2\left(\frac{1}{n_{DOE}}\sum_{j=1}^{n_{DOE}}\delta_{x_j^1}, \frac{1}{N}\sum_{i=1}^{N}\delta_{x_i}\right). \tag{86}$$

If $nDoE$ is large, the optimization problem might be difficult to solve, since it is a nonlinear and non-convex problem. A possible work-around is to use a greedy approach where we optimize x_1^1 first, then x_2^1, and so on. In kernel-herding [22], a greedy sequential algorithm is proposed, which consists of the following steps:

$$x_1^* = \text{argmax}_{x\in\mathcal{U}} \frac{1}{N}\sum_{j=1}^{N} k(x,u_j),$$

$$x_{t+1}^* = \text{argmin}_{x\in\mathcal{U}} \frac{1}{t+1}\sum_{i=1}^{t} k(x,x_t^*) - \frac{1}{N}\sum_{j=1}^{N} k(x,u_j).$$

From a computational perspective, in this case we only have to compute $k(\cdot,\cdot)$ for the samples (x_j), so we can precompute $\frac{1}{N}\sum_{j=1}^{N} k(x^1,x_j)$ for all $x^1 \in \{x_1,x_2,\dots,x_N\}$.

Regarding the choice of the kernel, we assume that (x_j) are continuous variables that lie in $\mathbb{R}^m$; we have to specify a kernel function $k(.,.)$;

$$\begin{aligned} k :&\mathbb{R}^m \times \mathbb{R}^m \longrightarrow \mathbb{R} \\ &(x,z) \longrightarrow k(x,z) \end{aligned} \tag{87}$$

Starting from a kernel $\tilde{k} : \mathbb{R}\times\mathbb{R} \longrightarrow \mathbb{R}$ defined in dimension 1, we can typically define the kernel in dimension m by two ways; either “isotropic":

$$k(x,x^1) = \tilde{k}(\|x-x^1\|),$$

or by “tensorization":

$$k(x,x^1) = \prod_{l=1}^{m} \tilde{k}(x^l,x^{1l}).$$

We use tensorization to construct the multidimentional kernel in our case.

Besides the kernel-herding approach that tries to minimize MMD, the Kernel Stein Discrepancy (KSD) [96] is also a tool to measure MMD. The main difference between these two approaches is that the greedy algorithm (see Algorithm 7 above) requires that the target distribution μ is

Algorithm 7: Greedy algorithm for continuous-variable problems

Input: $(x_j)_{j=1,2,\dots,N}, n_{DoE}$, and a p.d kernel k

- Precompute

$$\frac{1}{N}\sum_{j=1}^{N} k(x^1, x_j), \quad \forall x^1 \in (x_j).$$

- Compute

$$x_1^{1*} = \text{argmax}_{x^1 \in X_1} \frac{1}{N}\sum_{j=1}^{N} k(x^1, x_j). \tag{88}$$

- For $t = 1, 2, \dots, n_{DoE}$:

$$x_{t+1}^{1*} = \text{argmin}_{x^1 \in \{x_1, x_2, \dots, x_N\}} \frac{1}{t+1}\sum_{i=1}^{t} k(x^1, x_t^{1*}) - \frac{1}{N}\sum_{j=1}^{N} k(x^1, x_j). \tag{89}$$

Return $x_1^{1*}, x_2^{1*}, \dots, x_{n_{DoE}}^{1*}$

either explicit or can be easily approximated by sampling, whereas KSD can be used in the case where μ is not explicit, such as when it arises as an intractable posterior in a Bayesian context. Thus, the two methods have the same goal, but a different way of approach. As mentioned in the introduction section, there exists an extended KSD measure approach for discrete space with a new choice of "discrete" Stein operation [112], which is considered as a state-of-the-art method. However, in our case, when the experimental region is given, it is possible to generate a target distribution by uniformly sampling points over the domains. That is why we decided to use kernel herding in this context.

Remark that to apply the kernel-herding algorithm, the kernel to be used must necessarily be positive definite.

Besides, we can count **random design** which is one of the simplest way to sample points. It generates points randomly in the design space. This type of design has low uniformity, *i.e.,* design points are not uniformly spread over the domain, and it is possible to have duplicated points.

There are model-based DOEs which assume the form of the function to evaluate based on the correlation definition, also known as *statistical designs*. Some popular methods in this class can be counted such as D-optimality, I-optimality and G-optimality designs, [105]. Nevertheless, in this framework, we do not concentrate on this class of method so we only list them here for the readers to have a larger view of DOEs.

8 Design of experiments for mixed continuous and discrete variables problems

Contents

1 Introduction to mixed continuous and discrete variables DoEs **99**
2 Straightforward transformation of a continuous DoE into a mixed DoE **100**
3 Greedy-MDS: kernel herding and multi-dimensional scaling **101**
4 Adapted-Greedy: directly using a kernel defined in the mixed space **103**
5 Numerical experiments **105**
5.1 Design of experiments for a surrogate model with continuous, integer and categorical variables 106
5.2 Design of experiments for cyclic-symmetry problems involving continuous and binary variables 107
5.3 Design of experiments for times series 112
6 Conclusions **115**

In this section, we focus on the specific case of models with *mixed* experimental region defined as

$$\mathcal{D} = \{z = (x, y) \in \mathbb{R}^m \times \mathbb{I}^n\},$$

where $x \in \mathbb{R}^m$, and $y \in \mathbb{I}^n$ are the continuous and discrete variables, respectively, and where $\mathbb{I}$ denotes the discrete space.

1 Introduction to mixed continuous and discrete variables DoEs

For handling mixed DoEs, the first proposed approaches mainly consisted on simple extensions of continuous LHS, either by randomly discretizing continuous values [67] or by rounding continuous DoEs to obtain feasible integer candidates [30, 75]. These techniques can recover the integer domain but may destroy the desirable properties of the original DoEs, or even worse, they may generate identical points which must be avoided. On the other hand, it is possible to

generate independently a LHS for every given discrete possible values [92] or use a so-called Sliced LHS, with LHS for the continuous factor which is sliced into groups of smaller LHS designs associated with different discrete levels [88]. Although popular, these last two approaches usually require a very large number of samples when the number of discrete variables increases. A recent work [83] proposes to sample continuous variables through a single continuous LHS, while the discrete variables are obtained by randomly assigning an equal number of data samples to each discrete value. A different line of work mainly studied in the machine learning research area relies on the approximation of probability measures with an empirical probability distribution supported by a small number of points, also called *quantization* [46]. Interestingly, this point of view will be the main ingredient of our proposal.

More precisely, we will focus on the framework of kernel-embedding of probability distributions [94], which offers mathematical tools to define distances between probability distributions which can be very efficiently computed. It is then straightforward to recast the problem of finding a DoE with good space-filling properties as an optimization problem, where one wants to find the empirical probability distribution (the DoE) which is as close as possible to a target probability measure (for example the uniform measure on the feasible experimental region). For continuous probability measures, this point of view has already been proposed with the name of *kernel herding* [22] or support points [65]. Note that the target distribution here can either be given explicitly or as an empirical distribution itself but with a large number of points. Here, our goal is to extend this framework to the mixed case.

This chapter is organized as follows. First, we present in Section 2 a naive extension of continuous DoEs which will serve as a baseline for numerical comparisons. In Sections 3 and 4 two novel methods based on kernel-embedding are introduced and discussed with their respective advantages and imitations. Extensive numerical experiments are finally conducted in Section 6, where we show that on a variety of problems the kernel point of view clearly surpasses standard approaches.

2 Straightforward transformation of a continuous DoE into a mixed DoE

Recall that our experimental mixed region is

$$\mathcal{D} = \{z = (x, y) \in \mathbb{R}^m \times \mathbb{I}^n\}. \tag{90}$$

where $x \in \mathbb{R}^m$, $y \in \mathbb{I}^n$ are the continuous and discrete variables, respectively, and $\mathbb{I}$ denotes the discrete space. Our goal is to select n_{DoE} representative points from D. Inspired from

previous works [102], [103]), we first propose a straightforward algorithm which is based on the projection from continuous space to integers representing the distinct indices of discrete variables. We first generate a DoE for the discrete variables as detailed in the pseudo-code given in Algorithm 8.

Algorithm 8: Projected DoE for discrete variables

Input: $n, nDoE$

0. Pre-processing

- Compute all n_{levels} distinct discrete arrangements of size n and arrange them in the matrix L
- Compute the weight, $w = \dfrac{1}{n_{levels}}$

1. Main computation

- Build a continuous DoE of size n_{DoE} in dimension 1: $U = \{u_1, \ldots, u_{n_{DoE}}\}$
- Assign discrete values according to the following rule:
  if $u_i \in ((j-1)w, jw], i = 1, \ldots, n_{DoE}, j = 1, \ldots, n_{levels}$ then assign the $j-$th row of L to y_i

Return $y_1, \ldots, y_{n_{DoE}}$

The choice of the method to generate the continuous DoE is left to the user, but for the sake of having a uniform distribution in the discrete space, it is recommended to either use a low discrepancy method such as Sobol sequences, Halton sequences or others (see details in [41]), or standard space-filling DoEs such as the minimax or maximin ones for example [67, 87].

Remark that this adaptation focuses on picking distinct discrete candidates. If it yields badly-distributed points (this will happen if the continuous DoE is of poor quality), then the algorithm can post process duplicated discrete points (which need to be avoided). Algorithm 8 only provides the DoE for discrete variables: in order to obtain the full DoE in mixed space, the continuous parts are generated separately by another continuous DoE and added to the output of Algorithm 8. In the numerical comparisons of Section 5, we shall use a Sobol sequence and a classical LHS for both continuous DoEs.

3 Greedy-MDS: kernel herding and multi-dimensional scaling

In this section, we propose a first extension of the previous greedy algorithm to mixed continuous and discrete space by using Multi-Dimensional Scaling (MDS). The main idea is to apply MDS to build a continuous encoding of the discrete variables.

Continuous encoding of discrete variables is a standard way to handle mixed variables in regression or classification problems. Successful strategies rely on a data-driven approach, where the encoding is optimized during the supervised learning task [47, 113]. However in our unsupervised setting, such approaches cannot be envisioned. In order to circumvent this limitation, we take a different road by assuming that an user-defined distance in the discrete space characterizing any prior information is available. Such a distance can include symmetry properties such as in the numerical experiments presented below, or hand-crafted correlation given by experts of the phenomenon under study. Once this distance is available, it can be used to compute all the pairwise distances between the points from the target distribution u_i.

MDS can then be applied to this distance matrix. Indeed, MDS is a technique that creates a map displaying the relative positions of a number of objects, given only a table of distances or dissimilarities between them. In another words, MDS translates the information about the pairwise distances among a set of N objects (normally in a high dimensional space) into a configuration of N points mapped into a smaller dimensional space preserving the distances as much as possible. Suppose that we have a set of N points $\{u_1, u_2, \dots, u_N\}$ for which we know all the pairwise distances $d_{ij} = d(u_i, u_j)$. The output of MDS is a new set of points $v_1, v_2, \dots, v_N$ in a Euclidean space such that their Euclidean distance approximates d_{ij}, *i.e.*, $d_{ij} \approx \hat{d}_{ij} = \|v_i - v_j\|$. The dimension of this Euclidean space is chosen by the user, with a maximum equal to $N - 1$. To measure how well the original set of data are represented by the MDS model, it is typical to use the *goodness-of-fit* values, or *stress* values [59], based on the differences between the actual distances and their predicted values:

$$stress = \sqrt{\frac{\sum (d_{ij} - \hat{d}_{ij})^2}{\sum d_{ij}^2}}. \tag{91}$$

Based on the stress values, [59] asserts how the MDS model performs: if $stress = 0$ the MDS model is said to be *perfect*, if $stress = 0.025$ the MDS model is said to be *excellent*, if $stress = 0.05$ the MDS model is said to be *good*, if $stress = 0.1$ the MDS model is said to be *fair*, and finally if $stress = 0.2$ the MDS model is said to be *poor*.

Our proposal, called greedy-MDS, is then to apply a preliminary MDS step for all the discrete variables $Y = \{y_1, y_2, \dots, y_N\}$ in the large target sample for which we have all the pairwise distances defined by the user. After this step, we have a continuous encoding $V = \{v_1, v_2, \dots, v_N\}$ of the discrete variables, which can be stacked with the continuous variables $X = \{x_1, x_2, \dots, x_N\}$. This gives a new representation of the large target sample with only continuous values. Then, we use kernel-herding from Algorithm 7 with a standard continuous kernel to obtain a small subset of points which represents as well as possible the target distribution. The last step is finally to invert the encoding: this is an easy task, since by bijection we exactly know the correspondence

between the original discrete values and their encoding. A pseudo-code is given in Algorithm 9 below.

Algorithm 9: Greedy-MDS algorithm.

Input: $nDoE, d, Y$

1. Pre-processing step
   - Compute d, the matrix of distances $d_{ij} = d(y_i, y_j)$ for all $y_i, y_j \in Y$.
   - MDS step: apply MDS to d, obtain accordingly continuous points, $u_1, \ldots, u_N$
   - Stacking with continuous variables: for each $u_i, i = 1, \ldots, N$, add a large number of continuous values, save in matrix R
2. Greedy algorithm Algorithm 7 with inputs $R, nDoE$.
3. Retrieve the original values by inverting MDS

A strong potential limitation of Greedy-MDS lies in the choice of the user-defined distance. Indeed, very poor distance reconstruction performance can occur in practice, meaning that MDS cannot find an encoding which is able to well preserve the pairwise distances even with a maximum dimension of $N - 1$. To circumvent this issue, we observed numerically that in practice MDS has a much better reconstruction performance if instead of relying on a user-defined distance, we employ a user-defined kernel k for discrete variables and build the corresponding kernel-induced pseudo-distance

$$d_k(u, v) = \sqrt{k(u, u) + k(v, v) - 2k(u, v)}.$$

Note however that requiring a user-defined kernel is much more demanding than just a distance, and that if a kernel is available the following approach should be preferred.

4 Adapted-Greedy: directly using a kernel defined in the mixed space

In this section, we propose an adapted greedy algorithm for mixed continuous and discrete variables problems that takes into account directly an *appropriate* mixed kernel inside the greedy algorithm 7, *appropriate* being related here to the distance and type of information that are given by the user. The first step is to build a positive definite kernel on the discrete variables from a given distance $d(\cdot, \cdot)$. The naive approach consists in generalizing the *Laplace kernel* as follows:

$$k(y, y') = e^{-\gamma d(y, y')}$$

for any values y and y' of the discrete variables. Unfortunately this does not provide a positive definite kernel in general [104]. A particular case of interest where this result holds concerns *binary* variables with the Hamming distance, $d_H(y, y')$, where the kernel

$$k_H(y, y') = e^{-\gamma d_H(y,y')} \tag{92}$$

is positive definite for any $\gamma > 0$, see [52]. For the general case, we have to follow a different road and build upon the work of [110] where they introduce the so-called *soft string kernel*. For any distance d between any variables u and v in a space $\mathcal{I}$, it is given by

$$k^{soft}(u, v) := \int_{\omega \in \mathcal{I}} p(\omega) e^{-\gamma(d(u,\omega)+d(v,\omega))} d\omega \tag{93}$$

where $p(\omega) : \mathcal{I} \to \mathbb{R}$ is a probability distribution over $\mathcal{I}$, and it can be proved that it is positive definite [110]. When $\mathcal{I}$ is a discrete space it boils down to the discrete sum

$$k^{soft}(u, v) := \sum_{\omega \in \mathcal{I}} p(\omega) e^{-\gamma(d(u,\omega)+d(v,\omega))}. \tag{94}$$

Interestingly, for a uniform distribution, this kernel is an approximation of the naive proposal above, by using well-known approximation results for the *LogSumExp function (LSE)*. More precisely, the LSE function reads

$$\text{LSE}(x_1, x_2, \ldots, x_n) = \log(e^{x_1} + e^{x_2} + \ldots + e^{x_n})$$

and is an approximation to the maximum $\max(x_1, \ldots, x_n)$ [60]. Consequently, we also have

$$\min(x_1, \ldots, x_n) \approx -\log(e^{-x_1} + e^{-x_2} + \ldots + e^{-x_n}).$$

Now if p is a uniform distribution on $\mathcal{I}$, meaning that $\forall \omega \in \mathcal{I},\ p(w) = p_0$ the soft string kernel simplifies as follows

$$\begin{aligned} k^{soft}(u, v) &= \sum_{\omega_i \in \mathcal{I}} p_0\, e^{-\gamma(d(u,\omega_i)+d(v,\omega_i))}, \\ &= \sum_{\omega_i \in \mathcal{I}} e^{(-\gamma\eta_i + \log(p_0))}, \end{aligned} \tag{95}$$

with $\eta_i = d(u, \omega_i) + d(v, \omega_i)$ and applying $-\log$ to both sides leads to

$$\begin{aligned} -\log(k^{soft}(u, v)) &= -\log\left(\sum_{\omega_i \in \mathcal{I}} e^{(-\gamma\eta_i + \log(p_0))}\right) \\ &\approx \min_i(\gamma\eta_i - \log(p_0)) \end{aligned}$$

using the approximation of the minimum given above. From the distance triangle inequality we also have $\eta_i \geqslant d(u,v)$, which means that $\min_i(\gamma\eta_i - \log(p_0)) = \gamma d(u,v) - \log(p_0)$. This finally gives

$$\begin{aligned} k^{soft}(u,v) &= e^{\log(k^{soft}(u,v))} \\ &\approx e^{-(\gamma d(u,v) - \log(p_0))} \\ &= p_0\, e^{-\gamma d(u,v)}. \end{aligned}$$

This result can help interpret k^{soft} as a positive-definite approximation of the Laplace kernel where we would inset the distance d.

Combining one of the above kernels (92) or (94) for the discrete variables and any kernel k_{cont} for the continuous variables is finally straightforward with tensorization, and leads to a mixed kernel that can directly be used in kernel herding.

Extension beyond mixed variables problems.

Interestingly, the Adapted-Greedy approach presented before is generic, in the sense that we discussed building a kernel adapted for discrete variables, but the same principle can be applied to any other type of variables provided the user can provide a kernel or a distance. A prominent example in computer experiments involves time series variables, for which it is usually a challenge to build a space-filling DoEs. Some previous works are for example based on decomposition of functional data on orthogonal bases such as in [78], but here we can directly make use of the literature on kernels for time-series data. We can for example propose a normalized kernel between two time series $u(t)$ and $v(t)$ involving the recently proposed *global alignment kernel* [31]:

$$K^{GAK}(u(t),v(t)) = K(u(t),v(t)) - \frac{1}{2}(K(u(t),u(t)) + K(v(t),v(t))), \tag{96}$$

where K is the global alignment kernel for same length time series

$$K(u(t),v(t)) = \prod_{i=1}^{|} e^{-\phi_\sigma(u(t_i),v(t_i))}, \tag{97}$$

with $\phi_\sigma(u(t_i),v(t_i)) = \frac{1}{2\sigma^2}\|u(t_i)-v(t_i)\|^2 + \log(2 - e^{-\frac{\|u(t_i)-v(t_i)\|^2}{2\sigma^2}})$. We will see in Section 5 how it performs numerically on a time series example.

5 Numerical experiments

The aim of this section is to demonstrate the large range of applicability of the proposed design of experiment methods. The first application is the surrogate modelling of the simulated

performances of an electric engine with respect to design parameters of the rotor component. This application deals with continuous, integer and categorical variables. Then, we consider the case of cyclic-symmetry problems, with continuous and binary variables, motivated by an application of optimal design of the turbomachine of an helicopter engine [102]. For these specific problems, we address two different operational objectives: (a) surrogate modelling illustrated on various benchmark functions and (b) the choice of initial points for the shape optimization problem of the turbomachine. Finally, our DoE methods are applied to two examples involving time series.

5.1 Design of experiments for a surrogate model with continuous, integer and categorical variables

The first application relates to the optimal design of an electric engine with respect to the rotor component. The purpose of this study is to choose a set of values of the design variables that will be used to perform simulations of the engine operation in order to build surrogate models of the maximal power and of the maximal torque of the engine. The simulations are performed with the FEMM simulator[1] designed for electromagnetic problems. The design variables are composed of one continuous variables (the rotor length), 2 integer variables (number of wires and number of coils) with respectively 4 and 11 feasible values and one categorical variable that characterizes the type of rotor geometry (with 16 possible types of rotor). For this problem, we used a tensorized kernel composed of a Gaussian kernel for continuous variables and the kernel (92) based on the Hamming distance applied to mixed binary variables resulting from the encoding of integer and categorical variables. Here we compare designs of experiments built with *adapted-greedy* approach to the projected LHS (algorithm 8) with a number of points ranging from 20 to 100 points. Surrogate models based on Gaussian processes and adapted kernel for mixed continuous and categorical variables ([76], [88]) are built from the simulations performed on these designs. The accuracy of the obtained surrogate models is assessed by

$$1 - Q^2(F, \hat{F}) = \frac{\sum_{i=1}^{N_v}(F_i - \hat{F}_i)^2}{\sum_{i=1}^{N}(F_i - \bar{F})^2}, \tag{98}$$

where Q^2, the predictive accuracy coefficient, is computed on a new set of validation points of size N_v, $N = N_v + n_{DOE}$ is the total number of simulations including the points of the DoE, F is the vector of simulated values, $\hat{F}$ is the vector of predicted values (via the surrogate model) and $\hat{F}$ is the mean of F. In our experiments, we choose $N_v = 300$ validation points obtained by a projected LHS method. In Figure 21, we study the predictivity of each model built from the designs of experiments for the two simulator outputs of interest, the maximal torque and the maximal power of the electric machine. We achieve a better predictivity (smaller values of

[1] https://www.femm.info/wiki/HomePage

$1 - Q^2$) of the surrogate models for the *adapted-greedy* method compared to projected LHS for both responses of interest and for all the design sizes.

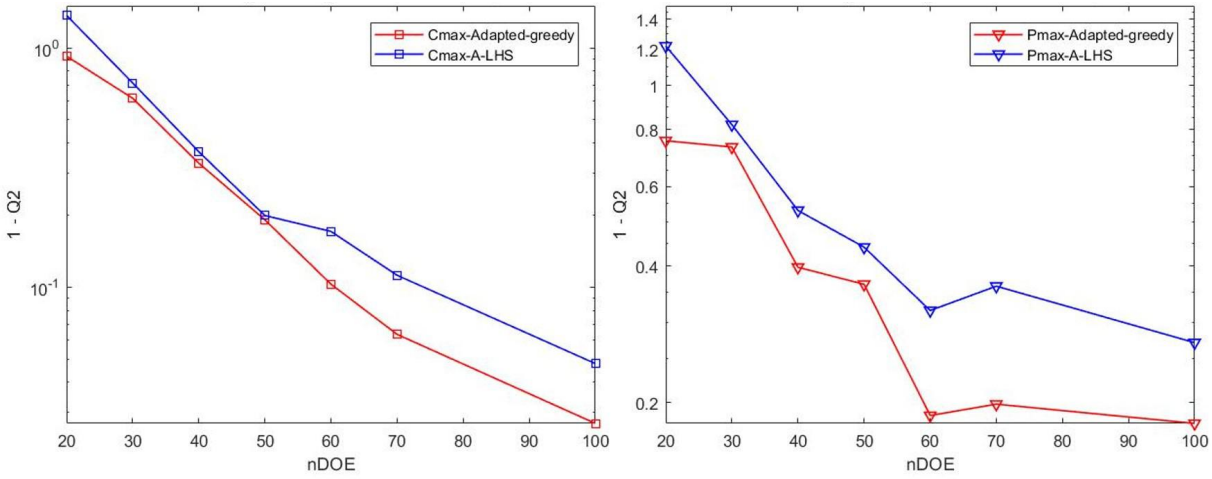

Fig. 21.: Accuracy of the surrogate models of the maximal torque (left) and the maximal power (right) built with designs of experiments obtained by the projected LHS method (in blue) and by the *adapted-greedy* algorithm (in red). The size of the designs, n_{DOE}, ranges from 20 to 100.

5.2 Design of experiments for cyclic-symmetry problems involving continuous and binary variables

The study of such cyclic-symmetry problems is driven by an application of optimal design for turbine blades in an helicopter engine [74], which involves mixed continuous and binary input variables.

We first apply our novel DoE methods on a set of benchmark functions (all with cyclic symmetry) with the aim of estimating the expectation of the actual functions. The second application presented in this section focuses on the choice of the initial points of an optimization method.

For both applications, the functions of interest are not affected by cyclic permutations of the vector of binary variables:

$$f(x, y') = f(x, y),\ \forall (x, y) \in \mathbb{R}^m \times \{0, 1\}^n \qquad (99)$$
$$\forall y' \in \{0, 1\}^n \text{ such that } \exists i \in \{1, 2, \ldots, n\}\ Rot^i(y') = y,$$

with $Rot^i(y)$ is the rotation of y by i positions.
For the *MDS-greedy* and the *adapted-greedy* methods, we use a tensorized kernel with, for the binary variables, the *soft kernel* (94) with the *necklace distance* ([102])

$$d_{neck}(y, y') = \min_{i=1,2,\dots,n} d_H(y, Rot^i(y')), \tag{100}$$

where d_H denotes the Hamming distance. A *necklace* of binary variables $y \in \{0,1\}^n$ is defined as the equivalence class of binary vectors considering all rotations as equivalent vectors (analogy with a necklace of n beads of 2 colors represented by 0 and 1 values): $\mathcal{N}_y = \{Rot^i(y)\ \forall i \in \{1,2,\dots,n\}\}$.

Results for expectation estimation

The benchmark consists of 6 different functions chosen from [36, 64, 80, 22], listed in table 6. These functions were set up originally for continuous optimization purpose. Therefore, we transform them into mixed binary functions with cyclic symmetry following the methodology of [102]. The problem dimensions are listed in Table 6 with a number of continuous variables ranging from 1 to 10 and a number of binary variables from 4 to 7 corresponding to 6 to 20 distinct necklaces.

Problem name	Dimension	# distinct binary vectors (necklaces)	Reference
$\sin(\|\|x\|\|)$	2×7	20	[22]
Wong2	10×4	6	[64]
Branin	1×7	20	[36]
Hartman3	3×6	14	[36]
Perm6	5×5	10	[80]
Perm8	7×5	20	[80]

Tab. 6.: List of benchmark functions.

DoE are built with the methods introduced in Section 3: two projected designs (8) based on LHS and Sobol sequence and our two methods based on kernel embeddings: *greedy-MDS* and *adapted-greedy*. These results are compared with random sampling and standard LHS with rounded values for binary variables. The empirical means of the benchmark functions for each necklace are computed from the points of the DoE and compared with the empirical means computed with a very large set of points obtained by Monte-Carlo sampling (10^4 evaluations).

For each DoE $\delta_{n_{DoE}}$, the accuracy of the estimations is rated by the root mean squared errors of the function expectations at the different necklaces

$$\text{Error}(\delta_{n_{DoE}}) = \sqrt{\frac{1}{n_{neck}} \sum_{h=1}^{n_{neck}} (F_h^{ref} - F_h^{DoE})^2}, \tag{101}$$

where n_{neck} is the number of necklaces, F_h^{ref} and F_h^{DoE} are respectively the empirical expectations of the function for the Monte-Carlo sampling points and the points of the DoE for the necklace h.

The results of 20 repeated runs of the 6 methods with a design size $n_{DOE} = k\, n_{neck}$, k ranging from 1 to 27, are shown in Figure 22. Figure 23 provides the distributions of estimation errors for each necklace obtained by one run of the 6 methods on Branin test case with a design size of 100 points ($k = 5$). We observe on both Figures that the approximation errors are much smaller for *greedy-MDS* and *adapted-greedy* methods compared to the four others whereas the two projected methods based on LHS and Sobol sequences provide slightly better global results on the benchmark compared to standard methods with rounded values (Figure 22).

Results for initial design of optimization

The aim of the turbine application from [74, 102] is the design of the turbine blades of an helicopter engine in order to minimize the vibrations of the compressor. This optimization problem involves one continuous parameter controlling the frequency amplitude and a vector of 12 binary variables describing the repartition of two reference blades on the turbine disk. In [102], the authors propose an adapted optimization method based on a derivative-free trust-region method that uses the *necklace* distance (100) to take into account the cyclic symmetry of the problem. In this work, the initial set of points are chosen by a LHS procedure with rounded values for binary variables. We compare here these optimization results with the results obtained with the same optimization method coupled with two types of initial DoE: the projected LHS and adapted-greedy methods. As for previous benchmark functions, the chosen kernel for the latter is a tensorized kernel of a Gaussian kernel for continuous variables and the *softmax kernel* with *necklace distance*. The size of the initial DoE is $n + m + 1 = 14$ and 100 repetitions of each DoE methods and associated optimizations are run. The results are reported in data and performance profiles in Figures 24 and 25, respectively. As explained in [73], the data and performance profiles with respect to the number of function evaluations are convenient tools to compare the performances of derivative-free optimization methods, counting the ratio of successful runs for each solver with respect to a chosen criterion, the number of simulations in our study. We

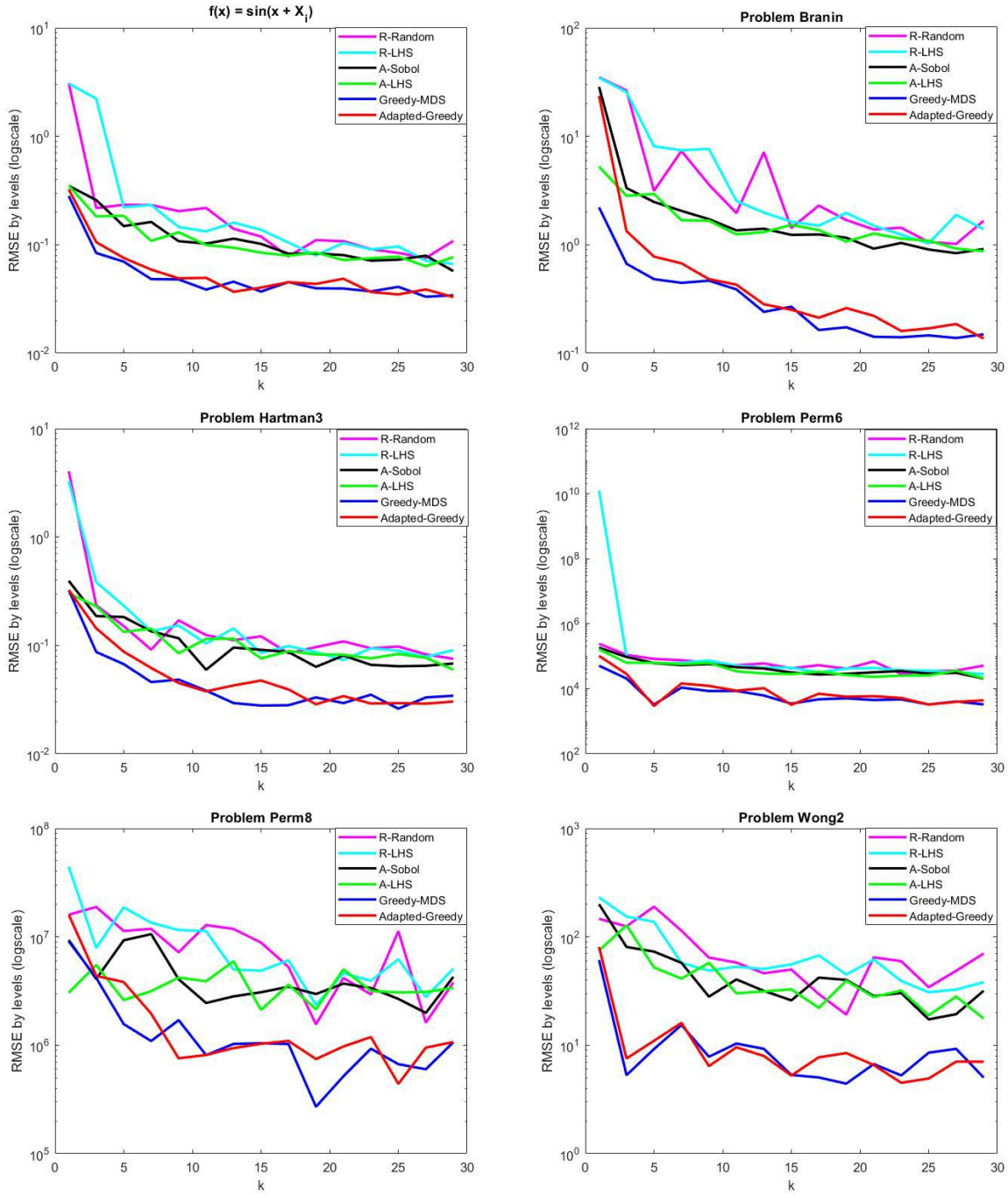

Fig. 22.: Median RMSE (in log-scale) of the estimation of benchmark function expectation obtained with 5 repetitions of the 6 methods: a random sampling and a standard LHS with rounded values for binary variables, 2 projected methods based on Sobol sequence and projected LHS, the greedy-MDS and adapted-greedy methods.

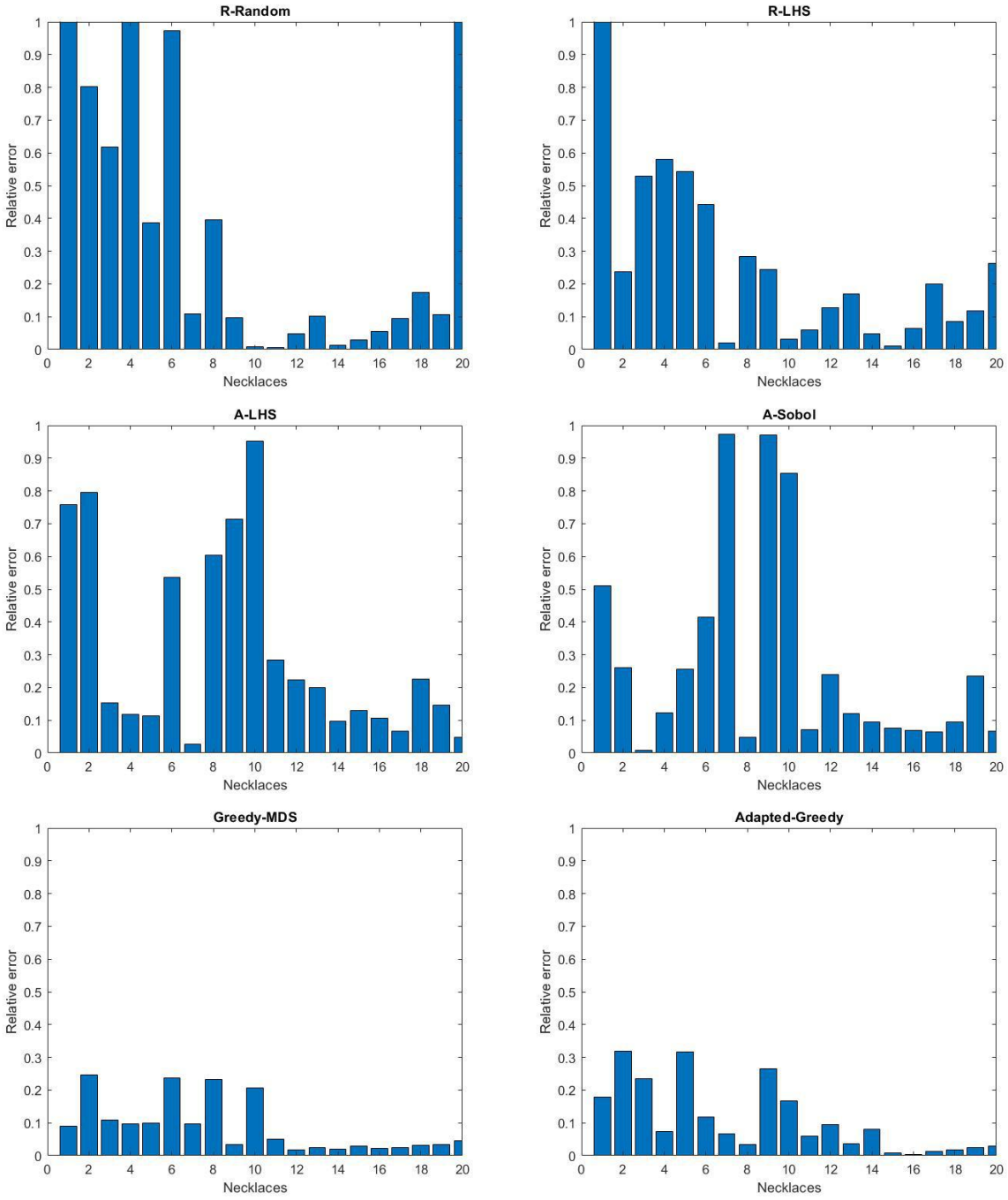

Fig. 23.: Distributions of estimation errors for each necklace of the Branin function obtained with 6 methods: a random sampling and a standard LHS with rounded values for binary variables, 2 projected methods based on Sobol sequence and projected LHS, the greedy-MDS and adapted-greedy methods. The size of the DoE is 100 points.

consider that an optimization run is successful if the best solution $\bar{z}$ obtained within a given number of function evaluations satisfies

$$f(z_0) - f(\bar{z}) \geqslant (1-\tau)(f(z_0) - f^*), \tag{102}$$

where, in the sequel, f^* denote the best function value found by any solver (or the global-minimum value, if known), z_0 is the best point of the initial point for each run and τ is the desired accuracy, a user-defined tolerance value (in our tests, $\tau = 10^{-5}$). If a run does not provide a solution that satisfies (102), we consider that it fails. The data profile displays the ratio of successful runs of each solver over the total number of runs with respect to the number of simulations (normalized by the number of variables: $n + m + 1 = 14$), whereas the performance profile displays this ratio with respect to a performance ratio that is, in our case, the ratio between the number of evaluations needed by a given method to satisfy the condition (102) and the smallest number of evaluations for all the compared methods.

Figures (24 and 25) show that the optimization method coupled with DoEs generated by projected LHS is the most robust method with 90% of successful runs, while the optimization coupled with adapted-greedy method succeeds to solve 88% of problems. The optimization method coupled with LHS with rounded values for binary variables is the least robust method with only 82% of successful runs. The optimization method coupled with adapted-greedy DoE appears to be less efficient for small budgets of simulations. For this kind of derivative-free optimization methods coupled with expensive simulators, the sizes of initial set are generally small in order to limit the number of expensive simulations. In the presented application, the size of the initial set (14) is small compared to the number of necklaces (equal to 352). This can explain the small differences between the results of the optimization method coupled with the three DoE methods: the exploration of the input space is done essentially along the optimization iterations and not during the initial phase.

5.3 Design of experiments for times series

To illustrate the ability of the proposed DoEs to be applied to kinds of variables other than the mixed discrete case, we propose in this section to apply the greedy-MDS and the adapted-greedy approaches coupled with the global alignment kernel (see (96) in Section 4) to two time series examples and compare the results with those obtained with some state-of-the-art approaches.

We consider two analytical functions with a functional input random variable $V(t)$ that is known through a sample of 200 realizations. Two cases are studied: V is either a standard Brownian motion (Figure 26) or a Max-stable process (Figure 27). The aim of these experiments is to estimate the expectation of the function with respect to the functional variable $V(t)$ with a limited

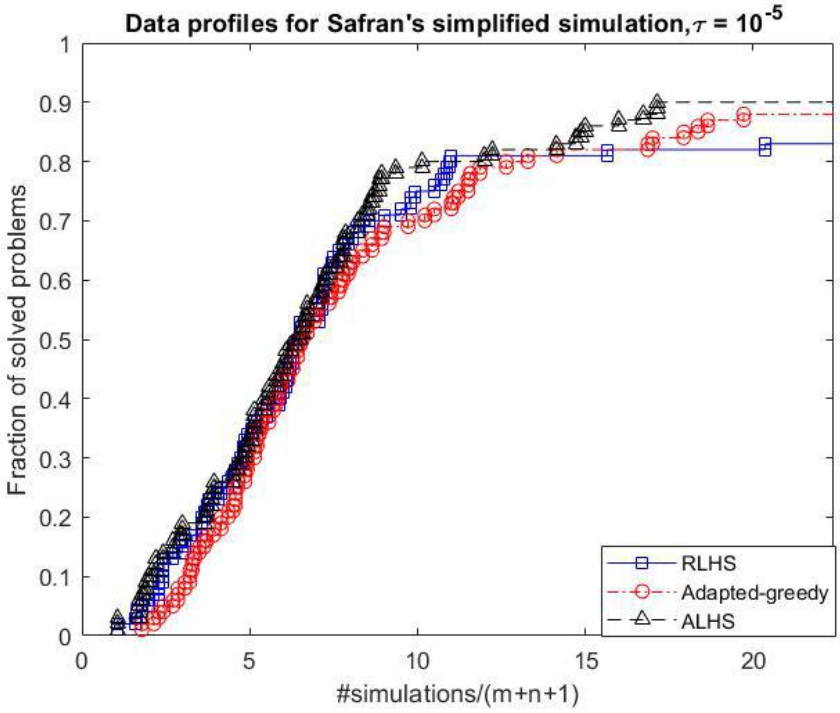

Fig. 24.: Safran's application: data profiles of the optimization runs with 100 initial DoE obtained with 3 methods: LHS with rounded values for binary variables, projected LHS, and adapted-greedy methods.

number of samples of V. To achieve this goal, [45, 37] propose a greedy functional data-driven quantization approach coupled with a functional principal component analysis (FPCA) to reduce the dimension of V to a finite (small) dimension. The impact of the number of components in FPCA on the explained variance of the functional data for two types of uncertainties (Brownian and Max-stable processes) is discussed in [38]. For the presented numerical experiments, we choose 8 components that explain respectively 97.6% and 70% of the data variance. We propose here to compare this quantization method to our two DoE methods applied directly in the functional domain with the global alignment kernel K^{GAK} introduced in Subsection 3 for time series. The greedy-MDS method relies on the associated *kernel distance* defined as

$$d^2_{GAK}(u(t), v(t)) = K^{GAK}(u(t), u(t)) + K^{GAK}(v(t), v(t)) - 2K^{GAK}(u(t), v(t)). \tag{103}$$

A standard LHS method applied in the reduced space obtained by FPCA is also evaluated. The two studied functions are defined as

$$f_1 : (x, V) \to x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7 + \int_0^T e^{V_t} dt, \tag{104}$$

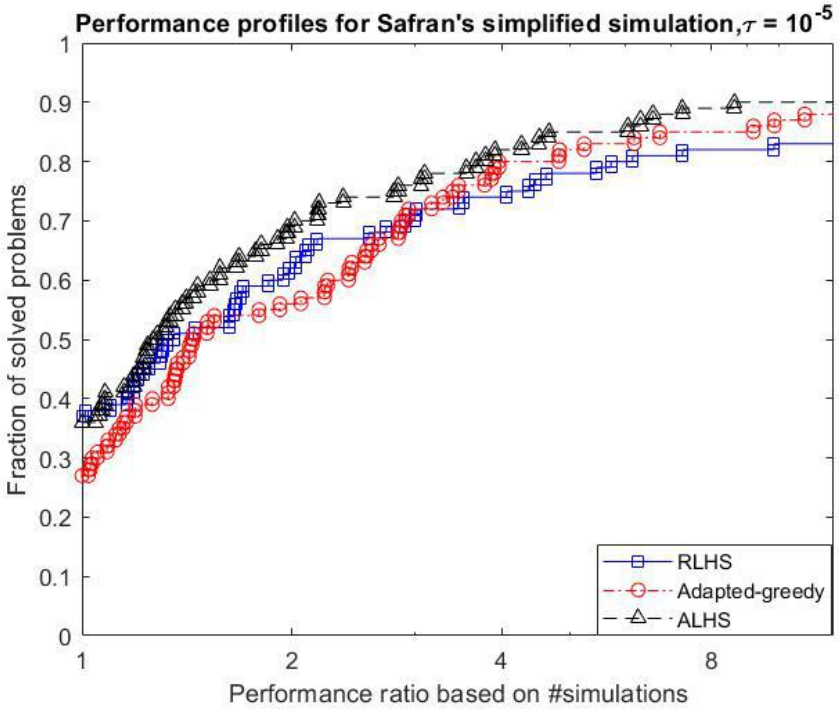

Fig. 25.: Safran's application: of the optimization runs with 100 initial DoE obtained with 3 methods: LHS with rounded values for binary variables, projected LHS, and adapted-greedy methods.

where we fix $x_1 = 50$ and $x_2 = -80$, and

$$f_2 : (x, V) \to 2x_1^2 + x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7 + \int_0^T 2\sin(V_t)dt, \tag{105}$$

where $x_1 = 50$ and $x_2 = -80$.

We apply the 4 DoE methods for the functions f_1 and f_2 and the two functional data, Brownian motion and max-stable process, with various sizes of DoE equal to $20, 40$ and 60 chosen among the 200 available realizations of V. The performances of the methods are measured in terms of accuracy and computational time in Tables 7, 8, 9, and 10. The errors are computed as the relative errors of the expectation estimations based on the obtained DoE and the estimation based on the 200 realizations. The results of LHS and quantization methods are the means of the results obtained for 50 repeated runs.

The quantization method which is dedicated to the expectation estimation gives often the best results but our two methods provide close results in terms of accuracy whereas the results of the LHS method applied in the reduced space are not as accurate in general. In terms of computational time, our two methods are more efficient than the quantization method implementation of [37].

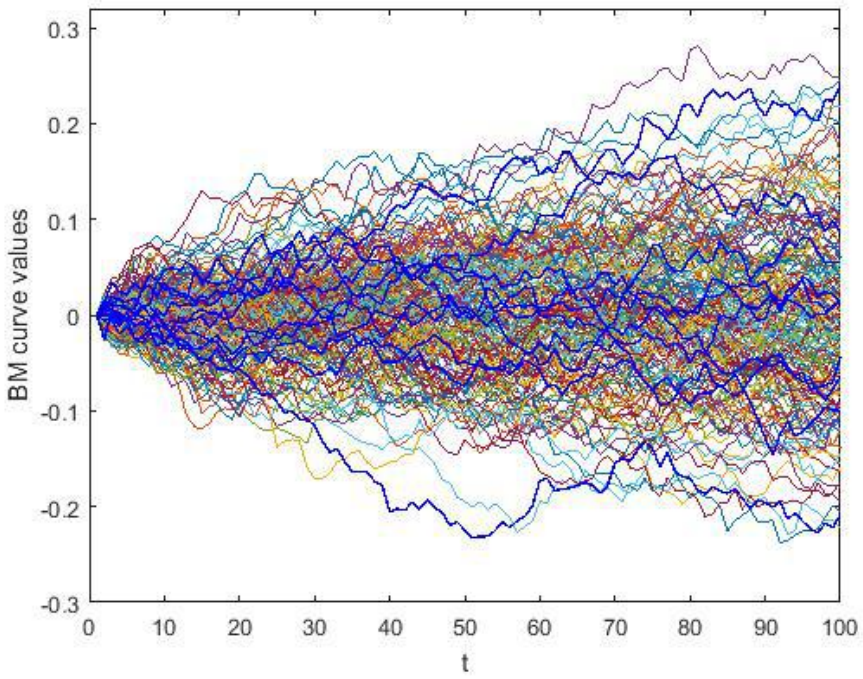

Fig. 26.: 200 realizations of a Brownian motion. The thick blue curves are the 10 curves selected by the first iterations of the adapted-greedy algorithm.

Moreover, these methods directly consider the functional variable and do not require to apply a dimension reduction method.

Tab. 7.: Computational times and relative errors of the expectation estimation obtained for the function f_1 with the functional data of Brownian motion. 4 methods are evaluated: the adapted-greedy and MDS-greedy methods coupled with the global alignment kernel, the LHS method and the quantization method coupled with the FPCA dimension reduction method. The best values among the 4 methods are indicated in bold.

	Adapted-greedy		Greedy-MDS		LHS + FPCA		Quant.+FPCA	
nDoE	time(s)	errors	time(s)	errors	time(s)	errors	time(s)	errors
20	0.04	$\mathbf{8.3.10^{-8}}$	**0.02**	$1.9.10^{-7}$	0.08	$1.6.10^{-7}$	0.36	$1.3.10^{-7}$
40	0.072	$9.3.10^{-8}$	**0.027**	$2.5.10^{-7}$	0.096	$2.9.10^{-8}$	0.94	$\mathbf{6.8^{-9}}$
60	0.109	$\mathbf{1.6.10^{-8}}$	**0.030**	$4.1.10^{-8}$	0.11	$5.7.10^{-8}$	2.04	$\mathbf{1.6^{-8}}$

6 Conclusions

The aim of this chapter is to propose design of experiments methods adapted to mixed discrete and continuous variable problems. A first proposal is the straightforward extension of continuous

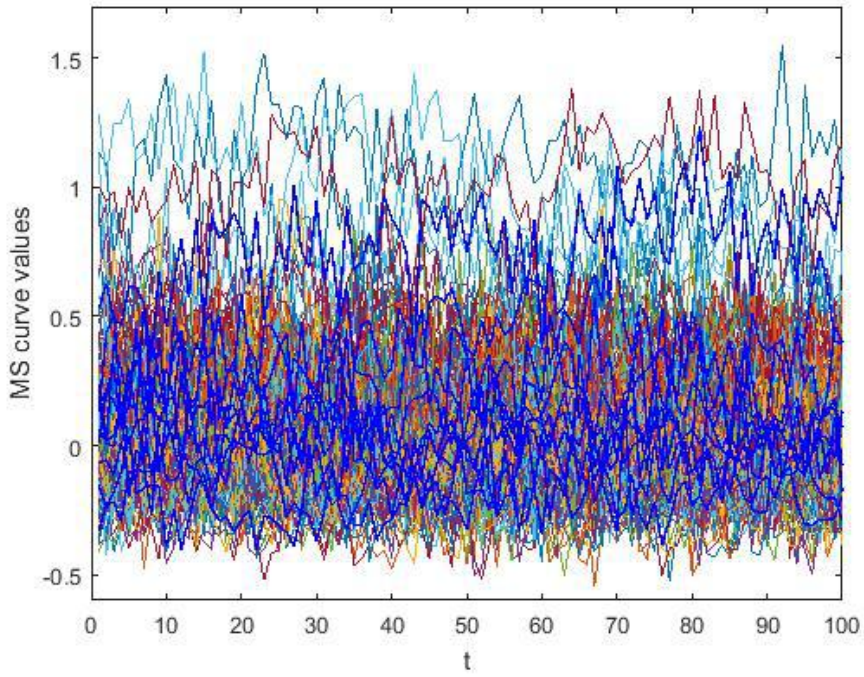

Fig. 27.: 200 realizations of a Max-stable process. The thick blue curves are the 10 curves selected by the first iterations of the adapted-greedy algorithm.

DoE techniques as low discrepancy methods or standard space filling approaches coupled with the projection from continuous space to integers representing the distinct indices of discrete variables. A second proposed approach relies on kernel-embedding of probability distributions extended to mixed continuous and discrete space, which leads to two new methods: greedy-MDS and adapted-greedy. These methods are generic and can address other types of objects than mixed discrete and continuous variables, provided that a suitable kernel is available. A variety of useful kernels are presented: in particular, the *softmax kernel* that can be built from any distance and ensures the positive-definite property and the global alignment kernel that is suited to time series.

We illustrate the performances of the proposed DoE methods in various context: building a training data set for a surrogate model handling continuous, integer and categorical variables, and for the estimation of the expectation of a function with cyclic symmetry for a vector of binary variables, choosing initial points of a mixed binary and continuous shape optimization problem for an helicopter engine, and finally DoE for time series for expectation estimation of a function that depends on a functional variable.

Future studies can be carried on with other types of objects and associated kernels (images, graphs ...) and also with different operational objectives. For instance, the exploration step

Tab. 8.: Computational times and relative errors of the expectation estimation obtained for the function f_1 with the functional data of Max-stable process. 4 methods are evaluated: the adapted-greedy and MDS-greedy methods coupled with the global alignment kernel, the LHS method and the quantization method coupled with the FPCA dimension reduction method. The best values among the 4 methods are indicated in bold.

	Adapted-greedy		Greedy-MDS		LHS + FPCA		Quant.+FPCA	
nDoE	time(s)	errors	time(s)	errors	time(s)	errors	time(s)	errors
20	0.036	$\mathbf{2.3.10^{-7}}$	**0.028**	2.10^{-6}	0.071	1.7^{-6}	0.353	$5.4.10^{-7}$
40	0.073	$1.8.10^{-6}$	**0.034**	$\mathbf{1.6.10^{-8}}$	0.09	8.10^{-7}	0.93	3.2^{-7}
60	0.11	5.10^{-7}	**0.039**	$4.1.10^{-7}$	0.11	$3.8.10^{-7}$	1.93	$\mathbf{2.10^{-7}}$

Tab. 9.: Computational times and relative errors of the expectation estimation obtained for the function f_2 with the functional data of Brownian motion. 4 methods are evaluated: the adapted-greedy and MDS-greedy methods coupled with the global alignment kernel, the LHS method and the quantization method coupled with the FPCA dimension reduction method. The best values among the 4 methods are indicated in bold.

	Adapted-greedy		Greedy-MDS		LHS + FPCA		Quant.+FPCA	
nDoE	time(s)	errors	time(s)	errors	time(s)	errors	time(s)	errors
20	0.036	3.10^{-7}	**0.022**	$6.3.10^{-7}$	0.05	$5.5.10^{-7}$	0.33	$\mathbf{2.3.10^{-7}}$
40	0.072	$2.7.10^{-7}$	**0.027**	$7.6.10^{-7}$	0.07	$1.8.10^{-7}$	0.89	$\mathbf{6.1.10^{-8}}$
60	0.11	3.10^{-8}	**0.03**	$\mathbf{1.1.10^{-8}}$	0.09	$1.1.10^{-7}$	1.9	$2.5.10^{-8}$

in global optimization methods such as surrogate optimization or direct search methods could exploit the proposed DoEs for mixed continuous and discrete problems or other type of variables provided that some prior information is available and can be associated to a kernel or a distance (as we did for cyclic symmetry).

Tab. 10.: Computational times and relative errors of the expectation estimation obtained for the function f_2 with the functional data of Max-stable process. 4 methods are evaluated: the adapted-greedy and MDS-greedy methods coupled with the global alignment kernel, the LHS method and the quantization method coupled with the FPCA dimension reduction method. The best values among the 4 methods are indicated in bold.

	Adapted-greedy		Greedy-MDS		LHS + FPCA		Quant.+FPCA	
nDoE	time(s)	errors	time(s)	errors	time(s)	errors	time(s)	errors
20	0.036	$\mathbf{5.2.10^{-7}}$	**0.028**	$1.1.10^{-6}$	0.071	$2.8.10^{-6}$	0.35	$1.6.10^{-6}$
40	0.073	$1.9.10^{-6}$	**0.034**	$1.9.10^{-6}$	0.091	1.10^{-6}	0.91	$\mathbf{8.9.10^{-7}}$
60	0.11	$\mathbf{4.10^{-7}}$	**0.04**	$2.1.10^{-6}$	0.11	$8.1.10^{-7}$	1.91	$6.3.10^{-7}$

Part III

Escaping from local minima

9 Escaping from local minima

> *Life is like riding a bicycle. To keep your balance you must keep moving.*
>
> **— Albert Eistein**

Contents

1 Augmented DoE strategies for sampling a discrete variable space **122**
2 Restart strategies **124**
2.1 Hard restart 126
2.2 Complete soft restart 127
2.3 Partial soft restart 127
3 Numerical results **127**
3.1 Results for the DS-MM function 128
3.2 Results for the Ackley function 128
4 Conclusion **130**

Global optimization, is an attractive objective for many applications dealing with complex multi-modal functions. We mentioned in Part I that DFO methods, for instance, DFOb-d_H or adapted method DFOb-d_{neck}, guarantee convergence but only toward a local minimum. As we test and compare DFOb-d_{neck} with RBFOpt (which is declared as a global method) in the previous part, we observe that for some instances, our DFO method is comparable, or even more efficient than RBFOpt with the same accuracy. Nevertheless, DFO trust-region methods may be trapped in a local minimum without exploring regions that includes a globally optimal solution. There are techniques have been used in DFO literature for escaping from local minima, for example multi-start mechanism using in RBFOpt [30] or in the work of [48], restart strategy in [21], or multistart using multi-level single linkage or maximum information from previous evaluations in GORBIT [109].

Our idea to circumvent this drawback is a restart strategy coupled with an augmented DoE strategy. Section 1 presents the principle of an augmented DoE based on the sampling of the discrete variable space. Section 2 introduces three restart strategies including two original proposals. Section 3 gives preliminary results. Section 4 concludes this third part of the thesis.

1 Augmented DoE strategies for sampling a discrete variable space

The purpose of this section is the improvement of the **standard “add-points" step** (see Chapter 3, Part I) as a diversifying strategy to help the exploration step for new values of the binary set y. As we mentioned, the current “add-points step" chooses a user-defined number of new points which only need to satisfy the trust-region and no-good cuts constraints without any requirement regarding the available simulated points and the spreading of the points. We propose here a strategy that uses the information of the available simulated points to anticipate whether we should explore more or less the continuous search space for a given binary configuration (or necklace). The proposed method is an extension of the DOE methods proposed in Part II integrating a given initial design of experiments (the available simulated points) to be completed.

Our strategy is based on the sampling method proposed in [76] for general mixed continuous and categorical variables in the Bayesian optimization context. In the context of cyclic-symmetry problems, we adapt the sampling method by considering each necklace as one category of a categorical variable, called *level* in [76].

The authors of [76] define a discrete probability distribution in the categorical space that takes into account the density of available evaluated points at each level. The discrete probability of level c_i is defined as

$$p_{k,i} = \alpha_k p^g_{k,i} + (1 - \alpha_k) p^m_{k,i}, \forall i = 1, 2, \ldots, n_{levels}, \tag{106}$$

where k is the optimization iteration counter, $\alpha_k \in [0, 1]$ is a weighting parameter, the quantity $p^g_{k,i}$ stands for the probability that the level c_i has not been fully explored (it is large when the number of simulations associated with this level is small) and is computed as:

$$p^g_{k,i} = 1 - \left(\frac{n_{k,i}}{n_k} \right)^l, \tag{107}$$

where $n_{k,i}$ and n_k are respectively the number of evaluated points in the level c_i and the total number of evaluations of the objective function over all the levels at iteration k; $l > 0$ is a user-defined parameter (that we shall set to $\frac{1}{2}$ in the implementation); the term $p^m_{k,i}$ is the probability that the level c_i has a high potential of containing the minimum of the objective function, which is approximated by:

$$p^m_{k,i} = \frac{S^R_{k,i}}{\sum_{j=1}^{n_{levels}} S^R_{k,j}}, \tag{108}$$

where $S^R_{k,i}$ is the sigmoid function calculated by:

$$S^R_{k,i} = \frac{1}{1 + e^{s_{k,i} - f_{\min}}}, \tag{109}$$

and

$$s_{k,i} = \bar{f}_{k,i} - 2\sigma_{k,i}, \tag{110}$$

with $f_{\min}, \bar{f}_{k,i}$ and $\sigma_{k,i}$ are respectively the current minimum objective-function value, the mean of the evaluated objective function values and the corresponding standard deviation in the level c_i; $p_{k,i}$ are for all levels $i = 1, 2, \ldots, n_{levels}$ which give, for each level, the probabilities that it has not been sufficiently explored. This probability distribution $p = (p_{k,1}, p_{k,2}, ..., p_{k,n_{levels}})$ is sampled to obtain a target distribution for our DOE method: MDS greedy or adapted-greedy DOE method (see Part II), completed with the available simulated points. The greedy algorithm is applied from the given set of simulated points instead of starting with an empty set as the original proposal.

We illustrate the proposed method on the toy problem introduced in [76], called DS-MM function in the sequel, with $m = 1$ continuous and $n = 5$ binary variables. Remark that the original problem is adapted to a mixed cyclic binary problem, with the following piecewise-defined function involving 8 univariate functions

$$f(x,y) = \begin{cases} \cos(3.6\pi(x-2)) + x - 1 & \text{if } y = (0,0,0,0,0), \\ 2\cos(1.1\pi e^x) - \dfrac{x}{2} + 2, & \text{if } y = (0,0,0,0,1), \\ \dfrac{-x^5}{2} + 1, & \text{if } y = (0,0,0,1,1), \\ -\cos(\dfrac{5\pi}{2}x)^2\sqrt{x} - \dfrac{\log(x+0.5)}{2} - 1.3, & \text{if } y = (0,0,1,0,1), \\ \dfrac{-x^2}{2}, & \text{if } y = (0,0,1,1,1), \\ 2\cos(\dfrac{\pi}{4}e^{-x^4})^2 - \dfrac{x}{2} + 1, & \text{if } y = (0,1,0,1,1), \\ x\cos(3.4\pi x) - \dfrac{x}{2} + 1, & \text{if } y = (0,1,1,1,1), \\ x\left(-\cos(\dfrac{7\pi}{2}x - \dfrac{x}{2})\right) + 2, & \text{if } y = (1,1,1,1,1). \end{cases} \tag{111}$$

We shall use this function as a test function for optimization in Section 3. As we can see in Figure 28, this function has several local minima and some of the individual one-dimension functions (corresponding to each level) are strongly correlated or anti-correlated. The global minimum is marked as a circle point in the individual function associated to level 4 (noted $OF4$). Finally, we observe that four individual functions have the local minima OF1, OF4, OF7, OF8, whereas the four others (OF2, OF3, OF5, OF6) are almost flat. As a result, we would expect that the augmented can put more points in these levels.

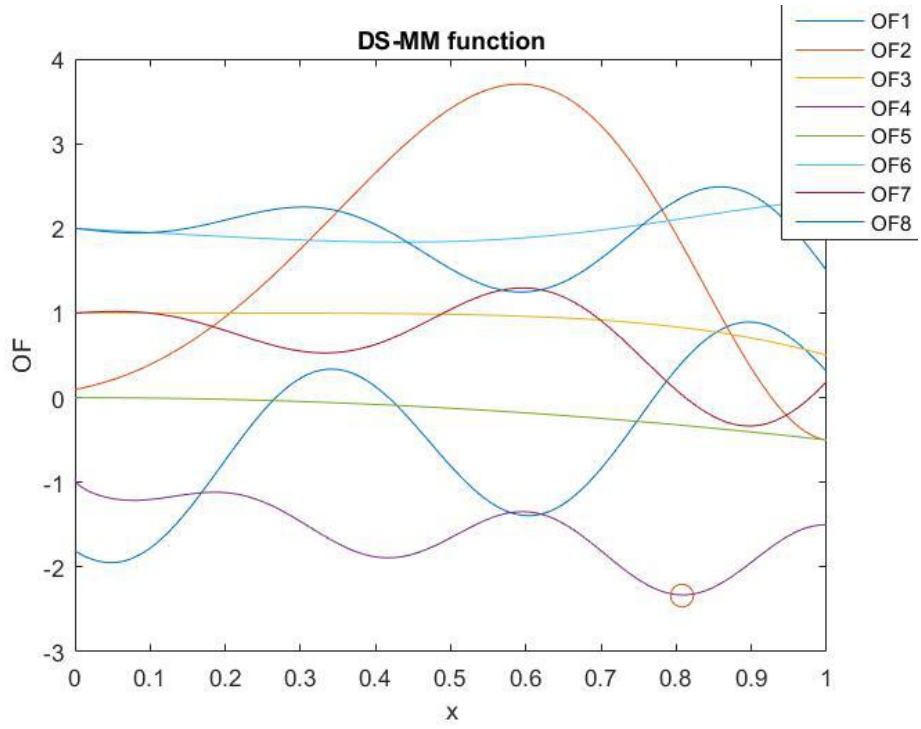

Fig. 28.: The eight individual one-dimensional functions for DS-MM example. The global minimum is located at $x = 0.808, y = (0, 0, 1, 0, 1)$, the objective-function value is -2.329.

The augmented DoE strategy can be compared to a uniform sampling (as used in the standard add-point step of DFOb method) in Figures 29 and 30. In both figures, we fix the set of available evaluated points (10 points) represented by the green points. The added points are represented by blue stars.

We realize two experiments: one with five adding points (see Figure 29), the other one with ten points (see Figure 30). On the left-hand side, the points provided by the standard procedure are displayed, whereas on the right-hand side points added by the augmented DoE algorithm are plotted. We observe, for two experiments, that the new procedure proposes points on the levels that were not already explored and with a minimization potential learnt from the available simulated points.

2 Restart strategies

This section presents different restart strategies that allow the DFO trust-region methods DFOb to escape from local minima based on ideas proposed in [48, 77, 21] for continuous derivative-free trust-region method. In [48, 77], the authors use a mechanism to restart the algorithm from completely new initial points whenever certain conditions are satisfied. Beside this *hard-*

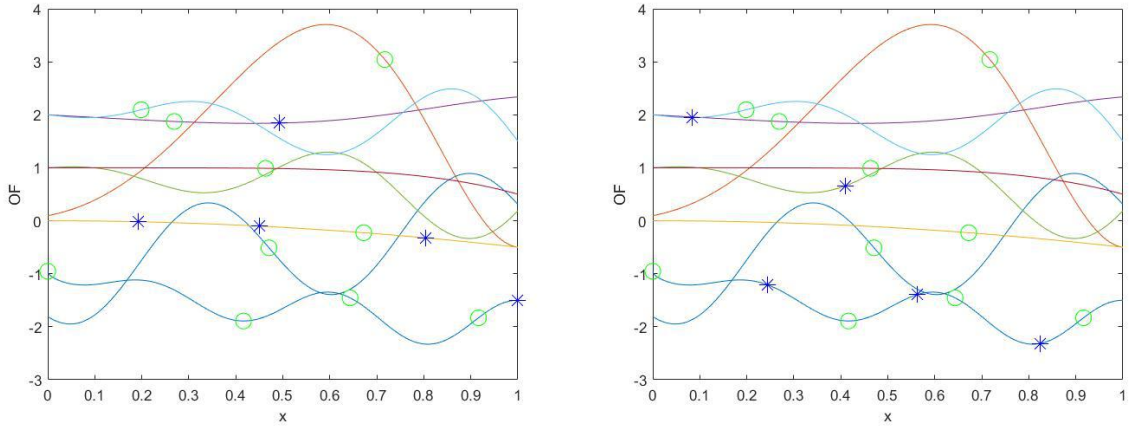

Fig. 29.: Five points (blue stars) to complete the set of ten evaluated points (green circles) by the standard procedure (left) and by the augmented DoE method (right).

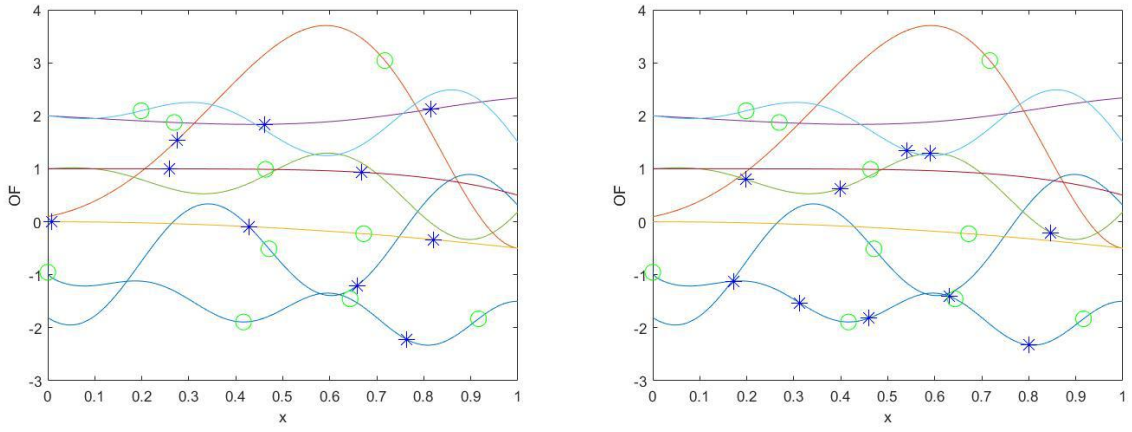

Fig. 30.: Ten points (blue stars) to complete the set of ten evaluated points (green circles) by the standard procedure (left) and by the augmented DoE method (right).

restart strategy, the authors in [21] propose an adapted *soft restart* which allows one to use the information from the evaluated points and restart with the algorithm scale of $1.1\Delta_0$: the algorithm starts with a trust-region radius set to $1.1\Delta_0$, where Δ_0 is the initial radius.

Another attractive track is the *multistart* strategy, proposed for example in the method GORBIT [108]. The principle of this tactic is to start the algorithm from different starting points in parallel.

In our work, we integrate the above *hard restart* in the DFOb method, and we propose two other restart methods: *complete soft restart* and *partial soft restart*. The next three subsections will give more details on the three approaches.

We choose two conditions proposed in [21] to trigger the restart strategies, we involve restart when either:

- The trust-region radius at the current iteration k, Δ_k, is small,
- The decrease of the objective function over consecutive iterations is too small.

Remark that we also use the re-initialized value of the trust-region radius followed by the adapted soft restart in [21]: we increase the trust-region radius to $1.1\Delta_0$.

2.1 Hard restart

The idea of *hard restart* is to restart the algorithm from totally new starting points which are generated from the same mechanism as generating the initial interpolation set. The advantage of this restart type is its simplicity of implementation. When the algorithm detects that it is trapped in the valley of a local solution, the algorithm stops and restarts from a new initial interpolation set.

However, this strategy has some drawbacks also. First, it is not guaranteed that the algorithm will not converge towards in the same local minimum again. Second, it is clear that with this strategy, the users do not use any information, *i.e.,* objective function evaluations, from the previous runs, which is not efficient, especially in the context of blackbox optimization where we want to save expensive simulations.

2.2 Complete soft restart

Our first proposal restart strategy restarts the algorithm with a subset of starting points that are chosen among already-evaluated objective-function points. The intuition comes from the fact that, ordinarily, the condition of restarting is met after a certain number of iterations that may provide a good exploration of the domain to set up a new initial interpolation set.

The advantage of this method is that we have no waste of new simulations to set up the new initial interpolation set. However, it can happen that the set of evaluated points are not very well distributed over the domain, yielding a set of starting points that can be biased. Also, like the *hard restart*, it is possible that the algorithm converges towards a local solution that was already explored.

2.3 Partial soft restart

Our second proposal improves the previous method by starting from a set of points that includes both already-evaluated points and new points using the augmented DOE method proposed in section 1.

The advantage of this proposal is that we use the available information from previous simulations, while ensuring the good properties of the new starting point set.

3 Numerical results

This section presents preliminary comparative numerical results. Subsection 3.1 and Subsection 3.2 report numerical results on the DS-MM and Ackley functions respectively.

We compare five options of DFOb-d_{neck}:

- The standard DFOb-d_{neck} with the standard option of add-point step (named Standard),
- DFOb-d_{neck} with the augmented DoE add-points step (Augmented),
- DFOb-d_{neck} coupling the augmented DoE add-point step and hard restart (A-hard restart),
- DFOb-d_{neck} coupling the augmented DoE add-point step and complete soft restart (A-complete-soft restart),

- DFOb-d_{neck} coupling the augmented DoE add-point step and partial soft restart (A-partial-soft restart).

We set the maximum number of evaluation to 300. The five experiments both share two reference options and three restart options with the same initial design (generated by adapted-greedy algorithm, $nDoE = m + n + 1$). The value of the parameter α_k in the augmented DOE method is set to 0.01. The maximum number of consecutive discarded iterations is 50.

3.1 Results for the DS-MM function

The first test function is the piecewise-defined DS-MM function (see equation (111)) involving one continuous and five binary variables (corresponding to 8 individual one-dimensional functions) as described in section 1.

Figure 31 shows the best current objective-function value achieved by the 5 options, with respect to the number of objective-function evaluations. The standard, augmented, and coupling augmented and hard restart DFOb-d_{neck} options yield the same solution with the associated objective-function value: -1.948 within $43, 38$ and, 38 evaluations respectively. The A-complete-soft restart method yield the globally optimal solution with associated objective-function value -2.328 within 66 evaluations. The A-partial-soft restart finds a solution with objective-function value -2.319 within 67 evaluations, a minimum local solution at the level (y value) corresponding to the global solution. Figure 32 shows the solutions obtained by the five options.

3.2 Results for the Ackley function

We now perform a similar study for the well-known Ackley function. The Ackley function is originally for continuous variables and given by:

$$f(x) = -ae^{-b\sqrt{\frac{1}{m}\sum_{i=1}^{m} x_i^2}} - e^{\frac{1}{m}\sum_{i=1}^{m}\cos(cx_i)} + a + e, \tag{112}$$

where the parameter values are set to $a = 20, b = 0.2, c = 2\pi$ and $m = 2$. The function is evaluated on the square $x_i \in [-5, 5], i = 1, 2$. We use the cyclic-symmetry form construction from Chapter 5 of Part I with $n = 4$ binary variables (corresponding to 6 individual two-dimensional functions to define the resulting piecewise-defined function). The global minimizer is $x = (0, 0), y = (0, 0, 0, 0)$. The associated objective-function value is -1.7634. The plots of the 6 individual two-dimensional functions associated with the 6 necklaces are shown in Figure 33.

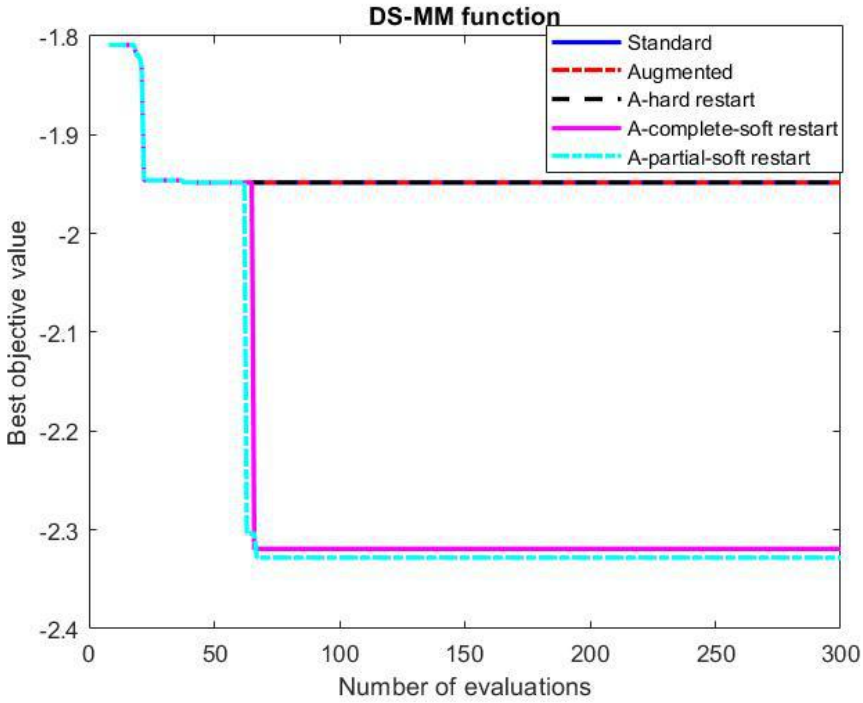

Fig. 31.: The best current objective-function values versus number of evaluations of DFOb-d_{neck} with standard, augmented add-point step, hard restart, complete soft restart, and partial soft restart on the DS-MM function.

We observe in the figure that each individual two-dimensional function has a numerous of local minima.

Figure 34 shows the best objective-function value achieved by each of the five options on the Ackley function, with respect to the number of objective-function evaluations. The solution found by the standard DFOb-d_{neck} has objective-function value: -1.336 and required 25 evaluations. The solution found by the augmented DFOb-d_{neck} has objective-function value: -1.7604 and required 92 evaluations. For A-hard restart the value is -1.7521, within 80 evaluations, A-complete-soft restart yields the objective-function value -1.7541 within 90 evaluations, and the solution found by A-partial-soft restart gives objective-function value -1.7607 and 110 evaluations. Figure 35 zooms on the main best objective-function values of the Figure 34. Figure 36 plots the solutions found by the five options; they all correspond to a same value of the binary-variable vector y corresponding to the first individual two-dimensional function.

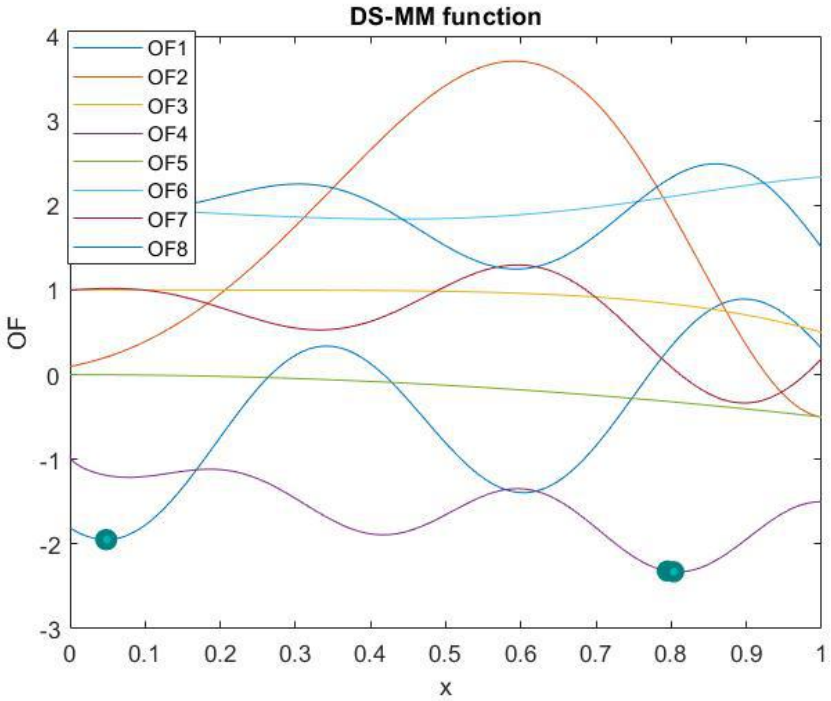

Fig. 32.: Solutions obtained by DFOb-d_{neck} with standard, augmented add-point step, hard restart, complete soft restart, and partial soft restart on the DS-MM function.

4 Conclusion

In this part of the thesis, we presented several original contributions to escape from local minima. A first contribution improves the "add-point" step of DFOb methods based on an original augmented design of experiment based on a discrete distribution taking into account the values of the available objective-function evaluations and the associated point distribution.

Another contribution is the introduction of two restart strategies based on new initial interpolation sets using previously simulated points. These methods were compared with the standard DFOb method, and with some reference restart strategies implemented in DFOb. The preliminary results obtained on two analytical functions with cyclic symmetry involving several local minima are encouraging. The proposed methods can also be used with DFOb-d_H for general mixed continuous and binary problems.

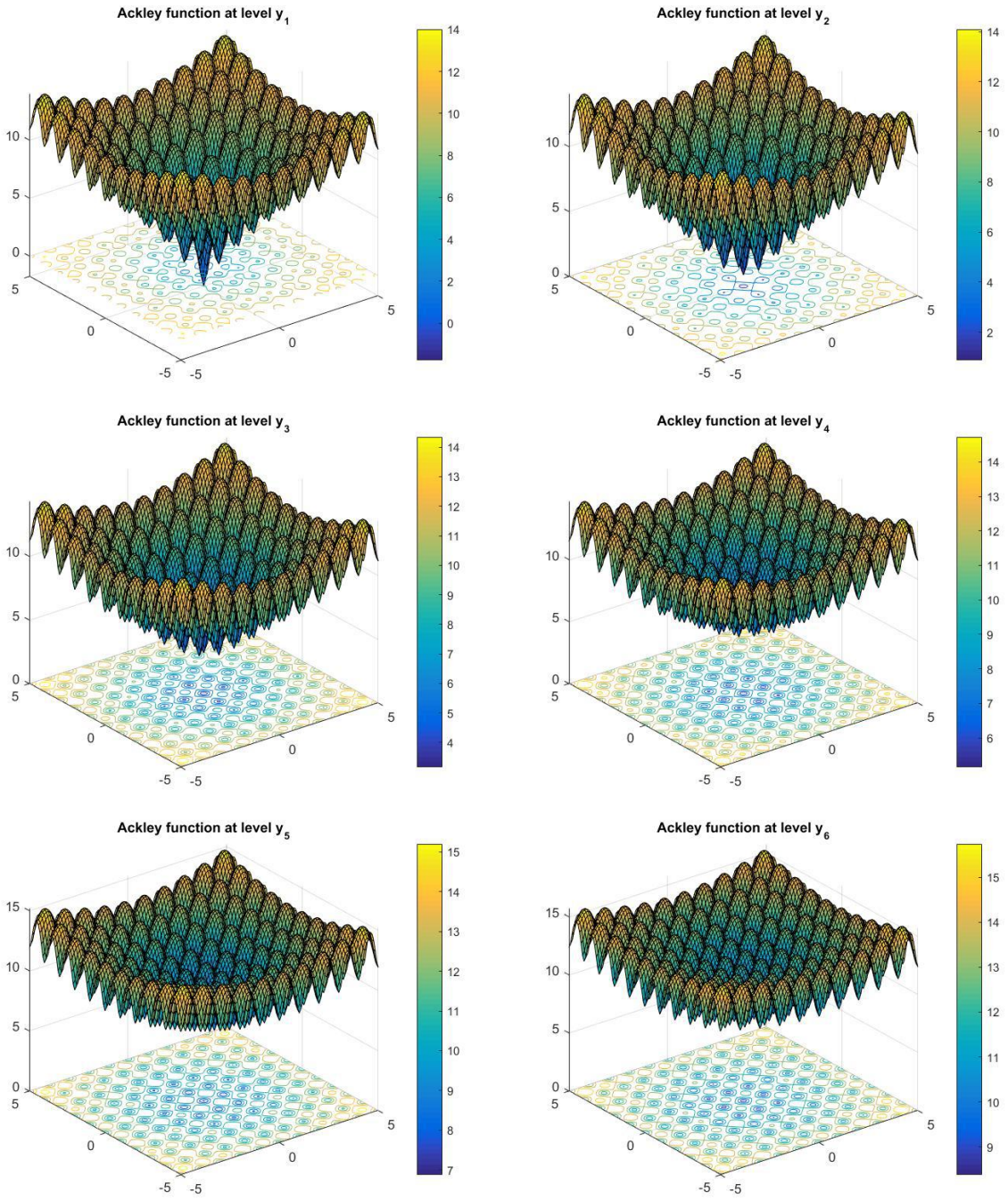

Fig. 33.: Ackley function with two continuous variables and four binary variables (corresponding 6 levels: $y_1 = (0,0,0,0), y_2 = (0,0,0,1), y_3 = (0,0,1,1), y_4 = (0,1,0,1), y_5 = (0,1,1,1), y_6 = (1,1,1,1)$).

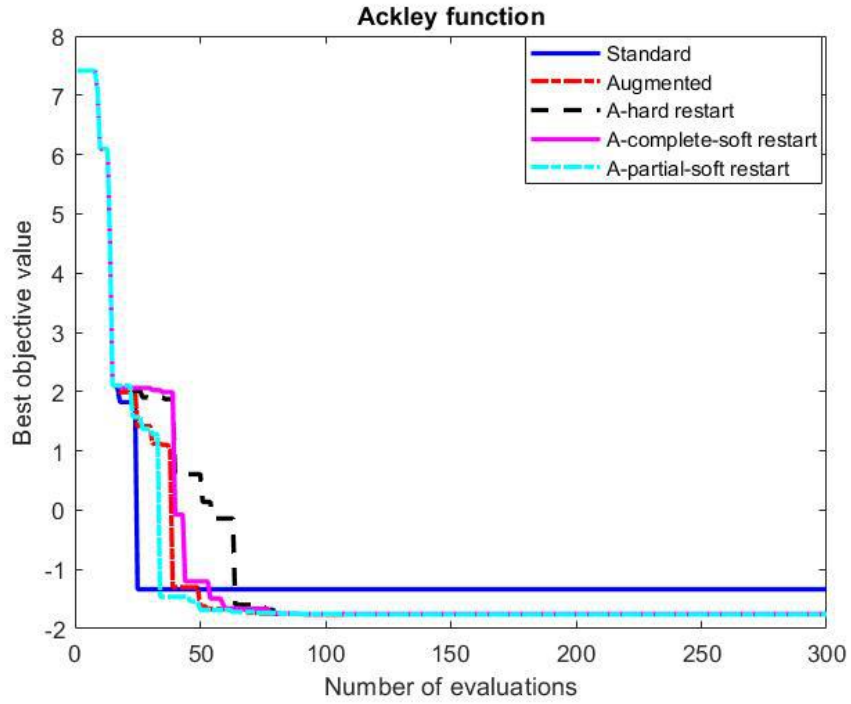

Fig. 34.: The best current objective-function values versus number of evaluations of DFOb-d_{neck} with the five options standard, augmented add-point step, hard restart, complete soft restart, and partial soft restart on the Ackley function.

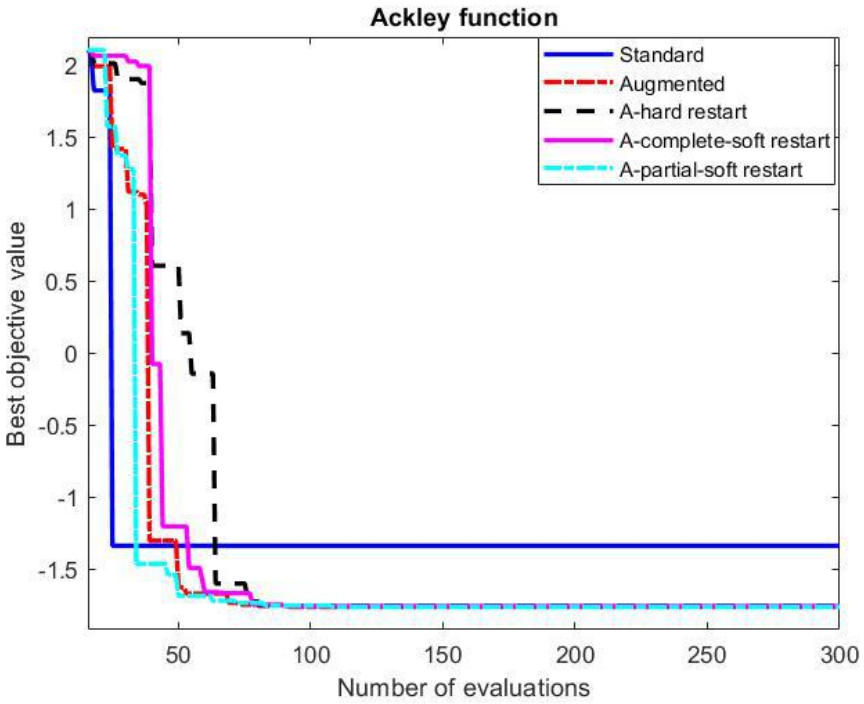

Fig. 35.: Zoom on the best current objective-function values versus number of evaluations of DFOb-d_{neck} with the five options.

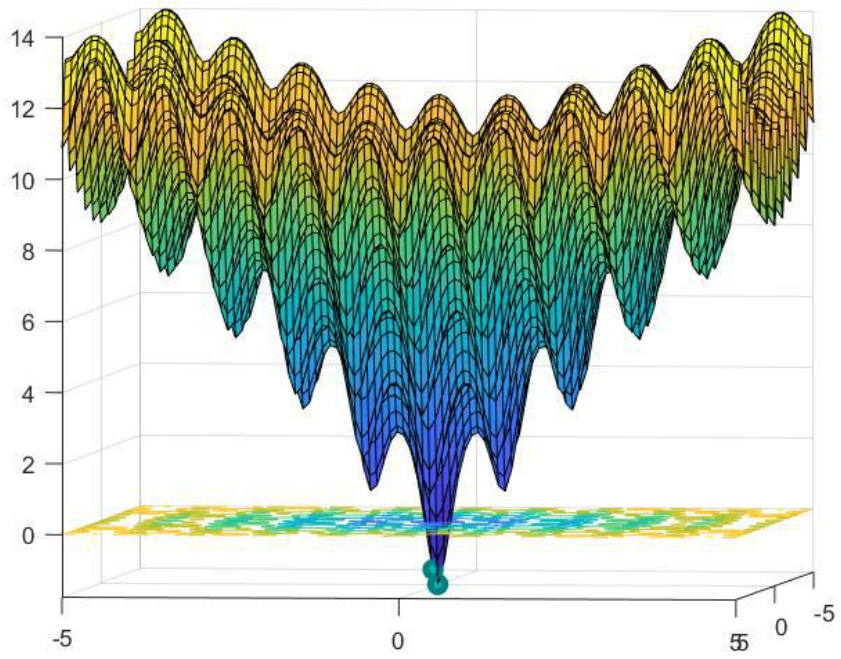

Fig. 36.: The two solutions obtained with DFOb-d_{neck} and the five options on the Ackley function.

General conclusion and perspectives

Conclusion

In this work, we presented several contributions to solve blackbox optimization problems involving computationally expensive simulators with mixed continuous and discrete variables. Numerous applications of this type of problems arise in the domain of optimal design: in particular, our research work was motivated by an application provided by Safran Tech for the design of turbomachine of aircraft.

- The first part of the thesis considers optimization methods for solving derivative-free problems. We presented an in-depth review of DFOb-d_H, a derivative-free trust-region method for mixed continuous and binary problems based on the Hamming distance. A first contribution was the convergence proof of this method which was not presented in the original work that proposed this method. The second contribution is a derivative-free trust-region method adapted to cyclic-symmetry mixed continuous and discrete problems: DFOb-d_{neck} method. This method relies on an adapted distance for cyclic-symmetry problems based on the "necklace" concept from combinatorics. Reformulations of the subproblems involved in the trust-region method DFOb were built to preserve the MIQP structure of the subproblems. The convergence proof of this new method is given. The proposed method was compared to three state-of-the art optimization methods: NOMAD, RBFOpt and DFLBOX and to the original DFOb-d_H method based on usual Hamming distance. This comparison was performed on a benchmark of 25 functions with cyclic symetry that was set up from various published benchmarks. The comparison is also done on an application problem provided by Safran Tech. DFOb-d_{neck} shows robust results within a limited number of function evaluations.

- The second part of this thesis is dedicated to the design of experiments for mixed continuous and discrete variables. We propose two extensions of a method based on kernel embedding of probability distribution for mixed variables space. The proposed methods can be applied

to various types of variables, provided a positive defined kernel. The performances of the methods are illustrated on examples with mixed discrete variables, cyclic-symmetry problems and also time series with dedicated kernels.

- Finally, in the third part of the thesis, new ideas are proposed and evaluated to improve the robustness of the optimization method DFOb-d_{neck} with respect to not getting trapped in local minima. Some encouraging preliminary results are presented.

Perspectives This work opens up several research avenues for future studies. From optimization perspective:

- First, apply the proposed method to an application of the optimal design problem in Safran's industrial context with the real complex black-box simulators (rather than on the simplified version of it).

- Regarding the problem of escaping from local minima:

  1. Extend the preliminary study described in Part 3: coupling the DOE methods proposed in Part 2 with restart strategies proposed in [21].

  2. Couple an exploration step based on global surrogate methods (EGO, RBFOpt) with the proposed trust-region method.

- Extend the proposed optimization methods DFOb for noisy problems: extension of works of [109] for mixed continuous and discrete variables.

- Extend the proposed optimization method DFOb-d_{neck} to general constrained problems. One technique can be used is a progressive barrier derivative-free trust-region in [10]: by using a constraint violation function to ensure the algorithm to find the feasible solutions.

From DoEs points of views:

- Evaluate the proposed DOE methods on various types of problems: for instance, images or applications with prior information.

Mathematical Background

A

We revise some concepts necessary to follow the work in this thesis. Some results in Section 1 and 2 are used in the proof of convergence of our method (see Chapters 4 and 5, Part I). Section 3 presents quadratic programming which is the core of our algorithm.

1 Vector and matrix norms

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

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

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

and the $\infty-$norm by

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

Similarly, we define the $p-$norm of matrices. Let $A = (a_{ij})_{m \times n}$ be a $m \times n$ matrix. The $p-$ norms of A, $p \geqslant 1$, 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 (non-negativity, scalar multiply, triangle inequality), they also satisfy the "submultiplicative" property

$$\|AB\| \leqslant \|A\| \|B\|.$$

There are several useful inequalities for matrix norms, namely

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

and for vector norms

$$\|x\|_\infty \leqslant \|x\|_2 \leqslant \sqrt{n}\|x\|_\infty.$$

2 Conditioning

One important property that we use to study a matrix is its condition number which gives a quantitative 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 noted $\chi(A)$ and is given by

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

which depends on the chosen norm. The value of the condition number gives an indication on the difficulty of solving numerically 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. More precisely, a small perturbation (due for instance to the finite precision of the machine or to the imprecision/uncertainty in the data) in the right-hand side of a system of linear equations can yield an enormous change in its solution when the matrix is ill-conditioned. If the matrix A is symmetric, there is another way to see the condition number, which is

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

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

3 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 opti-

mization algorithms. An optimization problem with a quadratic objective function and linear constraints is called a quadratic program, which can be stated as

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

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 definite positive (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 for instance [81].

Mixed Integer Quadratic Programming (MIQP)

MIQP is in the form of

$$\begin{cases} \min\limits_{x,z} f(x,z), \\ x \in \mathbb{R}^m, z \in \mathbb{Z}^n, \\ g(x,z) \leqslant 0. \end{cases} \tag{113}$$

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 [20, 33]).

B

Encoding techniques and list of benchmark functions

1 Categorical variables encoding techniques

There is a large number of engineering optimization problems with both continuous and discrete variables. In applications, the discrete variables are sometimes categorical ones; *i.e.*, they refer to a list or set of categories. Many optimization algorithms cannot deal directly with this type of variables: the categories must often be converted into numbers.

There are two types of categorical variables: *nominal* and *ordinal* variables. The term nominal refers to no intrinsic ordering, for instance, gender (male and female). Unlike nominal variables, ordinal variables have a clear ordering.

This section aims to give the readers a short review about different ways of encoding categorical variables. Since the work is more focused on mixed continuous and binary variables, we will give more detail on the techniques that convert categorical variables to binaries.

Binary encoding. This is one of the simplest way to encode categorical variables into quantitative variables (numbers). First, the categorical variables are converted to numeric (integer) numbers, denoted by $l_1, l_2, \ldots, l_N$, then one applies the following binary encoding to $l_k, k = 1, 2, \ldots, N$:

$$l_k = \sum_{i}^{\lfloor \log_2(n) \rfloor + 1} b_i^k 2^{i-1},$$

where $b_i^k \in \{0, 1\}$. Thus, the encoding of l_k is $(b_1^k, b_2^k, \ldots, b_{\lfloor \log_2(n) \rfloor + 1}^k), k = 1, 2, \ldots, N$.

One-hot encoding. This includes encoding each categorical variable with a different binary or boolean variable (also called dummy variable) which takes values 0 or 1, indicating whether a category is present in an observation.

2 Benchmark functions

Luksan and Vlcek (2000) benchmark

Problem B.1. *CB2.*

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

Problem B.2. *CB3.*

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

Problem B.3. *QL.*

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

Problem B.4. *WF.*

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

Problem B.5. *PENTAGON.*

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

Problem B.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 B.7. *WONG 2.*

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

Problem B.8. *WONG 3.*

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

Problem B.9. *MAD1.*

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

Problem B.10. *MAD4.*

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

Hock and Schittkowski benchmark

Problem B.11. *HS2.*

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

Problem B.12. *HS3.*

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

Problem B.13. *HS29log.*

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

Dixon–Szegö benchmark

Problem B.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 B.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 B.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 B.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 B.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 B.19. *Shekel7.*

```
a = [ [4.0, 1.0, 8.0, 6.0, 3.0, 2.0, 5.0],
      [4.0, 1.0, 8.0, 6.0, 7.0, 9.0, 5.0],
      [4.0, 1.0, 8.0, 6.0, 3.0, 2.0, 3.0],
      [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]
```

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

Problem B.20. *Shekel10.*

```
a = [ [4.0, 1.0, 8.0, 6.0, 3.0, 2.0, 5.0, 8.0, 6.0, 7.0],
```

```
        [4.0, 1.0, 8.0, 6.0, 7.0, 9.0, 5.0, 1.0, 2.0, 3.6],
        [4.0, 1.0, 8.0, 6.0, 3.0, 2.0, 3.0, 8.0, 6.0, 7.0],
        [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]
```

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

GLOBALLIB benchmark

Problem B.21. `ex8_1_1`

.

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

Problem B.22. `ex8_1_4`

.

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

Problem B.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 B.24. *Perm8.*

$$\beta = 100,$$

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

MINLPLib2 benchmark

Problem B.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] M. Abramson, C. Audet, G. Couture, et al. *The NOMAD project*. Software available at `https://www.gerad.ca/nomad/`. 2011.

[2] T. Achterberg, T. Berthold, T. Koch, and K. Wolter. "Constraint Integer Programming: A New Approach to Integrate CP and MIP". In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. LNCS 5015, CPAIOR 2008 Paris, France*. Ed. by L. Perron and M. Trick. Springer, 2008, pp. 6–20.

[3] *Air Transport Action Group (ATAG)*. 2016. URL: `http://www.atag.org/facts-and-figures.html`.

[4] I. Alsmadi and M. Nuser. "String matching evaluation methods for DNA comparation". In: *International Journal of advanced Science and Technology* 47 (2012), pp. 13–32.

[5] Y. J. P. Ardila, R. Clifford, and M. Mohamed. "Necklace Swap problem for rhythmic similarity measure". In: *Consens M., Navarro G. (eds) String Processing and Information Retrieval. SPIRE 2005. Lecture Notes in Computer Science, Springer* 3772 (2005).

[6] C. Audet and W. Hare. *Derivative-Free and Blackbox Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, 2017, p. 302.

[7] C. Audet and J. J. E. Dennis. "Analysis of Generalized Pattern Searches". In: *SIAM Journal on Optimization* 13 (Nov. 2000), pp. 889–903.

[8] C. Audet and J. J. E. Dennis. "Mesh Adaptive Direct Search Algorithms for Constrained Optimization". In: *SIAM Journal on Optimization* 17 (Jan. 2006), pp. 188–217.

[9] C. Audet, V. Béchard, and S. Le Digabel. "Nonsmooth optimization through Mesh Adaptive Direct Search and Variable Neighborhood Search". In: *Journal of Global Optimization* 41 (June 2008), pp. 299–318.

[10] C. Audet, A. Conn, S. Le Digabel, and M. Peyrega. "A progressive barrier derivative-free trust-region algorithm for constrained optimization". In: *Computational Optimization and Applications* 71 (Nov. 2018).

[11] C. Audet and J. E. Dennis. "Pattern Search Algorithms for Mixed Variable Programming". In: *SIAM Journal on Optimization* 11.3 (2001), pp. 573–594. URL: `https://doi.org/10.1137/S1052623499352024`.

[12] C. Audet and J. J. E. Dennis. "Analysis of generalized pattern searches". In: *SIAM Journal on Optimization* 13 (2003), pp. 889–903.

[13] C. Audet, S. Le Digabel, and C. Tribes. “The mesh adaptive direct search algorithm for granular and discrete variables”. In: *SIAM Journal on Optimization* 29.2 (2019), pp. 1164–1189. published.

[14] C. Audet and C. Tribes. “Mesh-based Nelder–Mead algorithm for inequality constrained optimization”. In: *Computational Optimization and Applications* 71 (2018), pp. 331–352.

[15] N. Bartoli, T. Lefebvre, S. Dubreuil, et al. “Adaptive modeling strategy for constrained global optimization with application to aerodynamic wing design”. In: *Aerospace Science and Technology* 90 (2019), pp. 85–102. URL: http://www.sciencedirect.com/science/article/pii/S1270963818306011.

[16] P. Belotti, C. Kirches, S. Leyffer, et al. “Mixed-integer nonlinear optimization”. In: *Acta Numerica* 22 (2013), pp. 1–131.

[17] P. Bonami, L. T. Biegler, A. R. Conn, et al. “An algorithmic framework for convex mixed integer nonlinear programs”. In: *Discrete Optimization* 5.2 (2008). In Memory of George B. Dantzig, pp. 186–204. URL: https://doi.org/10.1016/j.disopt.2006.10.011.

[18] M. A. Bouhlel, N. Bartoli, R. G. Regis, A. Otsmane, and J. Morlier. “Efficient global optimization for high-dimensional constrained problems by using the Kriging models combined with the partial least squares method”. In: *Engineering Optimization* 50.12 (2018), pp. 2038–2053. URL: https://doi.org/10.1080/0305215X.2017.1419344.

[19] D. Bremner, T. M. Chan, E. D. Demaine, et al. “Necklaces, Convolutions, and X+Y”. In: *Algorithmica* 69 (2014), pp. 294–314. URL: https://doi.org/10.1007/s00453-012-9734-3.

[20] S. Burer and A. N. Letchford. “Non-convex mixed-integer nonlinear programming: A survey”. In: *Surveys in Operations Research and Management Science* 17.2 (2012), pp. 97–106. URL: https://www.sciencedirect.com/science/article/pii/S1876735412000037.

[21] C. Cartis, L. Roberts, and O. Sheridan-Methven. “Escaping local minima with derivative-free methods: A numerical investigation”. In: *Optimization* (2021), pp. 1–31. URL: https://doi.org/10.1080/02331934.2021.1883015.

[22] Y. Chen, M. Welling, and A. Smola. “Super-Samples from Kernel Herding”. In: *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence* (2010), pp. 1–8.

[23] B. K. Choi, J. Lentz, A. J. Rivas-Guerra, and M. P. Mignolet. “Optimization of intentional mistuning patterns for the reduction of the forced response effects of unintentional mistuning: Formulation and assessment”. In: *Journal of Engineering for Gas Turbines and Power* 125.1 (Jan. 2003), pp. 131–140.

[24] B. Choi. “Pattern optimization of intentional blade mistuning for the reduction of the forced response using genetic algorithm”. In: *KSME International Journal* 17.7 (July 2003), pp. 966–977. URL: https://doi.org/10.1007/BF02982981.

[25] A. 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”. In: *SMAI-MODE, Toulouse, France* (2016). URL: https://mode2016.sciencesconf.org/file/223761.

[26] A. R. Conn and S. L. Digabel. “Use of quadratic models with mesh-adaptive direct search for constrained black box optimization”. In: *Optimization Methods & Software* 28.1 (2013), pp. 139–158. URL: https://doi.org/10.1080/10556788.2011.623162.

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

[28] A. R. Conn, K. Scheinberg, and L. N. Vicente. "Global Convergence of General Derivative-Free Trust-Region Algorithms to First- and Second-Order Critical Points". In: *SIAM Journal on Optimization* 20.1 (2009), pp. 387–415. URL: https://doi.org/10.1137/060673424.

[29] S. D. Constantin and T. Rao. "On the theory of binary asymmetric error correcting codes". In: *Information and Control* 40.1 (1979), pp. 20–36. URL: http://www.sciencedirect.com/science/article/pii/S0019995879903292.

[30] A. Costa and G. Nannicini. "RBFOpt: An open-source library for black-box optimization with costly function evaluations". In: *Mathematical Programming Computation* 10.4 (Dec. 2018), pp. 597–629. URL: https://doi.org/10.1007/s12532-018-0144-7.

[31] M. Cuturi, J. Vert, O. Birkenes, and T. Matsui. "A Kernel for Time Series Based on Global Alignments". In: *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*. Vol. 2. 2007, pp. II-413-II–416.

[32] C. D'Ambrosio, A. Frangioni, L. Liberti, and A. Lodi. "On interval-subgradient and no-good cuts". In: *Operations Research Letters* 38.5 (2010), pp. 341–345. URL: http://www.sciencedirect.com/science/article/pii/S0167637710000738.

[33] C. D'Ambrosio and A. Lodi. "Mixed integer nonlinear programming tools: An updated practical overview". In: *Annals of Operations Research* 204 (2013), pp. 1572–9338. URL: https://doi.org/10.1007/s10479-012-1272-5.

[34] E. D. Demaine, F. Gomez-Martin, H. Meijer, et al. "The distance geometry of music". In: *Computational Geometry* 42.5 (2009). Special Issue on the Canadian Conference on Computational Geometry (CCCG 2005 and CCCG 2006), pp. 429–454. URL: http://www.sciencedirect.com/science/article/pii/S0925772108001156.

[35] Y. Diouane, V. Picheny, R. L. Riche, and A. S. D. Perrotolo. *TREGO: a Trust-Region Framework for Efficient Global Optimization*. 2021. arXiv: 2101.06808 [math.OC].

[36] L. Dixon and G. Szego. "The global optimization problem: An introduction". In: *In: Dixon, L., Szego, G. (eds.) Towards Global Optimization, North Holland* (1975), pp. 1–15.

[37] M. R. El Amri, C. Helbert, O. Lepreux, et al. "Data-driven stochastic inversion via functional quantization". In: *Statistics and Computing* 30.3 (May 2020), pp. 525–541. URL: https://hal-ifp.archives-ouvertes.fr/hal-02291766.

[38] M. R. El Amri, C. Helbert, M. Munoz Zuniga, C. Prieur, and D. Sinoquet. "Set inversion under functional uncertainties with joint meta-models". preprint. 2021. URL: https://hal-ifp.archives-ouvertes.fr/hal-02986558.

[39] D. Eriksson, M. Pearce, J. R. Gardner, R. Turner, and M. Poloczek. *Scalable Global Optimization via Local Bayesian Optimization*. 2020. arXiv: 1910.01739 [cs.LG].

[40] A. Forrester, A. Sobester, and A. Kean. *Engineering Design via Surrogate Modelling, a Practical Guide*. 2008.

[41] J. Franco. “Planification d’expériences numériques en phase exploratoire pour la simulation des phénomènes complexes”. Ph.D. thesis. Ecole Nationale Supérieure des Mines de Saint-Etienne, France, Sept. 2008. URL: https://tel.archives-ouvertes.fr/tel-00803107.

[42] H. Fredricksen and I. J. Kessler. “An algorithm for generating necklaces of beads in two colors”. In: *Discrete Mathematics* 61.2 (1986), pp. 181–188. URL: http://www.sciencedirect.com/science/article/pii/0012365X86900890.

[43] D. Gabric and J. Sawada. “Constructing de Bruijn sequences by concatenating smaller universal cycles”. In: *Theoretical Computer Science* 743 (2018), pp. 12–22. URL: https://doi.org/10.1016/j.tcs.2018.06.039.

[44] M. Gendreau and J.-Y. Potvin. *Handbook of Metaheuristics*. Vol. 272. International Series in Operations Research & Management Science. 3rd edition. Springer, 2019.

[45] P. Gilles. *Introduction to optimal vector quantization and its applications for numerics*. Technical report. 2014. URL: https://hal.archives-ouvertes.fr/hal-01034196.

[46] S. Graf and H. Luschgy. *Foundations of quantization for probability distributions*. Springer, 2007.

[47] C. Guo and F. Berkhahn. “Entity embeddings of categorical variables”. In: *arXiv preprint arXiv:1604.06737* (2016).

[48] H. M. Gutmann. “A Radial Basis Function Method for Global Optimization”. In: *Journal of Global Optimization* 19.3 (Mar. 2001), pp. 201–227. URL: https://doi.org/10.1023/A:1011255519438.

[49] Y. Han and M. P. Mignolet. “Optimization of intentional mistuning patterns for the mitigation of the effects of randon mistuning”. In: *SME Turbo Expo 2008, Berlin, Germany* (2008).

[50] W. Hock and K. Schittkowski. *Test examples for nonlinear programming codes*. Vol. 87. 1981.

[51] K. Holmström, N.-H. Quttineh, and M. M. Edvall. “An adaptive radial basis algorithm (ARBF) for expensive black-box mixed-integer constrained global optimization”. In: *Optimization and Engineering* 9.4 (Dec. 2008), pp. 311–339. URL: https://doi.org/10.1007/s11081-008-9037-3.

[52] F. Hutter, L. Xu, H. H. Hoos, and K. Leyton-Brown. “Algorithm runtime prediction: Methods & evaluation”. In: *Artificial Intelligence* 206 (2014), pp. 79–111. URL: http://www.sciencedirect.com/science/article/pii/S0004370213001082.

[53] *International Air Transport Association, IATA Price Analysis*. 2016. URL: http://www.iata.org/publications/economics/fuel-monitor/Pages/price-analysis.aspx.

[54] H. John. *Adaptation in natural and artificial systems*. University of Michigan Press, 1975.

[55] M. Johnson, L. Moore, and D. Ylvisaker. “Minimax and maximin distance designs”. In: *Journal of Statistical Planning and Inference* 26 (1990), pp. 131–148.

[56] D. R. Jones, M. Schonlau, and W. J. Welch. “Efficient Global Optimization of Expensive Black-Box Functions”. In: *Journal of Global Optimization* 13.4 (Dec. 1998), pp. 455–492. URL: https://doi.org/10.1023/A:1008306431147.

[57] V. R. Joseph, E. Gul, and S. Ba. “Maximum projection designs for computer experiments”. In: *Biometrika* 102.2 (Mar. 2015), pp. 371–380. URL: https://doi.org/10.1093/biomet/asv002.

[58] K. Kandasamy, J. Schneider, and B. Poczos. *High Dimensional Bayesian Optimisation and Bandits via Additive Models*. 2016. arXiv: 1503.01673 [stat.ML].

[59] J. Kruskal. “Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis”. In: *Psychometrika* 29 (1964), pp. 1–27. URL: https://doi.org/10.1007/BF02289565.

[60] M. Lange, D. Zühlke, O. Holz, T. Villmann, and S.-G. Mittweida. “Applications of lp-Norms and their Smooth Approximations for Gradient Based Learning Vector Quantization.” In: *ESANN*. Citeseer. 2014, pp. 271–276.

[61] S. Le Digabel. “Algorithm 909: NOMAD: Nonlinear Optimization with the MADS algorithm”. In: *ACM Transactions on Mathematical Software* 37.4 (2011), pp. 1–15.

[62] G. Liuzzi, S. Lucidi, and F. Rinaldi. “Derivative-free methods for mixed-integer constrained optimization problems”. In: *Journal of Optimization Theory and Applications* 164 (2015), pp. 933–965.

[63] G. Luizzi, S. Lucidi, and F. Rinaldi. “Derivative-free methods for bound constrained mixed-integer optimization”. In: *Computational Optimization and Application* 53 (2012), pp. 505–526.

[64] L. Luksan and J. Vlček. *Test Problems for Nonsmooth Unconstrained and Linearly Constrained Optimization*. Jan. 2000.

[65] S. Mak and V. R. Joseph. “Support points”. In: *The Annals of Statistics* 46.6A (2018), pp. 2562–2592.

[66] M. D. Mckay, R. J. Beckman, and W. J. Conover. “A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code”. In: *American Statistical Association and American Society for Quality* 42 (2010), pp. 55–61. URL: http://www.jstor.org/stable/1271432.

[67] M. McKay, R. Beckman, and W. Conover. “A comparison of three methods for selecting values of input variables in the analysis of output from a computer code”. In: *Technometrics* 21 (1979), pp. 239–245.

[68] F. Meunier and B. Neveu. “Computing solutions of the paintshop–necklace problem”. In: *Computers & Operations Research* 39.11 (2012), pp. 2666–2678. URL: http://www.sciencedirect.com/science/article/pii/S0305054812000263.

[69] J. Minghui. “A Linear-Time Algorithm for Hamming Distance with Shifts”. In: *Theory of Computing Systems* 44 (2009), pp. 349–355.

[70] J. Minghui. “On the sum of distances along a circle”. In: *Discrete Mathematics* 308.10 (2008), pp. 2038–2045.

[71] *MINLPLib2*. URL: http://www.gamsworld.org/minlp/minlplib2/html/.

[72] H. Mohammadi, R. L. Riche, and E. Touboul. “Making EGO and CMA-ES Complementary for Global Optimization”. In: *Learning and Intelligent Optimization* (Jan. 2015), pp. 287–292. URL: https://doi.org/10.1007/978-3-319-19084-6_29.

[73] J. Moré and S. M. Wild. “Benchmarking Derivative-Free Optimization Algorithms”. In: *SIAM Journal on Optimization* 20 (Mar. 2009), pp. 172–191.

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

[75] J. Müller, C. A. Shoemaker, and R. Piché. “SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems”. In: *Computers & Operations Research* 40.5 (2013), pp. 1383–1400. URL: http://www.sciencedirect.com/science/article/pii/S0305054812001967.

[76] M. Munoz Zuniga and D. Sinoquet. “Global optimization for mixed categorical-continuous variables based on Gaussian process models with a randomized categorical space exploration step”. In: *INFOR: Information Systems and Operational Research* 58.2 (2020), pp. 310–341. URL: https://doi.org/10.1080/03155986.2020.1730677.

[77] G. Nannicini. “On the implementation of a global optimization method for mixed-variable problems”. en. In: *Open Journal of Mathematical Optimization* 2 (2021). URL: https://ojmo.centre-mersenne.org/articles/10.5802/ojmo.3/.

[78] S. Nanty, C. Helbert, A. Marrel, N. Pérot, and C. Prieur. “Sampling, metamodeling, and sensitivity analysis of numerical simulators with functional stochastic inputs”. In: *SIAM/ASA Journal on Uncertainty Quantification* 4.1 (2016), pp. 636–659.

[79] J. Nelder and R. Mead. “A Simplex Method for Function Minimization”. In: *The Computer Journal* 7 (1965), pp. 308–313.

[80] A. Neumaier. *Neumaier’s collection of test problems for global optimization*. Retrieved in May 2014. URL: https://www.mat.univie.ac.at/~neum/glopt/my_problems.html.

[81] J. Nocedal and S. Wright. *Numerical optimization*. 2006.

[82] R. Oeuvray and M. Bierlaire. “Boosters: A Derivative-Free Algorithm Based on Radial Basis Functions”. In: *International Journal of Modelling and Simulation* 29.1 (2009), pp. 26–36. URL: https://doi.org/10.1080/02286203.2009.11442507.

[83] J. Pelamatti, L. Brevault, M. Balesdent, E.-G. Talbi, and Y. Guerin. “Bayesian optimization of variable-size design space problems”. In: *Optimization Engineering* 22 (2021), pp. 387–447. URL: https://doi.org/10.1007/s11081-020-09520-z.

[84] J. Pelamatti, L. Brévault, M. Balesdent, E.-G. Talbi, and Y. Guerin. “Efficient global optimization of constrained mixed variable problems”. In: *Journal of Global Optimization* 73.3 (Mar. 2019), pp. 583–613.

[85] N. Ploskas and N. V. Sahinidis. “Review and comparison of algorithms and software for mixed-integer derivative-free optimization”. In: *Journal of Global Optimization* (2021).

[86] M. Powell. “On trust region methods for unconstrained minimization without derivatives”. In: *Mathematical Programming* 97 (2003), pp. 605–623.

[87] L. Pronzato. “Minimax and maximin space-filling designs: some properties and methods for construction”. In: *Journal de la Societe Française de Statistique, Societe Française de Statistique et Societe Mathematique de France* 158 (1) (2017), pp. 7–36.

[88] P. Z. G. Qian. “Sliced Latin Hypercube Designs”. In: *Journal of the American Statistical Association* 107.497 (2012), pp. 393–399. URL: http://www.jstor.org/stable/23239678.

[89] R. Regis and C. Shoemaker. “A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions”. In: *INFORMS Journal on Computing* 19 (Nov. 2007), pp. 497–509.

[90] R. Regis and C. Shoemaker. “Improved Strategies for Radial basis Function Methods for Global Optimization”. In: *Journal of Global Optimization* 37 (Jan. 2007), pp. 113–135.

[91] R. G. Regis. “Trust regions in Kriging-based optimization with expected improvement”. In: *Engineering Optimization* 48.6 (2016), pp. 1037–1059. eprint: https://doi.org/10.1080/0305215X.2015.1082350. URL: https://doi.org/10.1080/0305215X.2015.1082350.

[92] T. J. Santner, B. J. Williams, and W. I. Notz. *The design and analysis of computer experiments*. Springer, New York, NY, 2003. URL: https://doi.org/10.1007/978-1-4757-3799-8.

[93] D. K. Shin, Z. Gurdal, and J. O. H. Griffin. “A penalty approach for nonlinear optimization with discrete design variables”. In: *Engineering Optimization* 16.1 (1990), pp. 29–42. URL: https://doi.org/10.1080/03052159008941163.

[94] A. Smola, A. Gretton, L. Song, and B. Schölkopf. “A Hilbert space embedding for distributions”. In: *Algorithmic Learning Theory*. Vol. 4754. Springer. 2007, pp. 13–31.

[95] J. Tan, N. Nayman, M. Wang, and R. Jin. *CobBO: Coordinate Backoff Bayesian Optimization*. 2021. arXiv: 2101.05147 [cs.LG].

[96] O. Teymur, J. Gorham, M. Riabiz, and C. J. Oates. *Optimal quantisation of probability measures using maximum mean discrepancy*. 2020. arXiv: 2010.07064 [stat.ML].

[97] V. Torczon. “On the Convergence of Pattern Search Algorithms”. In: *SIAM Journal on Optimization* 7 (1997), pp. 1–25.

[98] G. Toussaint. “A comparison of rhythmic similarity measures”. In: *ISMIR*. 2004.

[99] G. Toussaint. “A mathematical analysis of African, Brazilian, and Cuban clave rhythms”. In: *Townson University*. 2002, pp. 157–168.

[100] G. Toussaint. “Computational geometric aspects of rhythm, melody, and voice-leading”. In: *Computational Geometry* 43.1 (2010). Special Issue on the 14th Annual Fall Workshop, pp. 2–22. URL: http://www.sciencedirect.com/science/article/pii/S092577210900042X.

[101] G. Toussaint. “The Geometry of Musical Rhythm”. In: *Discrete and Computational Geometry*. Ed. by J. Akiyama, M. Kano, and X. Tan. Springer, 2005, pp. 198–212.

[102] T. T. Tran, D. Sinoquet, S. Da Veiga, and M. Mongeau. “An Adapted Derivative-Free Optimization Method for an Optimal Design Application with Mixed Binary and Continuous Variables”. In: *6th International Conference on Computer Science, Applied Mathematics and Applications, ICCSAMA 2019*. Hanoï, Vietnam, Dec. 2020. URL: https://hal.archives-ouvertes.fr/hal-02494138.

[103] T. T. Tran, D. Sinoquet, S. D. Veiga, and M. Mongeau. “Derivative-free mixed binary necklace optimization for cyclic-symmetry optimal design problems”. In: *Accepted (on Aug 27, 2021) to be published in Optimization & Engineering* (2021).

[104] C. Van Den Berg, J. P. R. Christensen, and P. Ressel. *Harmonic analysis on semigroups: theory of positive definite and related functions*. Vol. 100. Springer Science & Business Media, 2012.

[105] K. K. Vu, C. D’Ambrosio, Y. Hamadi, and L. Liberti. “Surrogate-based methods for black-box optimization”. In: *International Transactions in Operational Research* 24 (2017), pp. 393–424.

[106] A. Wächter and L. T. Biegler. “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming”. In: *Mathematical Programming* 106 (2006), pp. 25–57.

[107] Z. Wang, F. Hutter, M. Zoghi, D. Matheson, and N. de Freitas. *Bayesian Optimization in a Billion Dimensions via Random Embeddings*. 2016. arXiv: 1301.1942 [stat.ML].

[108] S. M. Wild, R. Regis, and C. Shoemaker. "ORBIT: Optimization by Radial Basis Function Interpolation in Trust-Regions". In: *SIAM Journal on Scientific Computing* 30 (2008), pp. 3197–3219.

[109] S. M. Wild. "Derivative-Free Optimization Algorithms for Computationally Expensive Functions". PhD thesis. Cornell University, Ithaca, NY, 2009.

[110] L. Wu, I. E.-H. Yen, S. Huo, et al. *Efficient Global String Kernel with Random Features: Beyond Counting Substructures*. 2019. arXiv: 1911.11121 [cs.LG].

[111] S.-J. Wu and P.-T. Chow. "Genetic algorithms for solving mixed-discrete optimization problems". In: *Journal of the Franklin Institute* 331.4 (1994), pp. 381–401. URL: https://www.sciencedirect.com/science/article/pii/0016003294900043.

[112] J. Yang, Q. Liu, V. Rao, and J. Neville. "Goodness-of-Fit Testing for Discrete Distributions via Stein Discrepancy". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, Oct. 2018, pp. 5561–5570. URL: http://proceedings.mlr.press/v80/yang18c.html.

[113] Y. Zhang, S. Tao, W. Chen, and D. W. Apley. "A latent variable approach to Gaussian process modeling with qualitative and quantitative factors". In: *Technometrics* 62.3 (2020), pp. 291–302.