Algorithms for solving derivative-free optimization over mixed-integer domains

Juan José Torres Figueroa

UNIVERSITÉ PARIS XIII - SORBONNE PARIS NORD
École Doctorale Sciences, Technologies, Santé Galilée

ALGORITHMES POUR L'OPTIMISATION SANS DÉRIVÉES AVEC VARIABLES MIXTES

ALGORITHMS FOR SOLVING DERIVATIVE-FREE OPTIMIZATION OVER MIXED-INTEGER DOMAINS

THÈSE DE DOCTORAT
présentée par

Juan José TORRES FIGUEROA
Laboratoire d'Informatique de Paris Nord, CNRS UMR 7030

pour l'obtention du grade de
DOCTEUR EN INFORMATIQUE

soutenue le 11/07/2022 devant le jury d'examen composé de :

BASSINO Frédérique, Université Sorbonne Paris Nord ..... Examinatrice
GRATTON Serge, Toulouse INP ..... Rapporteur
RINALDI Francesco, Università di Padova ..... Rapporteur
SINOQUET Delphine, IFP Energies nouvelles ..... Examinatrice
WILD Stefan, Argonne National Laboratory ..... Examinateur
NANNICINI Giacomo, IBM Quantum ..... Co-encadrent de thèse
WOLFLER CALVO Roberto, Université Sorbonne Paris Nord . Directeur de thèse
TRAVERSI Emiliano, Université Sorbonne Paris Nord .... Co-encadrant de thèse

Résumé

L'optimisation sans dérivées est un outil populaire pour résoudre des problèmes complexes dans lesquels la description analytique de la fonction objectif n'est pas disponible et le calcul des dérivées n'est pas pratique, sinon impossible. Beaucoup de ces problèmes considèrent des variables discrètes non relaxables qui ajoutent une complexité supplémentaire à l'optimisation et à l'analyse de convergence. Cette thèse concerne le développement de deux algorithmes sans dérivées avec variables mixtes qui adressent certains de ces complications.

Le premier algorithme est une adaptation de la méthode à région de confiance qui utilise une approximation quadratique adaptée de la fonction inconnue. Ces modèles sont construites sous l'hypothèse d'une structure quadratique locale. De plus, ces modèles se sont prouvés complètement linéaires dans voisinages réels et entiers restreints. Cet algorithme se prouve globalement convergent vers plusieurs définitions d'optimalité locale, même dans l'optimisation de fonctions qui n'affichent pas la structure quadratique locale.

Le deuxième algorithme est un hybride de la programmation DC et de la méthode de région confiance qui tire parti des structures potentielles sur la fonction objectif. Il est basé sur l'hypothèse que l'optimisation partielle par rapport aux variables discrètes peut être effectuée en utilisant un nombre polynomial d'évaluations de la fonction objectives. Cet algorithme est globalement convergent vers un point stationnaire par rapport aux variables continues et à l'optimum global dans le domaine entier.

De plus, cette thèse présente une définition générale d'un modèle entièrement linéaire et explore les mécanismes pour évaluer et maintenir des modèles avec des erreurs bornées. La contribution finale de ce travail est l'introduction de nouvelles mesures de stationnarité entière qui empêchent la terminaison algorithmique précoce et facilitent l'analyse de convergence.

Mots clés: Optimisation sans dérivées, Programmation non linéaire mixte en nombres entiers, Optimisation combinatoire.

Abstract

Derivative free-optimization is a popular tool for addressing complex problems in which the analytical description of the objective function is not available and the computation of derivative information is impractical, if not impossible. Many of these problems consider non-relaxable discrete variables that add further complexity to the optimization and the convergence analysis. This thesis concerns the development of two mixed-integer derivative-free algorithms that address some of these complications.

The first algorithm is an adaptation of the trust-region method that uses a tailored quadratic-surrogate approximation of the unknown function. Such model approximations are constructed under the assumption of a local mixed-integer quadratic structure. Moreover, these surrogates are proved to be *fully-linear* in restricted mixed-integer neighborhoods. This algorithm is proved to be globally convergent to several definitions of local optimality, even in the optimization of functions that do not display the local quadratic structure.

The second algorithm is a hybrid of the difference of convex algorithm (DCA) and the trust-region method that takes advantage of potential structures on the objective function. It is based on the assumption that the partial optimization with respect to the discrete variables can be performed using a polynomial number of objective evaluations. This algorithm is globally convergent to a point that is stationary with respect to the continuous variables and the global optimum in the discrete domain.

In addition, this thesis presents a general definition of a mixed-integer *fully-linear* model and explores the mechanisms to evaluate and maintain error bounded approximations. The final contribution of this work is the introduction of new mixed-integer stationarity measures that prevent the early algorithmic termination and facilitate the convergence analysis.

Keywords: Derivative free-optimization, Mixed-integer non-linear programming, Combinatorial optimization.

Acknowledgements

Contents

List of Figures

List of Tables

List of Abbreviations

CLM	Combined Local Minimum
DC	Difference of Convex
DCA	Difference of Convex Algorithm
DFO	Derivative Free Optimization
DSM	Direct Seatch Methods
DDSM	Directional Direct Search Methods
GPS	Generalized Pattern Search
LQMI	Locally-Quadratic Mixed-Integer
MADS	Mesh Adaptive Direct Search
RBF	Radial Basis Functions
SLM	Separate Local Minimum
StLM	Stronger Local Minimum

List of Symbols

$f(x)$	Objective function
$\hat{f}_y(x)$	Continuous manifold $\hat{f}_y(x) = f(x, y)$
$m(x), m_k(x), m_k^{icb}(x)$	Surrogate model
l^c, l_k^c	Linear continuous terms (Equation 3.4)
l^z, l_k^z	Linear integer terms (Equation 3.4)
n_c	Number of continuous variables
n_z	Number of integer variables
p_k	Projected path (Definition 4.6)
red	Reduction operator (Definition 1.7)
$supp^+, supp^-$	Positive and negative support (Definition 1.2)
t_k^C	Generalized Cauchy step (Definition 4.7)
x	Vector of continuous variables
x_k^C	Generalized Cauchy point (Definition 4.7)
x_{lb}	Continuous lower bound
x_{ub}	Continuous upper bound
y	Vector of integer variables
y_{lb}	Integer lower bound
y_{ub}	Integer upper bound
A_M	Continuous - Integer bilinear coefficients (Equation 3.4)
A_z	Integer quadratic matrix (Equation 3.4)
I_y	Integer mapping (Definition 5.3)
M, M_k	Matrix of integer generating vectors (Definition 3.4)
$P_{\Omega_c}(x)$	Projection of the vector x onto the set Ω_c
$Q(y, M), Q_k$	Set of integer generating vectors (Definition 3.5)
Υ^k	Set of previously sampled points (Algorithm 4.1 and Algorithm 5.4)

$\Delta^c, \Delta_k^c, \Delta_k^{icbc}$	Continuous trust-region radius
$\Delta^z, \Delta_k^z, \Delta_k^{icbz}$	Integer trust-region radius
ϵ_a	Acceptance tolerance (Algorithm 4.1)
ϵ_c	Criticality tolerance (Algorithm 4.1 and Algorithm 5.4)
κ_{cri}	(Corollary 4.2)
$\kappa_{ef}, \bar{\kappa}_{ef}, \tilde{\kappa}_{ef}$	Error constant on function approximation
$\kappa_{eg}, \bar{\kappa}_{eg}, \tilde{\kappa}_{eg}$	Error constant on gradient approximation
κ_{eh}	Error constant on hessian approximation
κ_f	Global Lipschitz constant on $\hat{f}_y(x), \ \forall y \in \Omega_z$
κ_{frd}	(Corollary 4.1)
κ_g	Global Lipschitz constant on $\nabla_x \hat{f}_y(x), \ \forall y \in \Omega_z$
λ, λ_k	Regularization parameter
$\hat{v}(x)$	Surrogate approximation of ψ (Lemma 3.1)
$\tau(y), \bar{\tau}(y), \tilde{\tau}(y)$	Integer quadratic function
$\phi(x)$	Local bilinear representation functions (Assumption 3.2)
$\psi(x)$	Linear transformation of ϕ (Definition 3.4)
$\chi_k, \bar{\chi}_k$	First-order stationarity measure (Definition 4.3)
$\Gamma(x)$	Partial integer minimization (Definition 5.4)
Θ_k	Mixed-integer stationary parameter (Definition 4.1)
$\bar{\Theta}_k$	Adjusted mixed-integer staritonary parameter (Definition 4.1)
Φ_k	Combined stationary parameter (Definition 4.2)
$\Psi_k(x)$	Convex regularization (Definition 5.7)
Ω_c	Continuous domain
Ω_d	Discrete neighborhood
$\Omega_d^G(y, M)$	Reduced discrete neighborhood (Chapter 3.2.3)
$\Omega_d^Q(y)$	Discrete quadratic interpolation set (Chapter 4)
$\Omega_d^{LQMI}(y)$	LQMI discrete neighborhood (Equation 3.3)
Ω_m	Mixed-integer domain
Ω_z	Integer domain
$\|\cdot\|_1$	Manhattan norm
$\|\cdot\|_2$	Euclidean norm

$\|\cdot\|_\infty$	Maximum norm

Chapter 1

Introduction

In many real-world optimization problems the objective function and constraints are given by the complex computer simulations or the output of a black-box. In a black-box problem the analytical description of the objective function is not available; we only have access to a zeroth-order information of the function, and it is generally assumed that the oracle that provides the value of the function is computationally expensive to evaluate. Black-box problems arise in several settings, such as medical problems [1], [2], engineering design [3]–[5] and financial applications [6], among others.

Different methodologies have been developed to address the complexity of black-box problems, aiming to find near-optimal solutions while performing a reduced number of objective function evaluations to keep the optimization time acceptable. Such methodologies often include the use (and combination) of heuristics, direct-search algorithms, model based (or surrogate) approximation and randomized search [7]. Derivative-free optimization is a natural technique to solve black-box optimization problems, as it is intrinsically designed to avoid the computation of derivatives. Note that while a numerical approximation of the derivatives could in principle be computed (assuming they exist) even for a black-box problem, this is often too resource-intensive, e.g., $2n$ function evaluations to approximate the gradient of an n-variable function by finite central differences.

Black-box problems often include non-relaxable discrete variables. The presence of discrete (binary, integer or categorical) variables adds further challenges to the ones faced in the optimization of continuous functions [8]. There exist two crucial issues faced when studying mixed-integer derivative-free problems:

1. The definition of a proper mixed-integer minimizer.

2. The definition of adequate convergence criteria.

Newby and Ali [9] identify three possible local mixed-integer optimimum definitions: *separate local minimum*, *stronger separate local minimum* and *combined local minimum*. These three concepts deal with the degree of exploration of different integer manifolds around a tentative solution. In surrogate-based methods algorithm termination is commonly addressed via objective improvement or distance criteria, in those cases the convergence to a local minimizer cannot be guaranteed.

In this thesis we develop two convergent algorithms that tackle these two issues:

1. We propose a trust-region method that uses a tailored mixed-integer quadratic approximation (see Chapter 4), enabling the efficient reuse of previously sampled points. We establish global convergence of the algorithm to *separate local minimum* and *stronger separate local minimum*. The approximation is based on the assumption of dealing with what we call a "Locally-Quadratic Mixed-Integer" (LQMI) function, i.e., a structured objective function exhibiting properties that allow us to extend the notion of continuous *fully-linear models* to a mixed-integer context. This facilitates the convergence analysis, even when such an assumption does not hold. In addition, LQMI approximations can be obtained in a modular manner, providing flexibility in computing and maintaining the surrogate model within the trust region by using efficient methods taken from the continuous derivative-free optimization literature.

2. We propose a hybrid difference of convex and derivative-free algorithm (see Chapter 5) that takes advantage of potential combinatorial structures of the objective function. We establish global convergence of the algorithm to a strong version of *separate local minimum*. The approximation is based on the assumption that the partial optimization with respect to the discrete variables can be performed within polynomial number of function evaluations. It allows one to reformulate the problem as the difference of two functions and solve it with a modified version of the difference of convex algorithm that uses inexact subgradient information. This algorithm is related to first-order surrogate methods and consists of two phases: continuous local search and integer global

search. Moreover, the algorithm converges to a (weaker) local minimizer in functions that do not present such combinatorial structure.

Those two algorithms are the main contribution of this work. The following are additional results of this research:

- We provide a general definition of mixed-integer *fully-linear* models and we elaborate on the mechanisms to guarantee and maintain bounded error approximations (see Section 3.2.1).
- We revisit the concepts of mixed-integer optimality and describe a suitable ϵ-*strong separate minimum* that can be attained with model-based methods. In addition, we describe a mixed-integer stationarity parameter that helps to avoid premature convergence to stationary points and facilitates the convergence analysis (see Section 4.2).

In the next section we detail the main motivations behind this work.

1.1 Motivations

In this work we aim to develop convergent algorithms for derivative-free mixed-integer optimization while addressing the following two questions:

1. *How can we extend the convergence properties of the model-based methods to the mixed-integer domain?*
2. *How can we take advantage of the structure of the mixed-integer black-box function?*

The convergence of model-based derivative-free algorithms to stationary points depends on *fully-linear* models and criticality measures. It is not trivial to generalize these two concepts to the mixed-integer domain because of the discontinuity introduced by the discrete variables, this means that often the model-based methods proposed in the literature for the mixed case are heuristics [9]. This additional complexity of dealing with mixed-integer variables has been partially addressed by: (1) using quadratic surrogate approximations that are *fully-linear* in one fixed discrete point (Tran et al. [10]); or (2) relying on continuous search solvers to identify stationary points and explore surrounding continuous neighborhoods with different values

on the integer variables when the algorithm stops (Newby and Ali [9]). We remark that both strategies have strengths and weaknesses. With reduced *fully-linearity* one can guarantee convergence to a point that is stationary with respect to the continuous variables; however, the algorithm might get prematurely stuck at an undesirable solution. On the other hand, by using a local continuous solver one may be able to escape stationary points and improve the quality of the solution; nonetheless, the local search requires a large number of samples to certify stationarity/optimality thus reducing the rate of objective improvement, as observed by Newby and Ali. In this work we are interested in a methodology that exploits the strengths of both approaches: we develop algorithms that use a generalized definition of *fully-linearity* to guarantee convergence and that at the same time provides inside of surrounding continuous neighborhoods to prevent early stopping at bad quality stationary points.

Derivative-free optimization methods have been often used to solve black-box problems where the objective function presents known structures, including nonlinear least squares [11]–[13], sparse objective derivatives [14]–[16], composite (non) convex functions [17], bilevel and general min-max problems [18]. To the extend of our knowledge, only two studies have addressed the case where a structure on mixed-integer black-box is present: Tran et Al. [10] suppose that the function presents cyclic-symmetry properties, and Larson et al. [19] assume the function is globally convex. In this research we introduce two study assumptions of structure in mixed-integer functions: (1) locally bilinear interaction between discrete and continuous variables, and (2) combinatorial discrete properties when the vector of continuous variables is fixed. The first assumption allows us to easily derive error bounded approximations on both the function and its continuous gradient. The second assumption allows us to reformulate the problem and work with the two set of variables independently: using global zeroth order global methods to optimize with respect to the discrete variables and deal with the continuous variables via inexact subgradient methods. Note that both assumptions enable us to take advantage of the separable structure of the function to approximate different elements of the objective using continuous surrogate models. We show that the algorithms constructed under these structure assumptions can be extended to general mixed-integer functions

preserving convergence properties.

1.2 Problem Definition and Nomenclature

We are interested in solving the following mixed-integer optimization problem:

$$\min_{x,y} f(x,y) \tag{1.1}$$

$$x \in \Omega_c, \;\; y \in \Omega_z. \tag{1.2}$$

where $f : [\mathbb{R}^{n_c} \times \mathbb{Z}^{n_z}] \rightarrow \mathbb{R}$, $\Omega_c = \{x \in \mathbb{R}^{n_c} \mid x_{lb} \leq x \leq x_{ub}\}$ and $\Omega_z = \{y \in \mathbb{Z}^{n_z} \mid y_{lb} \leq y \leq y_{ub}\}$. We denote the mixed-integer box constraints as $\Omega_m = \{(x,y) \mid x \in \Omega_c, \;\; y \in \Omega_z\}$. When $n_z = 0$ the problem becomes purely continuous and we use the simplified notation $f(x)$, instead of $f(x,y)$. The following additional assumptions are imposed on the objective function:

Definition 1.1. *Let the function $\hat{f}_y : \mathbb{R}^{n_c} \rightarrow \mathbb{R}$ be the continuous manifold associated with y, defined as $\hat{f}_y(w) := f(w,y)$.*

Assumption 1.1. *The function $f(x,y)$ is bounded from below, i.e., there exists a constant $\kappa_{lb} \in \mathbb{R}$ such that $f(x,y) > \kappa_{lb}, \quad \forall (x,y) \in \Omega_m$.*

Assumption 1.2. *The continuous manifold $\hat{f}_y(w)$ is uniformly Lipschitz continuous with corresponding Lipschitz constant bounded by κ_f, $\forall y \in \Omega_z$.*

Assumption 1.3. *The continuous manifold $\hat{f}_y(w)$ is uniformly continuous differentiable with Lipschitz continuous gradient and corresponding Lipschitz constant bounded by κ_g, $\forall y \in \Omega_z$.*

Definition 1.2. *The positive support ($supp^+$) and the negative support ($supp^-$) of the vector $x \in \mathbb{R}^n$ are defined as:*

$$supp^+(x) = \{i \in \{1, \dots \; n\}, \;\; \mid x_i > 0\}$$

$$supp^-(x) = \{i \in \{1, \dots \; n\}, \;\; \mid x_i < 0\}$$

We denote by $\|\cdot\|_1$, $\|\cdot\|$ and $\|\cdot\|_\infty$ the Manhattan, Euclidean and maximum norm, respectively. To devise a trust region method, we define the neighborhoods used in the rest of the document.

Definition 1.3. *The continuous neighborhood centered at $\hat{x}$ of radius Δ^c is:*

$$\hat{B}_c(\hat{x}, \Delta^c) := \{x \in \mathbb{R}^{n_c} |\ \|x - \hat{x}\| \leq \Delta^c\}.$$

Definition 1.4. *The integer neighborhood centered at $\hat{y}$ of radius Δ^z is:*

$$\hat{B}_z(\hat{y}, \Delta^z) := \{y \in \mathbb{Z}^{n_z} |\ \|y - \hat{y}\|_\infty \leq \Delta^z\}.$$

Definition 1.5. *The combined mixed-integer neighborhood centered at $(\hat{x}, \hat{y})$ with radii Δ^c, Δ^z is:*

$$\hat{B}_v(\hat{x}, \hat{y}, \Delta^c, \Delta^z) := \left\{(x, y) \mid \ x \in \hat{B}_c(\hat{x}, \Delta^c), y \in \hat{B}_z(\hat{y}, \Delta^z)\right\}.$$

Note that we use different norms for the different neighborhoods; in particular, for integer variables it is more natural to use the ℓ_1-norm or ℓ_∞-norm distance. In this work we use the ℓ_∞-norm; we did not investigate how much effort is required to extend our results to the ℓ_1-norm.

Definition 1.6. *Let the mixed-integer neighborhood $\mathcal{N}_1(x, y)$ be defined as :*

$$\mathcal{N}_1(\hat{x}, \hat{y}) := \{(x, y) \in \Omega_m |\ \ x = \hat{x},\ \ \|y - \hat{y}\|_\infty \leq 1\}\ .$$

In the following sections we will be referring to a neighborhood $\mathcal{N}_D(x, y) \subseteq \mathcal{N}_1(x, y)$ for which the definitions of local optimality are constructed. It is named *user-defined neighborhood* as its size is defined by the final decision-maker. We introduce the concept of *reduced gradient* that allows us to evaluate first-order stationary conditions:

Definition 1.7. *The reduction (**red** : $[\mathbb{R}^{n_c} \times \mathbb{R}^{n_c}] \to \mathbb{R}^{n_c}$) of a vector g at the point x is defined as follows:*

$$\textbf{\textit{red}}(g, x)_i = \begin{cases} \max(g_i, 0) & \textbf{\textit{if}}\ \ x_i = x_{ub,i} \\ \min(g_i, 0) & \textbf{\textit{if}}\ \ x_i = x_{lb,i} \\ g_i & \textbf{\textit{otherwise}}. \end{cases}$$

Finally, we denote by the e_i the vector in which its ith component is equal to 1 and the rest are valued 0. Let ∇_x and ∇^2_{xx} be the gradient and the Hessian matrix with respect to the continuous variables, with $[\nabla_x f(x,y)]_i = \frac{\partial f(x,y)}{\partial x_i}$ and $[\nabla^2_{x,x} f(x,y)]_{i,j} = \frac{\partial^2 f(x,y)}{\partial x_i \partial x_j}$ for every $i, j \in \{1, \ldots, n_c\}$. The *convex hull* of a set S is denoted by $\mathbf{Co}(S)$.

1.3 Dissertation Outline

This document is organized as follows: In Chapter 2 we present the background of methodologies for the optimization of black-box functions, we define the versions of local optimality for mixed-integer functions and discuss the complications of proving optimality (Section 2.3). In Chapter 3 we define the class of LQMI functions and of mixed-integer fully-linear models. Section 3.2 introduces a quadratic mixed-integer interpolation scheme that yields a model with bounded error when approximating an unknown LQMI function, and shows that this model is accurate in a restricted neighborhood when the LQMI conditions do not hold. Chapter 4 presents a trust-region based algorithm, while Section 4.2 shows that it converges to a *separate local minimum* and *stronger local minimum* of a mixed-integer problem. Chapter 5 introduces the hybrid difference of convex and derivative free algorithm. Section 5.1 describes a class of functions that displays combinatorial structures, while Section 5.4 shows the convergence of the hybrid algorithm to a *separate local minimum* of a mixed-integer function. Chapter 6 presents the computational evaluation of the two proposed algorithms, and compares them with a state-of the art derivative-free solver. Finally, in Chapter 7 we draw our conclusions and detail future resarch perspectives.

Chapter 2

Literature Review

Different methodologies have been developed to solve optimization problems when only the zeroth order information is available. In this chapter, we summarize some of the most used derivative-free optimization techniques and highlight what elements we use in the development of our algorithmic framework. We separate this review of the literature in three sections: (1) methodologies for continuous local derivative-free optimization, (2) methodologies for mixed-integer derivative free optimization and (3) local optimality for mixed integer functions.

2.1 Continuous Derivative-Free Optimization

In this section, we address the methodologies for solving problem 1.1 when $n_z = 0$ and objective function f is assumed to be differentiable. These methodologies are typically classified in three different classes, depending on the (not) approximation of the function and its first and second-order information [7]: (1) Direct search methods, (2) model-based methods and (3) others. The latter group contains the methods that cannot be categorized in any of the first two and includes techniques such as the line-search based methods [20], [21], implicit filtering [22], [23] and adaptive regularized methods [24].

2.1.1 Direct Search Methods

The first class is called *Direct Search* Methods (DSM) which are described as the iterative evaluation of trial candidates given by a particular strategy [25]. DSM are

characterized for not approximating the function's gradient nor computing a surrogate model. This simplicity and their reliability make the DSM some of the most popular alternatives for solving black-box problems.

One of the most renown DSM variants consists in the Nelder-Mead [26] and simplex methods [27]–[29]; which are distinct from Dantzig's simplex linear programming method. Initially described in 1962 [30], these methodologies consist in the movement and manipulation of a *simplex*, which is defined as the convex hull of $n_c + 1$ linearly independent vectors in $\mathbb{R}^{n_c}$. Iteratively, the objective function is evaluated at all the vertices of the simplex to determine the vertex that yields the largest function value. This point is replaced with a new vertex that is computed by transforming the worst corner through reflection, extension, inner contraction and shrink operations. One of the weakness of the original Nelder-Mead method is the lack of theoretical guarantees of convergence into stationary solutions, even converging to points for which $\lim_{k\to\infty}\|\nabla_x f(x_k)\| \neq 0$ [31]. This problem has been continuously addressed and there exists simplex variants with proven convergence to stationary points [32], [33].

A second group of DSM consists in the *Directional Direct Search* Methods (DDSM). These methods consist on the computation of a new candidate solution x_{k+1} by performing local exploration around the current iterate x_k on the set $\{x_k + \Delta_k d_i \mid \forall i \in D_k\}$, where Δ_k is a positive step size and $d \in \mathbb{R}^{n_c}$ is a direction in the finite set of *polling* directions D_k. This procedure is usually referred as *poll step*. DDSM might also incorporate a preliminary exploration on a finite set of trial directions $\mathcal{H}_k$ that can be randomly generated or selected via heuristics. This second procedure is called the *search step*, and it is used to prevent early convergence to a bad quality local minimum. The basic DDSM is sketched in Algorithms 2.1 and 2.2.

We remark that when the *OPOR* parameter is set to 0, the poll step is denominated *complete poll* and its output is equivalent to $\min_{d_i \in D_k} f(x_k + \Delta_k d_i)$. On the other hand, if $OPOR = 1$ this step is named *opportunistic poll* [34]. DDSM are classified according to the way in which the poll directions are selected. If the set of directions is fixed at $D_k = \{\pm e_i \mid i \in \{1, \ldots, n_c\}\}$ the method is named *coordinate search*. Furthermore, when the set of directions D_k is fixed to a *positive spanning set* $D \subset \mathbb{R}^{n_c}$ the method is considered as one of the *Generalized Pattern Search* (GPS) methods [25],

Algorithm 2.1 TestDescend

Input: Point x_k, step length Δ_k, set of directions G and opportunistic search parameter $OPOR = \{0, 1\}$
Output: New candidate solution $\bar{x}$

Set $\bar{x} = x_k$
for $d_i \in G$ **do**
  if $f(\bar{x}) - f(x_k + \Delta_k d_i)$ is *acceptable* **then**
    Set $\bar{x} = x_k + \Delta_k d_i$
    if $OPOR = 1$ **then**
      break
    end if
  end if
end for

Algorithm 2.2 Generic DDSM Algorithm

Input: Initial candidate x_0 and step length Δ_0, geometric parameters $0 < \gamma < 1 \leq \gamma_{inc}$ and opportunistic search parameter $OPOR = \{0, 1\}$

for $k = 0, 1, 2, \ldots$ **do**
  Select a finite search of directions $\mathcal{H}_k$ (**Search step**)
  Set $\bar{x} =$ **TestDescend**$(x_k, \Delta_k, \mathcal{H}_k, OPOR)$
  if $x_k = \bar{x}$ **then**
    Select the set of poll directions $D_k \subset \mathbb{R}^{n_c}$ (**Poll step**)
    Set $\bar{x} =$ **TestDescend**$(x_k, \Delta_k, \{x_k + \Delta_k d_i \mid \forall i \in D_k\}, OPOR)$
  end if
  if $x_k = \bar{x}$ **then**
    Set $\Delta_{k+1} = \gamma \Delta_k$
  else
    Set $\Delta_{k+1} = \gamma_{inc} \Delta_k$
  end if
  Set $x_{k+1} = \bar{x}$
end for

[35]–[37]. Finally, we have the *Mesh Adaptive Direct Search* (MADS) methods [38]–[40] that consider a variable set of directions during the poll step and present strong theoretical results for convergence to second-order stationary points in continuous functions with locally Lipschitz gradients [41].

2.1.2 Model-Based Methods

The second group of optimization methodologies consists of the model-based methods, for which the computation of a new candidate solution rely on the prediction of a smooth, easy to evaluate, and accurate surrogate model approximating the objective function. Model-methods are widely used for the solution of constrained and unconstrained problems, performing more efficiently than the DSM [42]. Model-based methods are present in a number of solver implementations:

- *Derivative Free Optimization (DFO)*. Conn, Scheinberg and Toint [43].
- *WEDGE*. Marazzi and Nocedal [44].
- *CONDOR* Berghen and Bersini [45].
- Powell's *UOBYQA* [46], *NEWUOA* [47]–[49] and BOBYQA [50].

among others. We introduce the fundamentals for convergent model-based algorithms. The first element that is required is a measure of the accuracy of model m for the prediction of the objective f, its gradient and potentially its Hessian inside a local neighborhood $\hat{B}_c(x, \Delta)$. This measure is normally addressed through the concepts of *fully-linear* and *fully-quadratic* models:

Definition 2.1. *A model $m \in \mathcal{C}^1$ is called **fully-linear** with respect to $f(x)$, $f : \mathbb{R}^{n_c} \to \mathbb{R}$ in $\hat{B}_c(x, \Delta)$ if there exists a pair of constants $\kappa_{eg}, \kappa_{ef} > 0$, independent of Δ, for which*

$$\|\nabla_x f(x+s) - \nabla_x m(x+s)\| \leq \kappa_{eg}\Delta$$
$$|f(x+s) - m(x+s)| \leq \kappa_{ef}\Delta^2$$

for all $s \in \hat{B}_c(0, \Delta)$.

Definition 2.2. *A model $m \in \mathcal{C}^2$ is called **fully-quadratic** with respect to $f(x)$, $f : \mathbb{R}^{n_c} \to \mathbb{R}$ in $\hat{B}_c(x, \Delta)$ if there exists a set of constants $\kappa_{eh}, \kappa_{eg}, \kappa_{ef} > 0$, independent of Δ, for which*

$$\begin{aligned} \|\nabla^2_{xx} f(x+s) - \nabla^2_{xx} m(x+s)\| &\leq \kappa_{eh}\Delta \\ \|\nabla_x f(x+s) - \nabla_x m(x+s)\| &\leq \kappa_{eg}\Delta^2 \\ |f(x+s) - m(x+s)| &\leq \kappa_{ef}\Delta^3 \end{aligned}$$

for all $s \in \hat{B}_c(0, \Delta)$.

Types of surrogate approximation

Different types of surrogates can be used to achieve the accuracy described by the fully-linear and/or the fully quadratic models. The quality of the approximation of a sufficiently smooth function depends on the spatial distribution of the sampled elements inside $\hat{B}_c(x_k, \Delta_k)$; in other words, how *well-poissed* is the sampling set $X_k \subset \hat{B}_c(x_k, \Delta_k)$ on which the surrogate is computed [51]. The simplest and least expensive to evaluate and maintain correspond to the simplex gradients and linear interpolation models of the shape $m(x_k + s) = m(x_k) + l^\top s$ where $s, l \in \mathbb{R}^{n_c}$. We use this type of approximation in different parts of the two solution approximations developed in this work (see Algorithm 4.6).

A second group of surrogates corresponds to more general polynomial interpolation models. Let $\mathcal{P}^{d,n_c}$ be the space of polynomials of n_c and degree d, and $\bar{\phi}(x)$ the set of monomials $\bar{\phi} : \mathbb{R}^{n_c} \to \mathbb{R}^{dim(\mathcal{P}^{d,n_c})}$. For example, in the linear case $d = 1$ we have that $dim(\mathcal{P}^{d,n_c}) = n_c + 1$ and the set $\bar{\phi}(x) = \{1, x_1, x_2, \ldots, x_{n_c}\}$. Any model $m \in \mathcal{P}^{d,n_c}$ can be expressed in terms of $\phi(x)$ and the vector $a \in \mathbb{R}^{dim(\mathcal{P}^{d,n_c})}$ of coefficients:

$$m(x) = \sum_{i=1}^{dim(\mathcal{P}^{d,n_c})} a_i \bar{\phi}_i(x). \tag{2.1}$$

Given the set of interpolation points $X = \{x^1, x^2, \ldots, x^{\mathbf{p}}\} \subset \hat{B}_c(x_0, \Delta)$ the coefficients α are the solution of the linear system of equations:

$$f(x^j) = \sum_{i=1}^{dim(\mathcal{P}^{d,n_c})} a_i \bar{\phi}_i(x^j) \quad \forall j \in \{1, \ldots, \mathbf{p}\}.$$

The accuracy of a polynomial interpolation model (with $d \geq 2$) on the local neighborhood $\mathcal{B}$ is often assessed with the concepts of Lagrange polynomials and Λ-poisedness [51] . We highlight they are independent of the function f that is being approximated.

Definition 2.3. *Given the interpolation set* $X = \{x^1, x^2, \ldots, x^p\}$ *the basis* $\mathcal{L} : \mathbb{R}^{n_c} \to \mathbb{R}^p$ *of Lagrange polynomials satisfies*

$$\mathcal{L}_j(x^i) = \begin{cases} 1 & \textit{if } i = j \\ 0 & \textit{otherwise.} \end{cases}$$

A set of samples $X \subset \mathcal{B}$ is said Λ-poised for some $\Lambda > 0$ if and only if its elements are linearly independent and the basis $\mathcal{L}$ of related Lagrange polynomials satisfies the following condition:

$$\Lambda \geq \Lambda_{\mathcal{L}} = \max_{i=1,\ldots,\mathbf{p}} \max_{x \in \mathcal{B}} |\mathcal{L}_i(x)|.$$

The most relevant feature of the Lagrange polynomials is their relationship with the error bounds for the model approximation computed using the set $X \subset \mathcal{B}$. It is shown that for any x that lays in the convex hull of X

$$\|\mathcal{D}^r f(x) - \mathcal{D}^r m(x)\| \leq \frac{1}{(d+1)!} \nu_d \sum_{i=1}^{p} \|x^j - x\|^{d+1} \|\mathcal{D}^r \mathcal{L}_i(x)\|,$$

where $\mathcal{D}^r$ represents the r-th derivative and ν_d denotes an upper bound on $\mathcal{D}^{d+1} f(x)$ [52]. The latter implies that the $(d+1)$st derivative should be bounded requirement that is similar to one for the Taylor polynomial approximation. In the case $r = 0$ the bound on the prediction of f becomes

$$|f(x) - m(x)| \leq \frac{1}{(d+1)!} dim(\mathcal{P}^{d,n_c}) \nu_d \Lambda_{\mathcal{L}} \Delta^{d+1},$$

where Δ is the radius of the smallest ball that contains X.

Among the polynomial class of models, quadratic models ($d = 2$) have been

widely used for more than 50 years in the development of derivative-free methodologies [53], [54]. Quadratic models can be computed in different ways, including regression [55], [56] and interpolation. The computation of a fully determined quadratic interpolant requires $\frac{1}{2}(n_c+1)(n_c+2)$ samples, a quantity that adds complexity for the optimization. Several alternatives have been devised to construct less expensive approximations via undetermined interpolation, while preserving a sufficient level of accuracy [57], [58] equivalent to the fully-linear type of models. We discuss more in detail the *Least Frobenius Norm Update* strategy [59] in Section 3.2.2.

A final type of model approximation corresponds to the *Radial Basis Functions* (RBF) [60]. Given the interpolation set $X = \{x^1, x^2, \ldots, x^{\mathbf{p}}\}$ a radial basis function model is defined as

$$m(x) = \sum_{i=1}^{\mathbf{p}} b_i \varphi(\|x - x^i\|) + \varrho(x) \tag{2.2}$$

where $\varphi : \mathbb{R}_+ \to \mathbb{R}$ is a conditionally positive-definite univariate function and $\varrho : \mathbb{R}^{n_c} \to \mathbb{R}$ is a low order polynomial. Commonly used RBF functions are linear $\varphi(r) = r$, cubic $\varphi(r) = r^3$, multicuadratic $\varphi(r) = \sqrt{r^2+\gamma^2}$, inverse multiquaratic $\varphi(r) = \frac{1}{\sqrt{r^2+\gamma^2}}$ and Gaussian $\varphi(r) = exp(-\gamma r^2)$. The structure of the RBF's makes them flexible with respect to the geometric requirements of the sampling set X [61] and have been proposed to tackle global optimization problems [62], [63]. Furthermore, under certain conditions the RBF's can be certified fully-linear and used in algorithms based in the trust-region method that converge to stationary solutions [2], [64], [65].

Trust-region methods

Having discussed the different mechanisms that allow the computation of accurate surrogate representations of f, now we outline the basic derivative free trust-region approach for the unconstrained optimization case (Algorithm 2.3). This framework is easily adapted to bound constrains and other convex domains with use of a proper stationary measure. The trust-region method involves a sequence of iterations that consider the following 5 steps (apart from the initialization):

- The first one corresponds to the *model computation* (Lines 3-10) in which the

surrogate is constructed considering the current the set of samples X_k. In addition, it evaluates possible convergence to first-order stationary points (Lines 4 - 9). In some variants the check of stationary is called the *criticality step* and aims to establish an algebraic relationship between $\|\nabla_x m_k(x_k)\|$ and Δ_k, which plays an important role in assuring algorithmic convergence [66].

- The second step consists in the *candidate computation* in which a trial point is generated by minimizing the surrogate on $\hat{B}_c(x_k, \Delta_k)$. Similar to the traditional deterministic trust-region methods, the evaluation of the quality of the new candidate and the fit of the model is done according to the ρ test (Line 12), that measures the ratio in between the real and expected improvement.

- The third step considers the update and maintenance of the sampling set (Lines 13-17). Note that we refer to a geometry improvement procedure that should be able to complete the interpolation set and identify badly poised sample points.

- The fourth procedure is the *trust-region update* (Lines 18-24); we highlight that in this case the TR radius is only reduced when the model is accurate, in other words, if it is fully-linear or given the case, fully-quadratic.

- The final step is the *current candidate update* (Lines 25 - 29), note that a new candidate is only accepted when it yields sufficient objective reduction and the model presents an acceptable fit ($\rho_k \geq \eta_0 > 0$).

Observe that a candidate solution can be computed by partially optimizing inside the trust-region. For Algorithm 2.3 be convergent it is required that the optimization subproblem (Line 11) satisfies at least a fraction of the *Cauchy decrease* or the *Approximate Cauchy decrease*. The *Cauchy step* and *Approximate Cauchy point* are defined as follows:

Definition 2.4. *Let t_k^C be defined as*

$$t_k^C = \underset{t>0 | x_k - t\nabla_x m_k(x_k) \in \hat{B}_c(0,\Delta_k)}{\operatorname{argmin}} m_k(x_k - t\nabla_x m_k(x_k)).$$

Algorithm 2.3 Generic Trust-Region Algorithm

Input: Point $x_0 \in \mathbb{R}^{n_c}$, initial Trust-Region radius Δ_0, maximum Trust-Region radius Δ_{max}, model acceptance parameters $0 < \eta_0 \leq \eta_1$, geometry parameters $0 < \gamma < 1 < \gamma_{inc}$ and stationarity parameter ϵ_c

Output: Sequence $\{x_k\}$ with limiting values that are first-order stationary with respect to f

Select a set of samples $X_0 \subset \hat{B}_c(x_0, \Delta_0)$
for $k = 0, 1, 2, \ldots$ **do**
  Construct a model m_k that approximates f using the elements in X_k
  while $\|\nabla_x m_k(x_k)\| < \epsilon_c$ **do**
    if Model m_k is accurate on $\hat{B}_c(x_k, \Delta_k)$ **then**
      Set $\Delta_k = \gamma \Delta_k$
    else
      Update X_k to compute a model m_k accurate on $\hat{B}_c(x_k, \Delta_k)$
    end if
  end while
  Compute s_k by (partially) solving the problem $\min_{s \in \hat{B}_c(x_k, 0)} m_k(x_k + s)$
  Evaluate $f(x_k + s_k)$ and compute $\rho_k = \frac{f(x_k) - f(x_k + s)}{m_k(x_k) - m_k(x_k + s)}$
  if $\rho_k < \eta_1$ and m_k is not accurate in $\hat{B}_c(x_k, \Delta_k)$ **then**
    Compute X_{k+1} using a geometry improvement procedure on X_k
  else
    Set $X_{k+1} = X_k \cup \{x_k + s_k\}$. If X_k was fully-determined drop one of its previous elements
  end if
  if $\rho_k \geq \eta_1$ **then**
    Set $\Delta_{k+1} = \min\{\Delta_k \gamma_{inc}, \Delta_{max}\}$
  else if m_k is accurate on $\hat{B}_c(x_k, \Delta_k)$ **then**
    Set $\Delta_{k+1} = \Delta_k \gamma$
  else
    Set $\Delta_{k+1} = \Delta_k$
  end if
  if $\rho_k \geq \eta_0$ **then**
    Set $x_{k+1} = x_k + s_k$
  else
    Set $x_{k+1} = x_k$
  end if
end for

The corresponding Cauchy step is equivalent to

$$s_k^C = -t_k^C \nabla_x m_k(x_k).$$

Definition 2.5. *Let $\kappa_{bck} \in (0,1)$ and $\kappa_{ubs} \in (0, \frac{1}{2})$ be two given constants. For every $j \in \mathbb{Z}_+$ let the vector $x_k(j)$ be defined as*

$$x_k(j) = x_k - \kappa_{bck}^j \frac{\Delta_k}{\|\nabla_x m_k(x_k)\|} \nabla_x m_k(x_k)$$

with

$$j_C = \min\{j \in \mathbb{Z}_+ \mid m_k(x_k(j)) \leq m_k(x_k) + (\nabla_x m_k(x_k))^\top (x_k(j) - x_k)\}.$$

The approximate Cauchy point x_k^{AC} is defined as $x_k^{AC} = x_k(j_C)$.

Both the Cauchy step and the Approximate Cauchy point correspond to the (quasi) minimum of the model along the steepest descend direction inside the trust-region. Their decrease condition is bounded by:

Theorem 2.1. ***(Conn et al., [56], Theorem 10.1)*** *Consider a model of the shape $m_k(x+s) = m_k(x_k) + g^\top s + \frac{1}{2} s^\top H_k s$ and the Cauchy step from Definition 2.4, the Cauchy decrease is bounded by*

$$m_k(x_k) - m(x_k + s_k^C) \geq \frac{1}{2} \|\nabla_x m_k(x_k)\| \min\left\{\Delta_k^c, \frac{\|\nabla_x m_k(x_k)\|}{\|\nabla_{xx}^2 m_k(x_k)\|}\right\}.$$

Theorem 2.2. ***(Conn et al., [67], Theorem 6.3.19)*** *Consider a surrogate m_k. The Approximate Cauchy decrease is bounded by*

$$m_k(x_k) - m_k(x_k^{AC}) \geq \kappa_{dep} \|\nabla_x m_k(x_k)\| \min\left\{\Delta_k^c, \frac{\|\nabla_x m_k(x_k)\|}{\tilde{\beta}_k}\right\}.$$

with a constant $\kappa_{dep} \in (0,1)$ independent of k and $\tilde{\beta}_k = 1 + \max_{x \in \hat{B}_c(x_k, \Delta_k)} \|\nabla_{xx} m_k(x)\|$.

We remark that the Cauchy decrease bound from Theorem 2.1 can be only applied in linear and quadratic models; nonetheless, the approximate Cauchy decrease

is a proper substitute for other types of non-linear models. Consequently, any convergent algorithm should guarantee that

$$m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa_{fcd}}{2} \|\nabla_x m_k(x_k)\| \min\left\{\Delta_k^c, \frac{\|\nabla_x m_k(x_k)\|}{\|\nabla_{xx} m_k(x_k)\|}\right\} \tag{2.3}$$

or

$$m_k(x_k) - m_k(x_k + s_k) \geq \kappa_{facd} \|\nabla_x m_k(x_k)\| \min\left\{\Delta_k^c, \frac{\|\nabla_x m_k(x_k)\|}{\tilde{\beta}_k}\right\} \tag{2.4}$$

for some constants $\kappa_{fcd}, \kappa_{facd} \in (0, 1)$. This result and the aforementioned criticallity step -or any procedure that prevent the trust-region radius collapse when the gradient tends to zero- guarantee that both $\lim_{k\to\infty} \Delta_k = 0$ and $\lim_{k\to\infty} \|\nabla_x f(x_k)\| = 0$.

2.2 Mixed-Integer Derivative-Free Optimization

In this section, we review the methodologies for solving problem 1.1 when $n_z > 0$ and the set of discrete variables are unrelaxable. Most of these methodologies are adaptations of continuous derivative-free methods that tackle the integer constraints. Similar to the case for continuous derivative-free optimization, the solution schemes can be classified in three categories: (1) direct search methods, (2) model-based methodologies and (3) others.

In the first group we found the modification of the GPS [68] and MADS [69] algorithms to include a set of discrete search directions. One difference of both methods with the standard DDSM is the use of a third algorithmic step called the *extended poll* procedure. The *extended poll* performs additional search (polling) in the set of discrete neighborhoods whose objective value lie sufficiently close to the current objective. Porcelli and Toint [70] proposed a similar scheme on their *brute-force optimizer* (BFO); it includes a local mixed-integer search heuristic called the *recursive step* that fixes a subset of discrete coordinates and performs polling with the remaining integer variables. Abramson et al. [71] proposed a modified GPS algorithm that tackles general nonlinear constraints using a filter approach. The software package NOMAD [72] incorporates the algorithms developed by Audet et al. [68] and Abramson et al. [69].

The second group consists on methodologies that construct surrogate approximations of the objective and (simulated) constraints using polynomial or RBF interpolation over the mixed integer lattice. The method by Rashid [73] interpolates the objective and complicated constraints with multiquadratic RBF models. It also performs two surrogate optimization procedures that provide local and global information of the objective and help to maintain the quality of the interpolation model. The methods by Müller et al. [74], [75] and Müller [76] similarly use a global RBF models that interpolate the objective and constraints, and contain diverse strategies for the computation of new trial points and the preservation of the feasibility of the current candidate solution. Similar approaches are found in the work of Holmström et al. [77] and Costa and Nannincini [78].

On the other hand, the method by Newby and Ali [9] generalize Powell's Bound Optimization by Quadratic Approximation (BOBYQA) algorithm [50] for the optimization of box-constrained mixed-integer problem. Their algorithm proposal considers quadratic interpolation of the objective function and triggers termination on trust-region size. It also incorporates two variants that help to identify convergence to points under stronger definitions of mixed-integer optimality. A special case of the use of quadratic interpolation models is the method by Tran et al. [10] that aims to solve mixed-binary black-box problems that display a cyclic symmetry property. They use a tailored surrogate that is proven to be fully-linear if the vector of discrete variables is fixed, and their algorithm is proved to be convergent to a first-order stationary solution with respect to the continuous variables.

The third group consider methods that cannot be classified as DSM or model-based. Larson et al. [19] introduced a method tailored for integer convex functions based on the construction of piece-wise linear underestimators of the objective function. It can be adapted for mixed-integer optimization by solving continuous derivative-free subproblems at fixed discrete values. Liuzzi et al. [79] solve mixed-integer bound-constrained problems with an algorithm that relies on local information, their method uses a line-search algorithm to identify improvement related to the continuous variables and implements a local search method to identify improvement on a local discrete neighborhood. Their method was extended [80] to also address general nonlinear constraints using the SQP approach from Liuzzi et al

[81].

2.3 Local Optimality for Mixed-Integer Functions

One of the challenges met in the solution of mixed-integer derivative-free problems is the definition of a proper local minimizer. In purely continuous settings the definition is the following:

Definition 2.6. *If $n_z = 0$, a point x^* is a **Continuous Local Minimum** if there exists $\Delta^c > 0$ such that $f(x^*) \leq f(x) \quad \forall x \in \hat{B}_c(x^*, \Delta^c) \cap \Omega_c$.*

In contrast, for the mixed-integer case, the definition of a local minimum is less straightforward. The simplest definition is that of a *separate local minimum* (also known as a *mesh isolated solution* [7]):

Definition 2.7. *A point (x^*, y^*) is said to be a **Separate Local Minimum** (SLM) for f with respect to $\mathcal{N}_D(x^*, y^*)$ if there exists $\Delta^c > 0$ such that:*

$$f(x^*, y^*) \leq f(x, y) \quad \forall (x, y) \in (\hat{B}_c(x^*, \Delta^c) \times \{y^*\}) \cap \Omega_m$$
$$f(x^*, y^*) \leq f(x, y) \quad \forall (x, y) \in \mathcal{N}_D(x^*, y^*) \, .$$

The definition of SLM combines the definition of a continuous local minimum with an integer local minimum defined over a finite integer neighborhood $\mathcal{N}_D(x^*, y^*)$. A second definition of local optimality is that of a *stronger local minimum*:

Definition 2.8. *A point (x^*, y^*) is said to be a **Stronger Local Minimum** (StLM) for f with respect to $\mathcal{N}_D(x^*, y^*)$ if there exists $\Delta^c > 0$ such that:*

$$f(x^*, y^*) \leq f(x, y) \quad \forall (x, y) \in \Big(\bigcup_{(\tilde{x}, \tilde{y}) \in \mathcal{N}_D(x^*, y^*)} \hat{B}_c(\tilde{x}, \Delta_c) \times \{\tilde{y}\} \cap \Omega_m \Big).$$

A stronger definition is that of *combined local minimum*. It seeks to consider more information than a SLM and to be more computationally efficient to evaluate than a StLM.

Definition 2.9. *A point (x^*, y^*) is said to be a **Combined Local Minimum** (CLM) for f with respect to $\mathcal{N}_D(x^*, y^*)$ if there exists $\Delta^c > 0$ such that:*

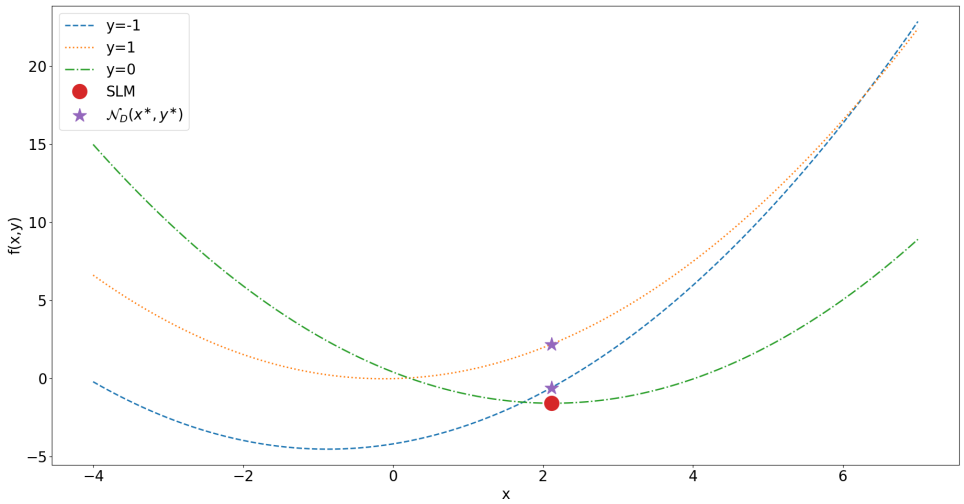
$$f(x^*, y^*) \leq f(x, y) \quad \forall (x, y) \in (\hat{B}_c(x^*, \Delta^c) \cap \Omega_c) \times \{y^*\}$$
$$f(x^*, y^*) \leq f(x, y) \quad \forall (x, y) \in \mathcal{N}_{comb}(x^*, y^*) \cup \mathcal{N}_D(x^*, y^*),$$

where $\mathcal{N}_{comb}$ is defined as the smallest local continuous minimum for which $\mathcal{N}_D$ has a point:

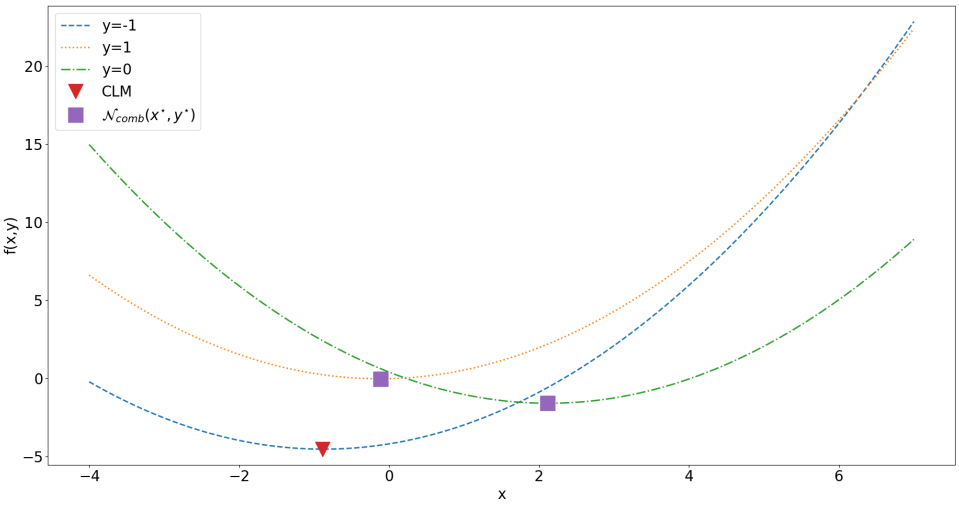
$$\mathcal{A}(\tilde{x}, \tilde{y}) := \left\{(\tilde{x}, \tilde{y}) \mid f(\tilde{x}, \tilde{y}) \leq f(x, \tilde{y}) \ \ \forall x \in \{\hat{x} \in \hat{B}_c(\tilde{x}, \Delta^c) \mid (\hat{x}, \tilde{y}) \in \Omega_m\}\right\}$$
$$\mathcal{N}_{comb}(x^*, y^*) := \left\{ \underset{(x,y)\in\mathcal{A}(\tilde{x},\tilde{y})}{\operatorname{argmin}} \ \{f(x, y)\} \mid (\tilde{x}, \tilde{y}) \in \mathcal{N}_D(x^*, y^*) \setminus (x^*, y^*) \right\}.$$

We illustrate how a point can be classified as a minimizer under one definition of local optimality but fails to be certified as such under a stronger optimality definition. Figure 2.1 represents the function $f(x, y)$ with $x \in \mathbb{R}$ and $y \in \{-1, 0, 1\}$. Figure 2.1a shows that the point marked with a circle $(x^* = 2.11, y^* = 0)$ is a SLM. This point is a local continuous minimizer when $y = 0$ and it presents the best objective among the points in the set $\mathcal{N}_D(x^*, y^*)$, which is represented by the stars. However, it is evident that this point is not a StLM nor a CLM as there exists a range of points $x \in (-4, 1),\ y = -1$ with smaller objective value. Figure 2.1b shows that the point marked with a triangle $(x^\star = -0.889, y^\star = -1)$ is a CLM. It is the local continuous minimizer when $y = -1$ and presents the best objective among the set of local minimizers $\mathcal{N}_{comb}(x^\star, y^\star)$.

Table 2.1 shows different optimality definitions considered by previous studies in mixed-integer derivative-free optimization. It is important to note that not all the studies implement an optimality criterion, and algorithmic convergence may more simply depend on distance and/or number of sampled points.



(A) Example of a separate local minimum



(B) Example of a combined local minimum

FIGURE 2.1: Examples of local optimality

Method	**Optimality criterion**	**I. Neighborhood**
Wah et al. [82]	Sepatate local minimum	$\mathcal{N}_D$
Lucidi et al. [83]	Stronger local mimimum	$\mathcal{N}_D$
Liuzzi et al. [79]	Separate local minimum	$\|y - \hat{y}\|_1$
Abrahmson et al. [69]	Stronger separate minimum	$\mathcal{N}_D$
Holmström et al. [77]	*Number of samples*	-
Larson et al. [19]	Global optima	-
Muller [76]	*Distance*	-
Costa and Naninncini [78]	*Number of samples*	-
Newby and Ali [9]	Combined local minimum	$\mathcal{N}_D$
Tran et al. [10]	Separate local minimum	$\{0,1\}^{n_z}$

TABLE 2.1: Examples of optimality criterion

In the literature, convergence to a mixed-integer local solution is typically attained by extensive search on multiple manifolds; it may even require using a continuous derivative-free optimizer to guarantee a CLM or even a SLM, e.g., [9]. We highlight that the DDSM are well suited for achieving convergence to SLM and StLM as they use data on surrounding manifolds to perform the *extended poll* step. Note that model-based methods normally do not include a convergence check, preferring alternate stopping criteria. In this work we introduce two model-based methods that converge to SLM and StLM under given conditions. We remark that we do not aim for convergence to CLM as the number of samples to evaluate this type of minimizer is intensive in terms on function evaluations.

Chapter 3

Locally Quadratic Mixed-Integer Functions and Approximations

Generally speaking, trust region methods are based on the assumption that it is possible to construct a model m to approximate a function f within a given (trust) region [67]. In particular, under some assumptions it is possible to construct a surrogate model for the function f inside the region $\hat{B}_c(x, \Delta)$ with the property that the approximation error for f and its derivatives is bounded. Using a Taylor-like model, the error bounds are directly proportional to Δ. This property enables the development of algorithms to find stationary points and proper local minimizers. Usually, the model is obtained via interpolation/regression of a set of sampled points [51]. From a practical point of view, it is important that the number of points sampled to obtain the model is bounded. The notions of fully-linear models are used to prove convergence to first-order stationary points of trust region methods.

In the general mixed-integer setting, it is difficult to construct fully-linear (and fully-quadratic) approximations, because functions defined over a mixed-integer set are discontinuous with respect to the discrete variables. There are two problems to overcome: the lack of error bounds on the interpolation over integer variables (due to discontinuity), and the difficulty of modeling the interaction between continuous and integer variables.

3.1 LQMI Function Definition

To extend a trust-region algorithm to the mixed-integer setting and overcome these problems, we introduce a structured class of mixed-integer functions for which we can construct bounded-error approximations. We state the characteristics of these functions as assumptions.

Assumption 3.1. *For every $(x^*, y^*) \in \Omega_m$ there exists a unique function $\tau(y)$, and not necessarily unique $\tilde{A}_z \in \mathbb{R}^{n_z \times n_z}$ and $\tilde{l}_z \in \mathbb{R}^{n_z}$, such that for every $(x, y) \in \mathcal{N}_1(x^*, y^*)$*

$$f(x,y) = \tau(y - y^*) := (y - y^*)^\top \tilde{A}_z (y - y^*) + (\tilde{l}_z)^\top (y - y^*) + f(x^*, y^*).$$

Although $\tau(y)$ is uniquely defined, multiple distinct $\tilde{l}_z, \tilde{A}_z$ could define the same τ. Assumption 3.1 states that, if we fix the vector of continuous variables to x^*, $f(x,y)$ reduces to a quadratic approximation τ centered at (x^*, y^*). Note that the linear term $\tilde{l}_z$ and the quadratic matrix $\tilde{A}_z$ can depend on (x^*, y^*). Exploiting this structure, we can determine the number of samples needed to compute $\tilde{A}_z$ and $\tilde{l}_z$:

Observation 3.1. *$\tilde{A}_z$ and $\tilde{l}_z$ can be obtained by sampling $\mathcal{O}(n_z^2)$ quadratically independent points in $\mathcal{N}_1(x^*, y^*)$ [51]. The exact number of points depends on whether the point y^* is located on the boundary of the box Ω_z, or not.*

Assumption 3.2. *There exists a radius $\Delta^c > 0$ such that, for every $(x^*, y^*) \in \Omega_m$, there exists a set of functions*

$$\phi_j : \mathbb{R}^{n_c} \to \mathbb{R}, \;\; \phi_j(0) = 0 \quad \forall j \in \{0, \dots, n_z\} \tag{3.1}$$

satisfying

$$f(x,y) = \tau(y - y^*) + \phi_0(x - x^*) + \sum_{j=1}^{n_z} \phi_j(x - x^*)(y_j - y_j^*) \quad \forall (x,y) \in \hat{B}_v(x^*, y^*, \Delta^c, 1)\,. \tag{3.2}$$

To have a more compact notation, we introduce $\phi(x)$, $\phi : \mathbb{R}^{n_c} \longrightarrow \mathbb{R}^{n_z}$ as the vector of functions $\phi_j(x)$ (with index $j \neq 0$), this allows one to rewrite $f(x,y) = \tau(y - y^*) + \phi_0(x - x^*) + (y - y^*)^\top \phi(x - x^*)$. Assumption 3.2 completes Assumption 3.1, stating that the interaction between continuous and discrete variables is

bilinear in the vector $\phi(x)$ and the discrete variables y. The Assumptions given so far allows us to define a class of black box functions:

Definition 3.1. *A function f satisfying assumptions 1.1, 1.2, 3.1 and 3.2 is called a **Locally-Quadratic Mixed-Integer (LQMI) function**.*

Assumption 3.2 implies that the derivative of an LQMI function with respect to a continuous variable can be computed as follows:

Observation 3.2. *The partial gradient f with respect to the continuous variables is given by:*

$$\begin{aligned}\nabla_x f(x,y) &= \nabla_x \phi_0(x) + \sum_{j=1}^{n_z} \nabla_x \phi_j(x)(y_j - y_j^*) \\ &= \nabla_x \phi_0(x) + (J\phi(x))^\top (y - y^*)\end{aligned}$$

Observation 3.3. *The partial gradient of f with respect to the continuous variables is given by:*

$$\begin{aligned}\nabla_x f(x,y) &= \nabla_x \phi_0(x) + \sum_{j=1}^{n_z} \nabla_x \phi_j(x)(y_j - y_j^*) \\ &= \nabla_x \phi_0(x) + (J\phi(x))^\top (y - y^*)\end{aligned}$$

Now, observing the affinity of differentials and the quadratic integer representation of LQMI functions, we introduce the definition of *mixed-integer fully-linear models*. Similar to fully linear/quadratic models, such approximations provide an error bound based on a continuous trust-region radius. In the definition of mixed-integer fully-linearity, we use an integer set $\Omega_d \subseteq \Omega_z$ that belongs to some class of discrete sets. In the next section we prove that is always possible to construct an accurate model for an LQMI function with respect to the class of discrete sets $\{\Omega_d^{LQMI}(y) \mid \forall y \in \Omega_z\}$, where

$$\Omega_d^{LQMI}(y) = \hat{B}_z(y,1) \cap \Omega_z. \tag{3.3}$$

To assess convergence to stationary points we require a structure that guarantees that a local surrogate m with uniformly good accuracy can be constructed. In other

words, we seek a method that computes surrogate approximations centered in any point of the domain with the same pair of error constants. In the LQMI setting we say that model m belongs to a *mixed-integer fully-linear* class of models:

Definition 3.2. *Let f be a function which satisfies Assumptions 1.1 and 1.3. A set of functions $\mathcal{M} = \{m : [\mathbb{R}^{n_c} \times \mathbb{Z}^{n_z}] \to \mathbb{R}\}$ is called a **mixed-integer fully-linear class** of models with respect to function f and the class of discrete sets $\bar{\Omega}$ for a given Δ_{max} if:*

1. *There exist two constants $\kappa_{eg}, \kappa_{ef} > 0$, such that, for every $(\bar{x}, \bar{y}) \in \Omega_m$, $\Omega_d \in \bar{\Omega}$ and $\Delta \in [0, \Delta_{max}]$, there exists a model $m \in \mathcal{M}$ satisfying:*

$$|f(x,y) - m(x,y)| \leq \kappa_{ef}\Delta^2 \;\; \forall (x,y) \in \hat{B}_c(\bar{x}, \Delta) \times (\Omega_d \cup \{\bar{y}\})$$
$$\|\nabla_x f(x,y) - \nabla_x m(x,y)\| \leq \kappa_{eg}\Delta \;\; \forall (x,y) \in \hat{B}_c(\bar{x}, \Delta) \times (\Omega_d \cup \{\bar{y}\}).$$

2. *For this class $\mathcal{M}$ there exists an algorithm, which we call a "model improvement" algorithm, that in a finite, uniformly bounded (with respect to $\bar{x}, \bar{y}$ and Δ) number of steps can:*
   - *establish if a model $m \in \mathcal{M}$ is fully-linear in $(x,y) \in \hat{B}_c(\bar{x}, \Delta)(\Omega_d \cup \{\bar{y}\})$, or*
   - *compute a new model $\tilde{m} \in \mathcal{M}$ which is fully-linear in $(x,y) \in \hat{B}_c(\bar{x}, \Delta) \times (\Omega_d \cup \{\bar{y}\})$.*

   for every $\Omega_d \in \bar{\Omega}$.

The class of discrete sets $\bar{\Omega}$ determines the neighborhood with respect to which the point computed by our algorithm is stationary.

3.2 Mixed-Integer Fully-Linear Models

When working with LQMI functions, it is natural to approximate $f(x,y)$ with a class $\mathcal{M}$ of models with the following structure:

$$m(x^* + s_c, y^* + s_z) := f(x^*, y^*) + l_c^\top s_c + l_z^\top s_z + s_z^\top A_z s_z + s_c^\top A_M s_z. \tag{3.4}$$

which mimics a truncated first-order Taylor expansion of the function $f(x,y)$ with respect to the continuous variables x. In the above expression, $m(x^* + s_c, y^* + s_z)$

consists of a constant term, representing the value of $f(x,y)$ at the point where the approximation is centered, a linear term in the continuous variables $l_c^\top s_c$, a quadratic term in the discrete variables $l_z^\top s_z + s_z^\top A_z s_z$ and a bilinear term $s_c^\top A_M s_z$. Recall that the quadratic terms l_z and A_z can be computed by sampling a polynomial number of points (see Observation 3.1), which can be identified using algorithms designed to ensure the well poisedness of a set of interpolation points [51]. What remains to identify are l_c and A_M. In this section we show how to obtain such parameters to have a model m that is mixed-integer fully linear with respect to an LQMI function and the class of sets $\bar{\Omega} = \{\Omega_d^{LQMI}(y) \mid \forall y \in \Omega_z\}$. Moreover, we prove that such model approximation is also mixed-integer fully linear with respect to a non-LQMI function in a class of discrete sets $\Omega_d^G(y, M_k)$ with the property that $|\Omega_d^G(y,\cdot)| = n_z + 1$ (see Section 3.2.3). Having a model that is mixed-integer fully linear is crucial to prove the convergence of the trust-region algorithm to a stationary point presented in Chapter 4.

The procedure proposed to construct the model m takes advantage of the fact that the set of points required to determine l_z and A_z is not unique: the samples can be chosen among any suitable set of integer directions within the neighborhood $\mathcal{N}_1(x^*, y^*)$. We take advantage of this degree of freedom by choosing, at each iteration, the manifold that allows us to reuse the maximum amount of previous sampled points, and satisfies the bound constraints for the integer variables. We therefore rely on suitably defined *integer generating vectors*, i.e., coordinate shifts that facilitate model computation by allowing the efficient reuse of samples generated in previous iterations, which are defined as follows:

Definition 3.3. *The matrix $M = \left[M^1, ..., M^{n_z}\right] \in \mathbb{Z}^{n_z \times n_z}$ is a matrix of **integer generating vectors** if it has the following properties: $M^j \in \{-1, 0, 1\}^{n_z}$ and M is a basis of vectors that spans $\mathbb{R}^{n_z}$. Let $\mathscr{M}(y)$ be the collection of all matrices of integer generating vectors at point $y \in \Omega_z$. The number of elements of $\mathscr{M}(y)$ depend on whether the point y lays on the boundary of Ω_z or not.*

Definition 3.3 is written in such a way that, for any set of integer generating vectors $M \in \mathscr{M}(y^*)$ and point $(x,y) \in \mathcal{N}_1(x^*, y^*)$, there exists a vector $d \in \mathbb{R}^{n_z}$ that allows rewriting (x,y) in terms of the vectors in M: $(x,y) = (x^*, Md + y^*)$.

Definition 3.4. *For a given $M \in \mathscr{M}(y^*)$, we define the function $\psi^M : \mathbb{R}^{n_c} \to \mathbb{R}^{n_z}$ as:* $\psi^M(x) := M^\top \phi(x)$.

Therefore, the function f can be rewritten as: $f(x,y) = \tau(y-y^*) + \phi_0(x-x^*) + (y-y^*)^\top (M^\top)^{-1}\psi^M(x-x^*)$. To make the notation less cumbersome, we remove the superscript "M" and we simply use ψ when it is clear from the context.

Definition 3.5. *The set of **integer generating points** $Q(y,M) \subset \Omega_z$, associated with a matrix M and integer point $y \in \mathbb{Z}^{n_z}$, is defined as*

$$Q(y,M) = \left\{ q^1, \ldots, q^{n_z} |\, q^j = y + M^j \right\}.$$

The set of generating points $Q(y,M)$ defines a set of neighboring points based on M and "centered" at y. This allows us to simplify the expression for the properties of f as follows.

Observation 3.4. *For any $M \in \mathscr{M}(y^*)$ and point (x^*, y^*) we have:*

$$f(x^*, q^j) = \tau(M^j) \quad \forall q^j \in Q(y^*, M)$$
$$f(x + s_c, q^j) = \tau(M^j) + \phi_0(s_c) + \psi_j(s_c) \quad \forall q^j \in Q(y^*, M), s_c \in \hat{B}_c(0, \Delta^c)$$

Next, we show how a mixed-integer fully-linear approximation can be constructed in the LQMI setting. The procedure involves the computation of the continuous term l_c and the interaction term A_M independently, using known methodologies devised for continuous derivative-free optimization.

3.2.1 Model Computation and Fully-Linearity in the LQMI Setting

Given the separable structure of an LQMI function and the linear transformations described in Definition 3.4 we propose the following surrogate model to be used in a trust-region algorithm:

$$m(x,y) = \tau(y-y^*) + \hat{m}_0(x-x^*) + (y-y^*)^\top (M^\top)^{-1}\hat{v}(x-x^*), \tag{3.5}$$

where $\tau(y - y^*)$ is described as in Assumption 3.1 and $\hat{m}_0$ and $\hat{v}$ are linear approximations of the functions ϕ_0 and ψ^M, respectively. We now prove that if both approximations are fully-linear in continuous neighborhoods, the mixed-integer approximation m is fully-linear as well.

Proposition 3.1. *[51] A fully-linear approximation $\hat{m}_0(x - x^*) := (x - x^*)^\top g^0$ of $\phi_0(x)$ on $\hat{B}_c(x^*, \Delta^c)$ with $g^0 \in \mathbb{R}^{n_c}$ and error bound coefficients $\kappa_{ef,0}$, $\kappa_{eg,0}$ can be computed by sampling $n_c + 1$ points $(x, y) \in \hat{B}_c(x^*, \Delta^c) \times \{y^*\}$.*

Proposition 3.2. *[51] Let $q^j \in Q(y^*, M)$ be as in Definition 3.5. A fully-linear approximation $\hat{m}_j^M(x)$ of $f(x, q^j)$ with error bound coefficients κ_{ef_j} and κ_{eg_j} can be computed by sampling $n_1 + 1$ points $(x, y) \in \hat{B}_c(x^*, \Delta^c) \times \{q_j\}$.*

Note that Proposition 3.2 allows us to isolate the entry $\psi_j(x)$; by Proposition 3.1, $\hat{m}_j^M(x)$ is equivalent to a linear approximation of the function $\phi_0(x) + \psi_j(x)$.

Lemma 3.1. *The function $\hat{v}_j(x) := \hat{m}_j^M(x) - \hat{m}_0(x)$, which can be expressed as $\hat{v}_j(x - x^*) = (x - x^*)^\top g^j$ $(j \in 1, \dots, n_z)$ with $g^j \in \mathbb{R}^{n_c}$, is a fully-linear approximation of $\psi_j^M(x)$ on $x_c \in \hat{B}_c(x^*, \Delta)$ with constants $\hat{\kappa}_{ef,j} := \kappa_{ef,j} + \kappa_{ef,0}$ and $\hat{\kappa}_{eg,j} := \kappa_{eg,j} + \kappa_{eg,0}$.*

Proof. We first show that the error on the function is bounded:

$$
\begin{aligned}
&|\psi_j(x) - \hat{v}_j(x)| \\
&= |\psi_j(x) + \phi_0(x) - \hat{m}_j^M(x) - \phi_0(x) + \hat{m}_0(x)| \\
&\leq |\psi_j(x) + \phi_0(x) - \hat{m}_j^M(x)| + |\hat{m}_0(x) - \phi_0(x)| \\
&\leq \kappa_{ef,j}\Delta^2 + \kappa_{ef,0}\Delta^2 = \hat{\kappa}_{ef,j}\Delta^2 \,.
\end{aligned}
$$

Also the error on the function gradient is bounded:

$$
\begin{aligned}
&\|\nabla_x \psi_j(x) - \nabla_x \hat{v}_j(x)\| \\
&= \|\nabla_x \psi_j(x) + \nabla_x \phi_0(x) - \nabla_x \hat{m}_j^M(x) - \nabla_x \phi_0(x) + \nabla_x \hat{m}_0(x)\| \\
&\leq \|\nabla_x \psi_j(x) + \nabla_x \phi_0(x) - \nabla_x \hat{m}_j^M(x)\| + \|\nabla_x \hat{m}_0(x) - \nabla_x \phi_0(x)\| \\
&\leq \kappa_{eg,j}\Delta + \kappa_{eg,0}\Delta = \hat{\kappa}_{eg,j}\Delta.
\end{aligned}
$$

□

Definition 3.6. *Let A_g be the matrix formed by the vectors $g^j, \forall j \in \{1, \dots, n_z\}$ defined in Lemma 3.1:*

$$A_g = \begin{bmatrix} | & | & & | \\ g^1 & g^2 & \dots & g^{n_z} \\ | & | & & | \end{bmatrix}.$$

We emphasize that the model introduced throughout equation 3.5 is equivalent to the one introduced in equation 3.4, where $l_c = g^0$ and $A_M = (M^\top)^{-1} A_g$.

Theorem 3.1. *Let $\hat{m}_0(s_c)$ and $\hat{v}(s_c)$ be obtained as in Proposition 3.1 and Lemma 3.1. For any $M \in \mathscr{M}$, the model $m(x^* + s_c, y^* + s_z)$ defined in Equation (3.4) with $l_c = g^0$ and $A_M = (M^\top)^{-1} A_g$ is mixed-integer fully linear with respect to $f(x, y)$ in the trust-region $\hat{B}_v(x, y, \Delta^c, 1)$, with error bound constants $\bar{\kappa}_{ef} = \kappa_{ef,0} + (n_z)^{1/2} \cdot \|M^{-1}\| \cdot \|\hat{\kappa}_{ef}\|$ and $\bar{\kappa}_{eg} = \kappa_{eg,0} + (n_z)^{1/2} \cdot \|M^{-1}\| \cdot \|\hat{\kappa}_{eg}\|$.*

In the remainder of this section, we provide the intermediate results necessary to prove Theorem 3.1.

Observation 3.5. *Note that the gradient of a model m centered at $(\bar{x}, \bar{y})$ in the point $(x, y) = (\bar{x} + s_c, \bar{y} + s_z)$ can be expressed as follows:*

$$\nabla_x m(x, y) = \nabla_x \hat{m}_0(s_c) + (J\hat{v}(s_c))^\top M^{-1} s_z.$$

Lemma 3.2. *$|s_z^\top (M^\top)^{-1}(\psi(s_c) - \hat{v}(s_c))| \leq (n_z)^{1/2} \|M^{-1}\| \cdot \|\hat{\kappa}_{ef}\| \cdot (\Delta^c)^2$, where $\hat{\kappa}_{ef} \in \mathbb{R}^{n_c}$ is the vector with entries $\hat{\kappa}_{ef,j}$ defined in Lemma 3.1.*

Proof. By the Cauchy–Schwarz inequality

$$|s_z^\top (M^\top)^{-1}(\psi(s_c) - \hat{v}(s_c))| \leq \|s_z^\top (M^\top)^{-1}\| \cdot \|(\psi(s_c) - \hat{v}(s_c))\|.$$

An upper bound to the first term is computed as follows:

$$\|s_z^\top (M^\top)^{-1}\| \leq \|s_z\| \cdot \|(M^\top)^{-1}\| \leq (n_z)^{1/2} \|M^{-1}\|.$$

For the second term, a upper bound is computed by mixed-integer linearity of the vector function $\psi(s_c)$:

$$\begin{aligned}\|(\psi(s_c) - \hat{v}(s_c))\| &= \left(\sum_{j=1}^{n_z} (\psi_j(s_c) - \hat{v}_j(s_c))^2\right)^{1/2} \\ &\leq \left(\sum_{j=1}^{n_z} \hat{\kappa}_{ef,j}^2\right)^{1/2} \cdot (\Delta^c)^2 = \|\hat{\kappa}_{ef}\| \cdot (\Delta^c)^2.\end{aligned}$$

□

Lemma 3.3. $\|(J\psi(s_c) - J\hat{v}(s_c))^\top M^{-1} s_z\| \leq (n_z)^{1/2} \|M^{-1}\| \cdot \|\hat{\kappa}_{eg}\| \cdot \Delta^c$, *where* $\hat{\kappa}_{eg} \in \mathbb{R}^{n_c}$ *is the vector with entries* $\hat{\kappa}_{eg,j}$ *defined in Lemma 3.1.*

Proof. By the Cauchy–Schwarz inequality

$$\|(J\psi(s_c) - J\hat{v}(s_c))^\top M^{-1} s_z\| \leq \|J\psi(s_c) - J\hat{v}(s_c)\| \cdot \|M^{-1} s_z\|.$$

The bound for the norm of the difference of Jacobians is computed as follows:

$$\|J\psi(s_c) - J\hat{v}(s_c)\| = \left(\sum_{i=1}^{n_z} \|\nabla_x \psi_i(s_c) - \hat{v}_i(s_c)\|^2\right)^{1/2}.$$

Recalling the fully-linear assumption on every model $\hat{v}_i(s_c) \;\; \forall i \in \{1, \ldots, n_z\}$

$$\|J\psi(s_c) - J\hat{v}(s_c)\| \leq \left(\sum_{i=1}^{n_z} \hat{\kappa}_{eg,i}^2 (\Delta^c)^2\right)^{1/2} = \|\hat{\kappa}_{eg}\| \cdot \Delta^c \,.$$

□

With all the provided proposition, we are able to give the proof of Theorem 3.1:

Proof. First, we prove that for all (s_c, s_z) in $\hat{B}_v(0, 0, \Delta^c, 1)$ the error in the function is bounded by $\bar{\kappa}_{ef} \Delta^2$. The absolute error of the approximation is given by:

$$\begin{aligned}&|f(x + s_c, y + s_z) - (\tau(s_z) + \hat{m}_0(s_c) + s_z^\top (M^\top)^{-1} \hat{v}(s_c))| \\ &= |\tau(s_z) + \phi_0(s_c) + s_z^\top (M^{-1})^\top \psi(s_c) - (\tau(s_z) + \hat{m}_0(s_c) + s_z^\top (M^\top)^{-1} \hat{v}(s_c))| \\ &= |\phi_0(s_c) - \hat{m}_0(s_c) + s_z^\top (M^\top)^{-1} (\psi(s_c) - \hat{v}(s_c))| \\ &\leq |\phi_0(s_c) - \hat{m}_0(s_c)| + |s_z^\top (M^\top)^{-1} (\psi(s_c) - \hat{v}(s_c))|.\end{aligned}$$

Recalling Lemma 3.2 stating that $|s_z^\top (M^\top)^{-1}(\psi(s_c) - \hat{v}(s_c))| \leq (n_z)^{1/2}\|M^{-1}\| \cdot \|\hat{\kappa}_{ef}\| \cdot (\Delta^c)^2$ and the definition of constant $\bar{\kappa}_{ef}$, we have that

$$\begin{aligned} &|f(x+s_c, y+s_z) - m(x+s_c, y+s_z)| \\ \leq \kappa_{ef,0}(\Delta^c)^2 &+ (n_z)^{1/2} \cdot \|M^{-1}\| \cdot \|\hat{\kappa}_{ef}\| \cdot (\Delta^c)^2 \leq \bar{\kappa}_{ef}(\Delta^c)^2. \end{aligned}$$

Now, we prove that the error in the continuous gradient is bounded by $\bar{\kappa}_{eg}\Delta^c$. The gradient deviation is given by:

$$\begin{aligned} &\|\nabla_x f(s_c) - (\nabla_x \hat{m}_{0,k}(s_c) + (J\hat{v}(s_c))^\top M^{-1} s_z)\| \\ = &\|\nabla_x \phi_0(s_c) + (J\psi(s_c))^\top M^{-1} s_z - (\nabla_x \hat{m}_0(s_c) + (J\hat{v}(s_c))^\top M^{-1} s_z)\| \\ \leq &\|\nabla_x \phi_0(s_c) - \nabla_x \hat{m}_0(s_c)\| + \|(J\psi(s_c) - J\hat{v}(s_c))^\top M^{-1} s_z\|. \end{aligned}$$

Recalling Lemma 3.3 that states that $\|(J\psi(s_c) - J\hat{v}(s_c))^\top M^{-1} s_z\| \leq (n_z)^{1/2}\|M^{-1}\| \cdot \|\hat{\kappa}_{eg}\| \cdot \Delta^c$ and the definition of $\bar{\kappa}_{eg}$, we have that:

$$\begin{aligned} &\|\nabla_x f(x+s_c, y+s_z) - \nabla_x m(x+s_c, y+s_z)\| \\ \leq \kappa_{eg,0}\Delta^c &+ (n_z)^{1/2} \cdot \|M^{-1}\| \cdot \|\hat{\kappa}_{eg}\| \cdot \Delta^c \leq \bar{\kappa}_{eg}\Delta^c. \end{aligned}$$

□

3.2.2 Practical Model Considerations

The existence of fully-linear model approximations of LQMI functions is necessary to prove convergence of a trust-region algorithm to stationary points; however, it might be desirable to build model approximations involving fewer objective value evaluations. In the continuous derivative-free methodology this is done by considering underdetermined quadratic interpolation and using an adaptative (but bounded) number of samples each time the model construction procedure is invoked [56]. For the LQMI setting we propose two ways to decrease the number of samples required to build and update a surrogate approximation.

The first one is to perform underdetermined approximation of each element constituting the model. The terms involving the continuous directions l_c and A_M can be approximated using a fraction of the points $n_c + 1$ required for the interpolation. The

elements related to the integer contribution in the objective function l_z and A_z can be estimated by solving the *Least Frobenius Norm Update* (see below) interpolation problem for the set of samples $Z \subset \Omega_z$:

$$\min_{l_z, A_z} \|A_z - A_z^0\|^2 \tag{3.6a}$$

$$\text{s.t. } f(x^*, y^*) + l_z^\top (y - y^*) + (y - y^*)^\top A_z (y - y^*) = f(x^*, y) \quad \forall y \in Z \tag{3.6b}$$

where $A_z^0 \in \mathbb{R}^{n_z} \times \mathbb{R}^{n_z}$ is a previous approximation of the matrix A_z and $|Z| \in [(n_z + 1), \frac{1}{2}(n_z + 1)(n_z + 2)]$. The second way to reduce the number of samples is to update a model that had a sufficiently good fit in a previous iteration k, using the information of the iterate $(x_k + s_c^k, y_k + s_z^k)$. We assume the quadratic terms A_M^k of the model remain constant, but the linear terms are affected by the translation:

$$l_c^{k+1} = l_c^k + A_M^k s_z^k \tag{3.7a}$$

$$l_z^{k+1} = l_z^k + (A_M^k)^\top s_c^k + 2A_z^k s_z^k. \tag{3.7b}$$

These two concepts allow us to devise several algorithm variants, based on the number of samples devoted to model construction and to an inexpensive update of the surrogate; these inexpensive model updates are only considered in the case of *successful* iteration. We highlight that the *Least Frobenius Norm* update 3.6a-3.6b is defined in a larger integer neighborhood, not necessarily $\Omega_d^{LQMI}(y^*)$, to enable a better initial exploration of the integer domain.

3.2.3 Mixed-Integer Fully-Linearity in the non-LQMI Setting

Finally, we show that the methodology described in Section 3.2.1 is able to generate a fully linear approximation with respect to the discrete set $\Omega_d^G(y^*, M) = \{y^*\} \cup Q(y^*, M)$ for a function that satisfies Assumptions 1.1 and 1.2, whether it behaves as an LQMI function or not.

Proposition 3.3. *Let $q_j \in Q(y^*, M)$ be as in Definition 3.5. A fully linear approximation $\hat{m}_j^M(x)$ of $f(x, q_j) - f(x^*, q_j)$ with coefficients κ_{ef_j} and κ_{eg_j} can be computed by sampling $n_c + 1$ points $(x, y) \in \hat{B}_c(x^*, \Delta^c) \times \{q_j\}$.*

Let $\hat{m}_0(s_c)$ and $\hat{m}_j^M(x)$ be obtained as in Propositions 3.1 and 3.3, respectively. Let

$$\hat{v}_j(x) := \hat{m}_j^M(x) - \hat{m}_0(x) := (x - x^*)^\top g^j \quad (j \in 1, \dots, n_z) \tag{3.8}$$

and let

$$\bar{\tau}(y - y^*) = f(x^*, y^*) + \bar{l}_z^\top (y - y^*) + (y - y^*)^\top \bar{A}_z (y - y^*) \tag{3.9}$$

be a quadratic interpolation of function $f(x, y)$ in the neighborhood $\{x^*\} \times \hat{B}_z(y^*, 1)$, which is obtained by solving problem 3.6a-3.6b over a set of samples Z that includes the set of points $\Omega_d^G(y^*, M)$. A direct result of equations 3.9 and 3.6b is that $f(x^*, y) = \bar{\tau}(y - y^*) \ \ \forall y \in \Omega_d^G(y^*, M)$.

Theorem 3.2. *Let f be a function that satisfies Assumptions 1.1 and 1.2. Let $m(x^* + s_c, y^* + s_c)$ be a surrogate approximation $\hat{m}_0(s_c)$, with the shape 3.5, where $\hat{v}(s_c)$ are obtained as in Proposition 3.1 and equation 3.8. Let $\bar{\tau}(s_z)$ be obtained as in equation 3.9. For any $M \in \mathscr{M}$, the model $m(x^* + s_c, y^* + s_z)$ defined in equation (3.4) with*

$$\begin{aligned} (l_z, A_z) &= (\bar{l}_z, \bar{A}_z) \\ l_c &= g^0 \\ A_M &= (M^\top)^{-1} A_g \end{aligned}$$

is mixed-integer fully linear with respect to $f(x, y)$ in the trust-region $\hat{B}_c(x^*, \Delta^c) \times \Omega_d^G(y^*, M)$ with constants:

- $\bar{\kappa}_{ef} = \max(\kappa_{ef_0}, \max_{j \in \{1,\dots,n_z\}}(\hat{\kappa}_{ef,j}))$
- $\bar{\kappa}_{eg} = \max(\kappa_{eg_0}, \max_{j \in \{1,\dots,n_z\}}(\hat{\kappa}_{eg,j}))$

Proof. We remark that this proof is equivalent to the one for Theorem 3.1, the main difference between the two is that in this case we restrict the error analysis on the set $\Omega_d^G(y^*, M)$, instead of $\Omega_d^{LQMI}(y^*)$. It is evident that if the model and gradient error are bounded on every discrete manifold $y \in \Omega_d^G(y^*, M)$, the error constants $\bar{\kappa}_{ef}, \bar{\kappa}_{eg}$ for $m(x, y)$ are equal to $\max(\kappa_{ef_0}, \max_{j \in \{1,\dots,n_z\}}(\hat{\kappa}_{ef,j}))$ and $\max(\kappa_{eg_0}, \max_{j \in \{1,\dots,n_z\}}(\hat{\kappa}_{eg,j}))$, respectively.

First, we prove that for all (s_c, s_z) in $\hat{B}_c(0, \Delta^c) \times \{M^j \mid j \in \{1, \dots, n_z\}\}$ the error in the function is bounded by $\bar{\kappa}_{ef}\Delta^2$. We recall Proposition 3.3 for the construction of

model approximation $\hat{m}_j^M(s_z)$

$$|\hat{m}_j^M(s_c) - (f(x + s_c, y^* + M^j) - f(x^*, y^* + M^j))| \leq \hat{\kappa}_{ef,j}(\Delta^c)^2.$$

Then, we add and subtract the term $\hat{m}_0(s_c)$ to rewrite the inequality in terms of $\hat{v}_j(s_c)$ (from Proposition 3.8):

$$|\hat{v}_j(s_c) + \hat{m}_0(s_c) + f(x^*, y^* + M^j) - f(x^* + s_c, y^* + M^j)| \leq \hat{\kappa}_{ef,j}(\Delta^c)^2.$$

From Proposition 3.9, $f(x^*, y^* + M^j) = \hat{\tau}(M^j)$ and $M^{-1}M^j v(s_c) = v_j(s_c)$:

$$|(M^j)^\top (M^\top)^{-1}\hat{v}(s_c) + \hat{m}_0(s_c) + \hat{\tau}(M^j) - f(x^* + s_c, y^* + M^j))| \leq \hat{\kappa}_{ef,j}(\Delta^c)^2.$$

The latter is equivalent to the definition of model m in the point $(x^* + s_c, y^* + M^j)$:

$$|m(x^* + s_c, y^* + M^j) - f(x^* + s_c, y^* + M^j)| \leq \hat{\kappa}_{ef,j}(\Delta^c)^2 \leq \bar{\kappa}_{ef}(\Delta^c)^2.$$

Now we prove that for all (s_c, s_z) in $\hat{B}_c(0, \Delta^c) \times \{M^j \mid \forall j \in \{1, \ldots, n_z\}\}$ the error in the approximation of the continuous gradient is bounded by $\bar{\kappa}_{eg}\Delta^c$:

$$\|\nabla_x \hat{m}_j^M(s_c) - \nabla_x f(x + s_c, y^* + M^j)\| \leq \hat{\kappa}_{ef,j}\Delta^c.$$

Now, adding and subtracting the term $\nabla_x \hat{m}_0(s_c)$ allows to rewrite the inequality in terms of $\hat{v}_j(s_c)$ (from Proposition 3.8):

$$\|\nabla_x \hat{v}_j(s_c) + \nabla_x \hat{m}_o(s_c) - \nabla_x f(x + s_c, y^* + M^j)\| \leq \hat{\kappa}_{ef,j}\Delta^c.$$

From Definition 3.4 we expand the expression in terms of the Jacobian matrix of vector $\hat{v}(s_c)$, $\|(J\hat{v}(s_c))^\top M^{-1}M^j + \nabla_x \hat{m}_o(s_c) - \nabla_x f(x + s_c, y^* + M^j)\| \leq \hat{\kappa}_{ef,j}\Delta^c$.
The latter is equivalent to the continuous gradient at the point $(x^* + s_c, y^* + M^j)$:

$$\|\nabla_x m(x^* + s_c, y^* + M^j) - f(x^* + s_c, y^* + M^j)\| \leq \hat{\kappa}_{eg,j}\Delta^c \leq \bar{\kappa}_{eg}\Delta^c,$$

which concludes the proof. □

3.2.4 Conditions for Mixed-Integer Derivative-Free Methods

Theorems 3.1 and 3.2 show that a mixed-integer approximation can always be computed for any point $(x, y) \in \Omega_m$, a continuous trust-region radius $\bar{\Delta}$ and a matrix $M \in \mathscr{M}(y)$. Let Δ_{max} be a given parameter. In order to define a proper class $\mathcal{M}$ of models that approximate the function f, it is required to prove that if a model m is fully linear in a neighborhood $\hat{B}_c(x, \bar{\Delta}) \times \Omega_d$ with constants κ_{ef}, κ_{eg}, then, it is also fully linear on $\hat{B}_c(x, \Delta) \times \Omega_d$ for any $\Delta \in [\bar{\Delta}, \Delta_{max}]$ with the same constants. This property will be used to prove the convergence of the Algorithm presented in Chapter 4.

Lemma 3.4. *Suppose Assumption 1.3 holds. Assume that the model $m(x + s_c, y + s_z)$ is mixed-integer fully-linear with respect to $f(x, y)$, the trust-region radius $\bar{\Delta}$ and the discrete set $\Omega_d \subset \Omega_z$ with constants κ_{eg}, κ_{ef}. Assume also, without loss of generality, that $\kappa_g \leq \kappa_{eg}$. Then, the model $m(x + s_c, y + s_z)$ is also mixed-integer fully-linear in $\hat{B}_c(x, \Delta) \times \Omega_d$ for $\Delta \in [\bar{\Delta}, \Delta_{max}]$ with the same constants κ_{eg}, κ_{ef} and $\kappa_{eg}/2 \leq \kappa_{ef}$. This condition is called **model accuracy in concentric spheres** [66], [67].*

Proof. First, we prove that the error constant for the gradient satisfies the desired properties. Let $s_c \in \mathbb{R}^{n_c}$ be a vector such $\bar{\Delta} \leq \|s_c\| \leq \Delta$, the scalar $\theta = \bar{\Delta}/\|s_c\| \leq 1$, and $w \in \Omega_d$. Recalling the mixed-integer fully-linearity of m and that $(x + \theta s_c, w) \in \hat{B}_c(x, \Delta) \times \Omega_d$ we have

$$\|\nabla_x f(x + \theta s_c, w) - \nabla_x m(x + \theta s_c, w)\| \leq \kappa_{eg}\bar{\Delta}. \tag{3.10}$$

A bound for $\|\nabla_x f(x + s_c, w) - \nabla_x m(x + s_c, w)\|$ is computed as follows:

$$\begin{aligned} &\|\nabla_x f(x + s_c, w) - \nabla_x m(x + s_c, w)\| \\ &= \|\nabla_x f(x + s_c, w) - \nabla_x f(x + \theta s_c, w) + \nabla_x f(x + \theta s_c, w) \\ &-\nabla_x m(x + s_c, w) - \nabla_x m(x + \theta s_c, w) + \nabla_x m(x + \theta s_c, w)\|. \end{aligned}$$

Applying the triangle inequality

$$\begin{aligned}
&\|\nabla_x f(x+s_c,w)-\nabla_x m(x+s_c,w)\| \\
&\leq \|\nabla_x f(x+s_c,w)-\nabla_x f(x+\theta s_c,w)-\nabla_x m(x+s_c,w)+\nabla_x m(x+\theta s_c,w)\| \\
&\quad +\|\nabla_x f(x+\theta s_c,w)-\nabla_x m(x+\theta s_c,w)\| \\
&\leq \|\nabla_x f(x+s_c,w)-\nabla_x f(x+\theta s_c,w)-\nabla_x m(x+s_c,w) \\
&\quad +\nabla_x m(x+\theta s_c,w)\|+\kappa_{eg}\bar{\Delta}.
\end{aligned} \tag{3.11}$$

A bound on the first term of (3.11) is computed using the Lipschitz continuity of ∇f_x and that $\nabla_{xx} m(s_c,w)=0,\ \ \forall s_c\in\mathbb{R}^{n_c}, w\in\Omega_d$

$$\begin{aligned}
&\|\nabla_x f(x+s_c,w)-\nabla_x f(x+\theta s_c,w)\|+\|\nabla_x m(x+s_c,w)-\nabla_x m(x+\theta s_c,w)\| \\
&\qquad \leq \kappa_g\|s_c(1-\theta)\| \leq \kappa_{eg}(\|s_c\|-\bar{\Delta}).
\end{aligned} \tag{3.12}$$

By combining (3.11) and (3.12) we obtain

$$\|\nabla_x f(x+s_c,w)-\nabla_x m(x+s_c,w)\| \leq \kappa_{eg}\|s_c\| \leq \kappa_{eg}\Delta. \tag{3.13}$$

Now we establish the constants for the error in the model. Let $r:\mathbb{R}\to\mathbb{R}^{n_c},\ \ r(t)=x+ts_c$ and $\zeta(t)=f(r(t),w)-m(r(t),w)$. We have that $|\zeta(r(\theta))|=|f(x+\theta s_c,w)-m(x+\theta s_c,w)|\leq\kappa_{ef}\bar{\Delta}$ and $|g(r(1))|=|f(x+s_c,w)-m(x+s_c,w)|$. Recalling the fundamental theorem of line integrals:

$$|\zeta(r(1))-\zeta(r(\theta))|=\left|\int_\theta^1 \nabla_x\zeta(r(t))\cdot\frac{dr(t)}{dt}dt\right|\leq\int_\theta^1\|\nabla_x\zeta(r(t))\|\cdot\left\|\frac{dr(t)}{dt}\right\|dt.$$

As $\frac{dr(t)}{dt}=s_c$ and using (3.13)

$$\begin{aligned}
&\|\nabla_x\zeta(r(t))\|\leq t\kappa_{eg}\|s_c\|\ \ \forall t\geq\theta \\
&|\zeta(r(1))-\zeta(r(\theta))|\leq\int_\theta^1 t\cdot\kappa_{eg}\|s_c\|^2dt\leq\frac{(1-\theta^2)}{2}\kappa_{eg}\|s_c\|^2\leq\kappa_{ef}(\|s_c\|^2-\bar{\Delta}^2).
\end{aligned}$$

The bound is finally obtained by using the triangle inequality

$$|f(x+s_c,w)-m(x+s_c,w)|\leq|\zeta(r(1))-\zeta(r(\theta))|+|\zeta(r(\theta))|\leq\kappa_{ef}\|s_c\|^2\leq\kappa_{ef}\Delta^2. \tag{3.14}$$

The proof is complete. □

Lemma 3.4 implies that there exists a suitable pair of constants κ_{ef}, κ_{eg} for every $(x, y) \in \Omega_m$, for which the error in every fully-linear model constructed in the domain $\hat{B}_c(x, \Delta) \times \Omega_d$ (with $\Delta \in (0, \Delta_{max}]$) is bounded by $\kappa_{ef}\Delta^2, \kappa_{eg}\Delta$. Theorems 3.1 and 3.2 show that the error in the approximation of a mixed-integer function in a suitably-defined neighborhood depends on the error constants of the linear approximation of the function with respect to the continuous variables; in the LQMI case, it also depends on the $\|M^{-1}\|$. Conn et al. [51] show that the error constants in the linear interpolation case are a function of the Lipschitz constant and the geometry of the points inside a trust-region. Thus, if $\|M^{-1}\|$ and κ_g are bounded we guarantee that we are able to construct a mixed-integer fully-linear approximation of function f at any point in the domain, with error bounded by a global set of constants κ_{ef}, κ_{eg}:

Proposition 3.4. *For any given function f that satisfies Assumption 1.3 and the class of discrete sets $\bar{\Omega} = \{\Omega_d^G(y, M) \mid y \in \Omega_z, M \in \mathscr{M}(y)\}$, we guarantee that there exists a fully-linear class of models $\mathcal{M}$ (Definition 3.2) with suitable positive global constants κ_{ef}, κ_{eg} such that, for any given $\Delta \in (0, \Delta_{max}]$, $(x, y) \in \Omega_m$ and $\Omega_d \in \bar{\Omega}$, the error in the function and gradient approximation is bounded. Moreover, we can obtain a fully-linear model from this class in a finite, uniformly bounded number of operations and function evaluations.*

Proposition 3.5. *For any given LQMI function f, (i.e, f satisfies Assumptions 1.1, 1.3, 3.1 and 3.2) and the class of discrete sets $\bar{\Omega} = \{\Omega_d^{LQMI}(y) \mid \forall y \in \Omega_z\}$, we guarantee that there exists a fully-linear class of models $\mathcal{M}$ (Definition 3.2) with suitable positive global constants κ_{ef}, κ_{eg} such that, for any given $\Delta \in (0, \Delta_{max}]$, $(x, y) \in \Omega_m$, and $\Omega_d \in \bar{\Omega}$, the error in the function and gradient approximation is bounded. Moreover, we can obtain a fully-linear model from this class in a finite, uniformly bounded number of operations and function evaluations.*

3.3 Conclusions and Future Work

In this chapter we have introduced the concept of Locally Quadratic Mixed-Integer function (LQMI), which allows us to overcome the lack of knowledge on the contribution of the discrete variables on the objective function and the interaction between

integer and continuous variables. Moreover, we extended the concept of *fully-linear* surrogate models into the mixed-integer domain and introduced a general framework for the computation of accurate models for the LQMI function. In addition, we introduced methods for the fast computation and update of LQMI surrogates. Finally, we proved that the framework for computing LQMI models can be used for the approximation of a general mixed-integer function, the output is a quadratic model that is accurate in a reduced mixed-integer domain. These results are used in Chapter 4 to devise a trust-region algorithm for the solution of Problem 1.1.

We highlight that the model construction framework introduced in this chapter resembles the *manifold sampling* method [84]–[86] that computes gradient approximation by sampling in different parts of the domain.

In the future, we hope to extend the notions of mixed-integer *fully-linear* models into other types of surrogate approximation. We are particularly interested in using the RBFs, a class of functions that globally approximates the objective function and are less restrictive than the polynomial models for the management of the geometry of the interpolating set. We believe that the method introduced in this chapter can extend the results of the work of Wild and Shoemaker [64].

Chapter 4

LQMI-Based Trust-Region Algorithm

Theorems 3.1 and 3.2 show that the model $m(x^* + s_c, y^* + s_z)$ obtained after imposing $l_c = g^0$, $A_M = (M^\top)^{-1} A_g$ and fully determining the integer coefficients l_z, A_z, is mixed-integer fully linear. In this chapter, we present a trust-region algorithm based on this principle. The output of this algorithm is a point $(\hat{x}, \hat{y})$ that is stationary with respect to a discrete set $\Omega_d^{LQMI}(\hat{y})$ if f is an LQMI function, and stationary with respect to a discrete set $\Omega_d^Q(\hat{y})$ for which $\Omega_d^G(\hat{y}, M) \subseteq \Omega_d^Q(\hat{y}) \subset \Omega_d^{LQMI}(\hat{y})$ with $|\Omega_d^Q(\hat{y})| \leq \frac{(n_z+1)(n_z+2)}{2}$, otherwise.

We introduce Θ_k and $\bar{\Theta}_k$, mixed-integer stationarity parameters that allow us to prevent early convergence into a first-order stationary point that only considers the continuous variables:

Definition 4.1. *Let ϵ_a and ϵ_c be positive user defined parameters in Algorithm 4.1. For given $x_k, y_k \in \Omega_m$ let the partial mixed integer stationarity parameter Θ_k and the adjusted mixed-integer stationarity parameter $\bar{\Theta}_k$ be:*

$$\Theta_k = \max_{y \in Q(y_k, M_k)} \min_{x \in \hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c} m_k(x_k, y_k) - m_k(x, y)$$

and

$$\bar{\Theta}_k = \begin{cases} \Theta_k & \text{if } \Theta_k \geq \max\{\epsilon_a, \epsilon_c\} \\ 0 & \text{otherwise.} \end{cases}$$

Definition 4.2. *Let the combined stationary parameter Φ_k be defined as*

$$\Phi_k = \max\{\|\boldsymbol{red}(l_k^c, x_k)\|, \bar{\Theta}_k\}.$$

4.1 Overview of the Proposed Algorithm

The LQMI-based trust-region algorithm is presented in Algorithm 4.1. For a given iteration k, the center of the trust-region is denoted by (x_k, y_k). Y^k indicates the set of points sampled up to iteration k and $X_w^k = \{x \in \mathbb{R}^{n_c} \mid (x, w) \in Y^k\}$ is the set of points with same integer component $w \in \mathbb{Z}^{n_z}$. The basic idea is to compute at each iteration a model m_k by sampling on the continuous manifolds of the integer points $q_k^j \in Q(y_k, M^k)$ and on the continuous manifold of y_k. The set of integer generating vectors M^k can potentially change at each iteration and it is selected according to rules that allows one to reuse sampled points from previous iterations. We identify by $\bar{X}_o^k \subseteq X_{y_k}^k$, $\bar{X}_j^k \subseteq X_{q_k^j}^k$ and $\bar{Z}^k$ the points used to compute the model approximations at iteration k.

The inputs of the algorithm are: the black-box function $f(x, y)$, the variable lower bounds (x_{lb}, y_{lb}) and upper bounds (x_{ub}, y_{ub}), the initial center of the trust-region $(x_0, y_0) \in \Omega_m$, the starting and maximum size of the trust-region $\Delta_0^{icbc}, \Delta_0^{icbz}, \Delta_{max}^c$ and Δ_{max}^z, the parameters used to update the region γ_0 and γ_1, the parameters used to evaluate the quality of an iteration η_0 and η_1, the scaling parameters of the critical step μ and β, and the reduction factor of the trust-region ω used in the criticality step. Tolerance values ϵ_c and ϵ_a are used to activate the *Criticality Step* and to accept new iterates in a rescue procedure, respectively. Finally, we consider the parameters used for model construction and maintenance: *UP* determines if a inexpensive update of the model is done after a *successful* iteration, *IC* determines if the quadratic interpolation $\tau(y)$ is fully-determined or not, *SH* identifies if a search heuristic is activated before or after the solution of the surrogate approximation, and c_{fr} represents the fraction of samples required to compute the approximation of terms $l_{c,k}$ and $A_{M,k}$.

Algorithm 4.1 DFO MINLP algorithm

Input: Function $f(x,y)$, variable bounds x_{lb}, x_{ub} and y_{lb}, y_{ub}, initial point $(x_o, y_o) \in \Omega_m$. Initial trust-region radius Δ_0^{icbc}, $\Delta_0^{icbz}, \Delta_{max}^c$ and Δ_{max}^z; γ_0, γ_1 such that $0 < \gamma_0 < 1 < \gamma_1$, η_0, η_1 such that $0 < \eta_0 < \eta_1 < 1$; $0 < \beta < \mu$; $\epsilon_a, \epsilon_c > 0$; and $\omega \in (0,1)$
$UP \in \{0,1\}$, $c_{fr} \in (0,1]$, $IC \in \{0,1\}$ and $SH \in \{0,1\}$

Output: Local minimum $(\hat{x}, \hat{y})$ of $f(x,y)$

Set $k := 0$, set $\bar{\epsilon} = \max\{\epsilon_c, \epsilon_a\}$
Set $M_0 =$ **IntegerTransformation**$(y_0, y_{lb}, y_{ub}, \emptyset, \Delta_0^{icbz})$
Set $Q_0 = Q(y_0, M_0)$
Set $Y^0 = (x_0, y_0) \cup \bigcup_{q^j \in Q^0}(x_0, q^j)$
Set $(m_o^{icb}, \bar{X}_o^k, \bar{X}_j^k, \bar{Z}^k) =$
MixedIntegerModelComputation$(x_0, x_{lb}, x_{ub}, y_0, y_{lb}, y_{ub}, \Delta_0^{icbc}, \Delta_0^{icbz}, Q_0, Y^0, c_f, IC)$
repeat
  if $\Phi_k^{icb} < \epsilon_c$ and (m_k^{icb} is not fully-linear or $\Delta_k^{icb} > \mu \|$**red**$(l_{c,k}^{icb}, x_k)\|$) **then**
    Set $(\tilde{m}_k, \tilde{\Delta}_k, Y^k, M_k) =$
    CriticalityTest$(x_k, y_k, x_{lb}, x_{ub}, y_{lb}, y_{ub}, \Delta_k^{icbc}, \Delta_k^{icbz}, Y^k, M_k, \omega, \mu, \epsilon_c, c_{fr}, IC)$
    Set $m_k = \tilde{m}_k$
    Set $\Delta_k^c = \min\{\max\{\tilde{\Delta}_k, \beta \|$**red**$(\tilde{l}_{c,k}, x_k)\|\}, \Delta_k^{icbc}\}$
    Set $\Delta_k^z = \max\left\{\frac{\Delta_k^c}{\Delta_k^{icbc}}\Delta_k^{icbz}, 1\right\}$
  else
    Set $m_k = m_k^{icb}$, $\Delta_k^c = \Delta_k^{icbc}, \Delta_k^z = \Delta_k^{icbz}$
  end if
  Set $(x', y', \rho_k) =$ **CandidateComputation**$(x_k, y_k, Y^k, \epsilon_a, SH)$
  if $\rho_k \geq \eta_0$ **then**
    if $\rho_k \geq \eta_1$ **then**
      Set $\Delta_{k+1}^{icbc} = \min\{\gamma_1 \Delta_k^c, \Delta_{max}^c\}$, $\Delta_{k+1}^{icbz} = \min\{\gamma_1 \Delta_k^z, \Delta_{max}^z\}$
    end if
    Set $(m_{k+1}^{icb}, \bar{X}_o^{k+1}, \bar{X}_j^{k+1}, \bar{Z}^{k+1}, M^{k+1}) =$
    ModelUpdate$(m_k, x', x_{lb}, x_{ub}, y', y_{lb}, y_{ub}, \Delta_{k+1}^{icbc}, \Delta_{k+1}^{icbz}, Y^k, UP, c_{fr}, IC)$
  else
    Set $x' = x_k, y' = y_k$
    if Model m_k is not mixed-integer fully linear **then**
      Set $(m_{k+1}^{icb}, \bar{X}_o^k, \bar{X}_j^k, \bar{Z}^k) =$ **GeometryImprovement**$(\bar{X}_o^k, \bar{X}_j^k, \bar{Z}^k)$
    else
      $(\bar{x}, \bar{y}) =$ **RescueProcedure**$(m_k, Q_k Y^k, \epsilon_a)$
      if $f(\bar{x}, \bar{y}) < f(x_k, y_k)$ **then**
        Set $(x', y') = (\bar{x}, \bar{y})$
        Set $(m_{k+1}^{icb}, \bar{X}_o^{k+1}, \bar{X}_j^{k+1}, \bar{Z}^{k+1}, M^{k+1}) =$
        ModelUpdate$(m_k, x', x_{lb}, x_{ub}, y', y_{lb}, y_{ub}, \Delta_k^c, \Delta_k^z, Y^k, UP, c_{fr}, IC)$
      else
        Set m_{k+1}^{icb} , $\Delta_{k+1}^{icbc} = \gamma_0 \Delta_k^c$, $\Delta_{k+1}^{icbz} = \max\{\gamma_0 \Delta_k^z, 1\}$
        Set $M_{k+1} =$ **IntegerTransformation**$(y_k, y_{lb}, y_{ub}, \Delta_{k+1}^{icbz})$
      end if
    end if
  end if
  Set $(x_{k+1}, y_{k+1}) = (x', y')$
  Set $k = k + 1$
until Convergence is proven

The initialization (Lines $1-5$) corresponds to setting the first set of integer generating vectors to the identity matrix, computing the associated set of integer generating points (see Definitions 3.5 and Definition 3.3) and the initial set of samples Y^0 is initialized by adding the initial center of the trust-region and the points obtained after moving the center of the trust-region along the directions given by M_0. Finally, a first incumbent model m_0^{icb} is computed (Line 5). See Equation 3.4 for a definition of the model m. The main part of the algorithm consists of a loop (Lines $6-38$) that is repeated until a stopping criterion is reached. The loop starts by checking if the combined stationarity parameter Φ_k^{icb} of the incumbent model m_k^{icb} is smaller than a given threshold ϵ_c (Line 7), implying algorithmic convergence into a first-order stationary point. In this case, the *Criticality Step* (Lines $8-11$) is invoked. The **CriticalityTest** evaluates at the same time if the incumbent model is fully-linear (under the assumption that the function is LQMI) and if there exists certain relationship between the reduced gradient $\mathbf{red}(l_{c,k}^{icb}, x_k)$ and the continuous trust-region radius. If m_k^{icb} does not accomplish both conditions, the **CriticalityTest** (Algorithm 4.2) is used to generate a new model for which $\Phi_k \geq \bar{\Theta}_k \geq \epsilon_c$, or, a model where both conditions hold (Line 8). However, if the model m_k^{icb} is fully-linear and the desired relationship is satisfied,the model m_k^{icb} is accepted (Line 13) and subsequently used to compute a new candidate solution.

After the *Criticality Step*, a new candidate solution (x', y') is computed together with the update parameter ρ_k (Line 15). If $\rho_k \in [\eta_o, \eta_1)$ iteration k is said to be a *successful iteration*, when the new solution yields a sufficiently large improvement with respect to the previous solution. Furthermore, if $\rho_k > \eta_1$ iteration is said to be *very-successful*. If we have a *successful* or *very-successful* iteration (i.e., $\rho_k \geq \eta_0$), we use (x', y') as new center of the trust-region, and we generate the model to be used in the next iteration. This procedure takes into account the parameter UP to use quick linear update of the current model or to generate a new one centered on (x', y'). Moreover, if we have a *very-successful* iteration (i.e., $\rho_k \geq \eta_1$), we increase the size of the trust-region (Line 18).

In case $\rho_k \leq \eta_0$ we check if the model used is fully-linear (Line 23). Under the assumption that the function is LQMI, the mixed-integer fully-linearity is achieved when all the continuous-related elements of the model (l_c, A_M) are computed using

fully-linear approximations on the continuous manifolds at y_k and $Q(y_k, M_k)$, and the quadratic elements of the model l_z, A_z determine all the degrees of freedom for Problem 3.6a-3.6b. In case model m_k is mixed-integer fully-linear the rescue procedure is invoked (Line 26). The goal of the rescue procedure is to search for candidate points to be used as center of the trust-region, exploiting the fully-linearity of m_k with respect to the integer neighborhood $\Omega_d^G(y_k, M_k)$ as described in Theorem 3.2. The rescue procedure is required for the algorithmic convergence in functions that are not LQMI and in general it can be complemented with any heuristic without affecting said convergence. If the rescue procedure succeeds in improving the objective function (Line 27), the iteration is said *acceptable* and the model is updated accordingly (Line 29). If this is not the case, the iteration k is called *unsuccessful* and the size of the trust-region is reduced (Line 31). If the fully-linearity condition is not satisfied, we have a *model-improving* iteration, where a geometry procedure is called to add (if necessary) a new sample to each of the sets of interpolation points $\bar{X}_0^k$, $\bar{X}_j^k$ and $\bar{Z}^k$ to improve the quality of the model (Line 24).

In the subsequent subsections we describe all the procedures used in Algorithm 4.1.

Remark 1. *Section 3.2 details that a fully-linear surrogate is obtained by considering a matrix M with columns in $\hat{B}_z(0,1)$ (see Definition 3.3). However, Algorithm 4.1 considers a set of matrices dependent on Δ_k^z, with the elements of M_k in $\hat{B}_z(0, \Delta_k^z)$. This does not affect algorithmic convergence with respect to the sets $\Omega_d^Q(y_k)$ and $\Omega_d^{LQMI}(y_k)$, since in Section 4.2 we prove that Algorithm 4.1 generates an infinite number of iterations such that $\Delta_k^z = 1$, as long as the matrix M_k is not singular (or $\|M_k^{-1}\| > 0$).*

4.1.1 CriticalityTest (Algorithm 4.2)

This procedure is used to evaluate the convergence into a first-order stationary point. The **CriticalityTest** aims to generate an approximation $\tilde{m}_k$ of $f(x_k, y_k)$ for which at least one of the followings conditions holds:

- $\tilde{\Theta}_k \geq \bar{\epsilon} \geq \epsilon_c$
- $\mu \|\mathbf{red}(\tilde{l}, x_k)\| \geq \tilde{\Delta}_k$ and $\tilde{m}_k$ is fully-linear with respect to $\tilde{\Delta}_k$.

To guarantee these conditions, at first we use the **ManifoldSearch** (Line 1) procedure that attempts to retrieve a model $\tilde{m}_k$ and an matrix M_k such that $\max\{\epsilon_c, \epsilon_a\} \leq$

$\tilde{\Theta}_k \leq \Phi_k$. The **ManifoldSearch** iteratively selects one point y on a randomly defined finite set $\Omega_k^{MS} \subset \hat{B}_z(y_k, \Delta_k^z)$ and constructs a continuous fully-linear approximation of $\hat{f}_y(x)$ with respect to $\tilde{\Delta}_k$. If the model improvement (with respect to $f(x_k, y_k)$) exceeds $\max\{\epsilon_c, \epsilon_a\}$, the vector $y - y_k$ is included in the matrix M_k, the mixed-integer model is updated and the **ManifoldSearch** procedure is stopped. Otherwise the procedure is repeated until the last element of Ω_k^{MS}. We remark that the **ManifoldSearch** does no affect convergence to a fist-order stationary point nor necessary for proving algorithmic convergence; however, it is useful in preventing the early convergence of Algorithm 4.1 to a suboptimal solution.

If the **ManifoldSearch** fails in achieving the desired $\tilde{\Theta}_k$, we invoke an iterative procedure (Lines 8 - 13), whose outcome is a fully-linear model with $\mu\|\mathbf{red}(\tilde{l}, x_k)\| \geq \tilde{\Delta}_k$. If $\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| > 0$ this loop must be finished in a finite number of iterations [66]. We highlight that the outcome of this algorithm is a model m_k such that $\Phi_k \geq \epsilon_c$ or $\Phi_k \geq \|\mathbf{red}(l_k^c, x_k)\|\mu^{-1} \geq \Delta_k^c \mu^{-1}$ and fully-linear.

Algorithm 4.2 CriticalityTest

Input: Point (x_k, y_k), variable bounds (x_{lb}, x_{ub}) and (y_{lb}, y_{ub}), sampling set Y^k, transformation matrix M_k, continuous radius Δ_k^{icbc}, integer radius Δ_k^{icbz},the reduction factor of the trust-region ω, geometric parameter μ, criticality tolerance $\bar{\epsilon}$ and modelling parameters c_{fr}, IC.

Output: $\tilde{m}_k, \tilde{\Delta}_k, Y^k, M_k$

1: Set $\tilde{m}_k, \tilde{\Theta}_k, M_k =$ **ManifoldSearch**$(x_k, y_k, x_{lb}, x_{ub}, y_{lb}, y_{ub}, \Delta_k^{icbc}, \Delta_k^{icbz}, Y^k, M_k, \bar{\epsilon})$
2: **if** $\tilde{\Theta}_k \geq \bar{\epsilon}$ **then**
3: Set $\tilde{\Delta}_k = \Delta_k^{icb}$
4: **return** $\tilde{m}_k, \tilde{\Delta}_k, Y^k, M_k$
5: **else**
6: Set $i = 1$
7: **repeat**
8: Set $\tilde{\Delta}_k = \omega^{i-1}\Delta_k^{icb}$
9: Set $\tilde{\Delta}_k^z = \max\{1, \omega^{i-1}\Delta_k^{icbz}\}$
10: Set $\tilde{M}_k =$ **IntegerTransformation**$(y_k, y_{lb}, y_{ub}, \Delta^z)$, $\tilde{Q}_k = Q(y_k, \tilde{M}_k)$
11: Set $(\tilde{m}_k, \bar{X}_o^k, \bar{X}_j^k, \bar{Z}^k) =$ **MixedIntegerModelComputation**$(x_k, x_{lb}, x_{ub}, y_k, y_{lb}, y_{ub}, \tilde{\Delta}_k, \tilde{\Delta}_k^z, \tilde{Q}_k, Y^k, 1, 1)$
12: i = i +1
13: **until** $\tilde{\Delta}_k \leq \mu\|\mathbf{red}(\tilde{l}_{c,k}, x_k)\|$
14: **end if**

4.1.2 CandidateComputation (Algorithm 4.3)

In this procedure the information from model m_k and the set Y^k is used to generate a point (x', y') that potentially yields an objective reduction, and to provide an estimate of the fitness of the surrogate approximation by computing the update parameter ρ_k. The parameter SH defines if the surrogate optimization is done before or after evaluating the best solution in previous samples. Note that if a better solution is found among the points used for model construction (Lines 3 and 9), the update parameter ρ_k is set to η_0 to not modify the trust-region radii Δ^c and Δ^z.

Algorithm 4.3 CandidateComputation

Input: Point (x_k, y_k), set of samples Y^k, acceptance tolerance ϵ_a and search heuristic parameter SH
Output: New candidate solutions (x', y') and update parameter ρ_k

1: Set $(\bar{x}, \bar{y}) = \text{argmin}_{(x,y)\in Y^k}\{f(x,y)\}$
2: **if** $SH = 1$ **then**
3: **if** $f(x_k, y_k) - f(\bar{x}, \bar{y}) \geq \epsilon_a$ **then**
4: $\rho_k = \eta_0$
5: Go to line 14
6: **end if**
7: **end if**
8: Set $(x, y) = \text{argmin}_{(x,y)\in \hat{B}_z(x_k, y_k, \Delta_k^c, \Delta_k^z)\cap\Omega_m} m_k(x, y)$ and $s_c = x - x_k$, $s_z = y - y_k$
9: **if** $\|s_z\|_1 \geq 1$ and $m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z) \geq \epsilon_a$ **then**
10: Set $(x', y') = (x_k, y_k)$, $\rho_k = 0$
11: **else**
12: $\rho_k = \frac{f(x_k,y_k) - f(x_k+s_c, y_k+s_z)}{m_k(x_k,y_k) - m_k(x_k+s_c, y_k+s_z)}$
13: **end if**
14: Set $(\tilde{x}, \tilde{y}) =$
$\text{argmin}_{(x,y)\in Y^k}\{f(x,y) \mid f(x_k, y_k) - f(x,y) \geq (\min\{\|y - y_k\|_1, 1\})\epsilon_a\}$
15: **if** $\rho_k \geq \eta_0$ **then**
16: **if** $f(\tilde{x}, \tilde{y}) < f(x_k + s_c, y_k + s_z)$ **then**
17: Set $(x', y') = (\tilde{x}, \tilde{y})$, $\rho_k = \eta_0$
18: **else**
19: Set $(x', y') = (x_k + s_c, y_k + s_z)$
20: **end if**
21: **end if**

4.1.3 ModelUpdate(Algorithm 4.4)

This procedure is activated when a *successful* iteration occurs $\rho_k \geq \eta_0$. In it, a new basis of generating vectors M_{k+1} is computed. Such discrete directions are chosen from the set $\hat{B}_z(y_k, \hat{\Delta}^z) \cap \Omega_z$ using a modified pivoting algorithm that aims for the reduction of $\|M_{k+1}^{-1}\|$, a term related to the error constants of an LQMI function (see

Theorem 3.1). Finally, a new incumbent model m_{k+1}^{icb} is computed for the next iteration by updating the current one, or by computing a new surrogate using Algorithm 4.6 centered in the point (x_{k+1}, y_{k+1}). Model computation is explained in the next subsection.

Algorithm 4.4 ModelUpdate

Input: Model m_k, new trust-region center (x_{k+1}, y_{k+1}), variable bounds (x_{lb}, x_{ub}) and (y_{lb}, y_{ub}), trust-region radii Δ^c, Δ^z set of samples $\mathcal{Y}^k$, update decision UP, and, modelling parameters c_{fr}, IC

1: Set $M_{k+1} =$ **IntegerTransformation**$(y_k, y_{lb}, y_{ub}, \Delta^z)$
2: **if** UP **then**
3: Use equation 3.7 to update linear terms in m_k to generate m_{k+1}^{icb}
4: **else**
5: $(m_{k+1}^{icb}, \bar{X}_o^k, \bar{X}_o^k, \bar{Z}^k) =$ **MixedIntegerModelComputation**$(x_{k+1}, y_{k+1}, \Delta^c, \Delta^z, Q_{k+1}, \mathcal{Y}^k, c_{fr}, IC)$
6: **end if**

4.1.4 RescueProcedure (Algorithm 4.5)

In this procedure we use the properties of mixed-integer fully-linear models in a final attempt to obtain a decrease in the objective. This procedure takes as input the model m_k, the set of discrete directions Q_k, the set of previously sampled points $\mathcal{Y}^k$ and an acceptance tolerance ϵ_a.

The new candidate is estimated by optimizing the surrogate model, restricting the integer search in the directions M_k, as well as in the current manifold ($s_z = 0$):

$$\min_{(x,y)} m_k(x, y) \tag{4.1}$$

$$x \in \hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c \tag{4.2}$$

$$\begin{bmatrix} V \\ y = y_k \end{bmatrix} \vee \begin{bmatrix} \neg V \\ y \in Q(y_k, M_k) \\ m_k(x_k, y_k) - m_k(x, y) \geq \epsilon_a \end{bmatrix} \tag{4.3}$$

$$V \in \{True, False\} \tag{4.4}$$

Problem 4.1-4.4 can by solved by enumeration or by formulating it as a disjunctive optimization problem.

Algorithm 4.5 RescueProcedure

Input: Model m_k, set of discrete directions Q_k, set of previously sampled points Y^k, acceptance tolerance ϵ_a
Output: Point (x', y')
1: Set $(\tilde{x}, \tilde{y}) = \text{argmin}_{(x,y)\in Y^k}\{f(x,y) \mid f(x_k, y_k) - f(x,y) \geq \epsilon_a\}$
2: Solve problem 4.1 to compute $(\bar{x}, \bar{y})$
3: **if** $f(\tilde{x}, \tilde{y}) < f(\bar{x}, \bar{y})$ **then**
4: Set $(x', y') = (\tilde{x}, \tilde{y})$
5: **else**
6: Set $(x', y') = (\bar{x}, \bar{y})$
7: **end if**

4.1.5 MixedIntegerModelComputation (Algorithm 4.6)

A new surrogate is created using the previous samples Y^k and considering the parameters c_f, IC. If $c_f = 1$ and $IC = 1$ the model immediately becomes mixed-integer fully linear. **LinearInterpolationSet** is a procedure that selects $n_c + 1$ samples to compute a linear interpolation, using a geometry improvement that improves the poisedness of the interpolation set. We remark that if $c_{fr} < 1$, the approximation is underdetermined and some samples are kept for future use in the **GeometryImprovement** procedure, unless all the points selected by the algorithm have already been explored in previous iterations. **QuadraticInterpolationSet** is a procedure designed to select the best sampling set to interpolate a second order approximation of the function $f(x_k, y)$ in the set $\hat{B}_z(y_k, \Delta_k^z) \cap \Omega_z$ based on an existing set of samples QN (Line 8). Now we briefly explain methodologies to compute linear and quadraticaly independent sets of interpolating samples. We point out that such techniques have been developed for real variables, however, they can be easily extended to the integer case.

4.1.6 Geometry Improvement - LinearInterpolationSet

This procedure can be performed in two ways: using Pivoting algorithms, or using the Λ-parameter [87]. To exemplify their use we now detail the pivoting algorithms. In the continuous case ($n_z = 0$) the quality of a surrogate model, computed from interpolation or regression, depends directly on the position of the samples inside the trust-region. In the linear interpolation case, the two error constants κ_{ef}, κ_{eg} can

Algorithm 4.6 MixedIntegerModelComputation

Input: Point (x_k, y_k), variable bounds (x_{lb}, x_{ub}) and (y_{lb}, y_{ub}), Trust-Region Radii Δ^c and Δ^z, set of generating points Q_k and set of sampled points Y^k
Modeling parameters c_f and IC

Output: Mixed-integer model m and the points used to compute it $\bar{X}_o^k$, $\bar{X}_j^k$ and $\bar{Z}^k$

1: $\bar{X}_o^k = \textbf{LinearInterpolationSet}(X_{y_k}, x_k, x_{lb}, x_{ub}, \Delta^c, c_f)$
2: $Y_k = Y_k \cup \{(x, y_k) \mid x \in \bar{X}_o^k\}$
3: Compute l_c from $\bar{X}_o$
4: **for all** $q^j \in Q_k$ **do**
5: $\quad \bar{X}_j^k = \textbf{LinearInterpolationSet}(X_{q^j}, x_k, x_{lb}, x_{ub}, \Delta^c, c_f)$
6: $\quad Y_k = Y_k \cup \{(x, q^j) \mid x \in \bar{X}_j^k\}$
7: **end for**
8: Compute A^M from $\bar{X}_j^k$ and l_c
9: $QN = \{y \in \Omega_z \mid (x_k, y) \in Y^k, \ \ y \in \hat{B}_z(y_k, \Delta^z)\}$
10: $\bar{Z}^k = \textbf{QuadraticInterpolationSet}(QN, y_k, y_{lb}, y_{ub}, \Delta^z, IC)$
11: Compute the integer interpolation terms l_z, A_z from $\hat{Z}^k$ by solving the quadratic interpolation problem 3.6a

be defined as follows [51]:

$$\kappa_{eg} = v(1 + n_1^{1/2}\|\tilde{X}^{-1}\|/2)$$
$$\kappa_{ef} = \kappa_{eg} + v/2,$$

where v is the Lipschitz constant of $f(x)$ and $\tilde{X}$, $\tilde{x}_{i,j} \in [0, 1]$ is the matrix of scaled displacements from the trust-region center x^*:

$$\tilde{X} = \frac{1}{\Delta^c}\begin{bmatrix} x_{1,1} - x_1^*, & \dots, & x_{n_1,1} - x_1^* \\ \vdots, & \ddots, & \vdots \\ x_{1,n_1} - x_{n_1}^*, & \dots, & x_{n_1,n_1} - x_{n_1}^* \end{bmatrix}.$$

Pivoting algorithms based on the LU and QR factorization consist in the use of Gaussian elimination to compute the simplex geometry that yields the smallest possible bound of $\|\tilde{X}^{-1}\|$ (which is related to the concept of a positive uniform basis [88]), thus reducing the error constants in the process. A **LinearInterpolationSet** procedure based on the pivoting algorithm takes as input the set of points which have been previously sampled inside the trust-region and it is able to complete the sampling set if needed. It requires a pivoting tolerance $\xi \in (0, 1/4]$ that sets an upper bound for $\|\tilde{X}^{-1}\|$: $\|\tilde{X}^{-1}\| \leq n_1^{1/2}\epsilon_{growth}/\xi$, where $\epsilon_{growth} > 0$ is an estimated growth

factor that occurs during factorization.

4.2 Convergence of Algorithm 4.1 to a First-Order Critical Point

In this section, we prove that Algorithm 4.1 is globally convergent to a first-order critical point. The proof consists of the following steps. First, we introduce the concepts of *generalized Cauchy step*, *generalized Cauchy point* and *criticality measure*, which are crucial for the convergence of trust-region methods. Then, we show that unless a point is stationary, model update procedures compute a model for which the gradient $\mathbf{red}(l_{c,k}, x_k)$ and the trust-region radius Δ_k^c diverge from zero, and that objective improvement is always possible. Finally, we prove by contradiction that the sequence $\{x_k, y_k\}$ is convergent, and its limiting value $(\tilde{x}, \tilde{y})$ is first-order stationary with respect to the continuous variables, as well as an ϵ_a minimizer with respect to the integer set $\Omega_d^Q(\tilde{y})$. This is the same approach typically used in convergence proofs for trust-region methods in the continuous case [67] [66], but here we show that similar arguments also hold in the mixed-integer case, using the approximation models discussed in previous sections. We remark that from Definition 3.2, and Propositions 3.4 and 3.5 we consider the same error constants κ_{ef}, κ_{eg} in every fully-linear approximation.

4.2.1 Stationarity Conditions on Continuous Variables

In this subsection, we introduce several building blocks for the main convergence proof. Let us define the function χ_k as a measure of stationarity of $f(x, y)$ with respect to the continuous variables.

Definition 4.3. *For $s \in \mathbb{R}^{n_c}$ and $r \in \mathbb{R}$, the function χ_k is defined as:*

$$\begin{aligned} \chi_k(x_k, s, r) = |\min_{d \in \mathbb{R}^{n_c}} & \; d^\top s| \\ s.t. \quad & x_k + d \in \Omega_c \\ & \|d\| \leq r \, . \end{aligned}$$

A point $(x_k, y_k) \in \Omega_m$ is said to be stationary with respect to $f(x,y)$ and Ω_c if $\chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$. χ_k is a direct substitute of $\|\nabla_x f(x,y)\|$ for the constrained optimization case.

Equipped with this definition, we can now discuss the type of local optimality that we aim for.

Definition 4.4. *The point (x^*, y^*) at iteration k is **mixed-integer first-order critical** with respect to the mixed-integer set $\mathcal{N}_D$ and the optimality tolerance ϵ_{opt} if:*

$$f(x^*, y^*) \leq f(x,y) + \epsilon_{opt} \quad \forall (x,y) \in \mathcal{N}_D(x^*, y^*)$$
$$\chi_k(x^*, \nabla_x f(x^*, y^*), 1) = 0.$$

We now define the projection of a vector $P_{\Omega_c}(x)$, and the projected path with respect to the continuous box Ω_c and the continuous center x_k at iteration k. Then, we extend the concept of generalized *Cauchy step* and *Cauchy point* to the mixed-integer case.

Definition 4.5. *The **projection** $P_{\Omega_c} : \mathbb{R}^{n_c} \to \mathbb{R}^{n_c}$ is defined as:*

$$[P_{\Omega_c}(x)]_i = \begin{cases} x_{ub,i} & \textbf{\textit{if}}\ x_i \geq x_{ub,i} \\ x_{lb,i} & \textbf{\textit{if}}\ x_i \leq x_{lb,i} \\ x_i & \textbf{\textit{else}}. \end{cases}$$

Definition 4.6. *Let $x_k \in \Omega_c$. The* projected path *with respect to vector $s \in \mathbb{R}^{n_c}$ is defined as $p_k(x_k, s, t) = P_{\Omega_c}(x_k - t \cdot s)$.*

Definition 4.7. *Let $(x_k, y_k) \in \Omega_m$. The Generalized Mixed-Integer Cauchy step t_k^C of m_k with respect to Ω_c and Δ_k^c is defined as follows:*

$$t_k^C = \operatorname*{argmin}_{t \geq 0} m_k(p_k(x_k, l_k^c, t), y_k)$$
$$s.t. \quad p_k(x_k, l_k^c, t) \in \hat{B}_c(x_k, \Delta_k^c),$$

which for the setting of Algorithm 4.1 is equivalent to:

$$t_k^C = \operatorname*{argmin}_{t \geq 0,\ p_k(l_k^c, t) \in \hat{B}_c(x_k, \Delta_k^c)} (l_k^c)^\top (p_k(l_k^c, t) - x_k).$$

The Generalized Cauchy Point is given by $x_k^C = p_k(x_k, l_k^c, t_k^C)$.

We highlight that for the unbounded optimization case the *Cauchy step* can be viewed as a line-search optimization in the direction $-l_k^c$ on the sphere $\hat{B}_c(x_k, \Delta_k^c)$, and it determines stationarity with respect to the continuous variables. However, the presence of box constraints requires the use of an additional criterion mimicking the first-order Karush–Kuhn–Tucker [89] conditions of the constrained problem. Such criterion is incorporated in the function $\chi_k(x_k, s, r)$. Let $\bar{\chi}_k := \chi_k(x_k, l_k^c, 1)$. The *criticality measure* $\bar{\chi}_k$ and the function χ_k have some properties that allow us to establish convergence to a first-order stationary point over the set Ω_c:

Lemma 4.1. (Conn et al. [90], Lemma 2.2) *Suppose that* $x_k \in \Omega_c$ *and* Ω_c *is nonempty, closed and convex; then:*

1. *The function* $\chi_k(x_k, s, r)$ *is continuous and non-decreasing as a function of* r *for all* $r > 0$.

2. *The function* $\frac{\chi_k(x_k, s, r)}{r}$ *is non-increasing as a function of* r *for all* $r > 0$.

Theorem 4.1. (Conn et al. [67], Theorem 12.1.4) *Suppose that* $x_k \in \Omega_c$, $s \in \mathbb{R}^{n_c}$ *and* $t > 0$. *Then, a solution* $\tilde{d}$ *of the problem*

$$\min_{d \in \mathbb{R}^{n_c}} \{d^\top s \mid x_k + d \in \Omega_c, \ \ \|d\| \leq \|p_k(x_k, s, t) - x_k\|\}$$

is given by $\tilde{d} = p_k(x_k, s, t) - x_k$.

Theorem 4.2. (Conn et al. [67], Theorem 12.1.6) *Suppose that Assumption 1.2 holds, then the quantity* $\chi_k(x_k, \nabla_x f(x_k, y_k), 1)$ *is a proper first-order criticality measure for the optimization problem 1.1. In other words,* $\chi_k(x_k, \nabla_x f(x_k, y_k, 1))$ *is nonnegative, continuous with respect to* x *(at fixed values of* y*) and vanishes if and only if* x_k *is first-order critical.*

Considering that Algorithm 4.1 is aimed at tackling simple bounds constraints, it is natural to consider a stationarity measure which does not involve the solution of any additional optimization problem as in $\bar{\chi}_k$. We consider the norm of $\mathbf{red}(l_k^c, x_k)$ as a valid stationarity measure. Now we show the equivalence between the *Cauchy step*, criticality measure $\bar{\chi}_k$ and $\|\mathbf{red}(l_k^c, x_k)\|$:

Lemma 4.2. *Let $x_k \in \Omega_c$. Then, for any vector $s \in \mathbb{R}^{n_c}$, we have:*

$$\chi_k(x_k, \boldsymbol{red}(s, x_k), 1) \geq \chi_k(x_k, s, 1) \qquad \textit{and} \qquad \|\boldsymbol{red}(s, x_k)\| \geq \chi_k(x_k, s, 1).$$

Proof. First we suppose that $x_{lb,i} < x_{k,i} < x_{ub,i} \ \ \forall i \in \{1, \dots, n_c\}$. In this case $\mathbf{red}(s, x_k) = s$, thus $\chi_k(x_k, \mathbf{red}(s, x_k), 1) = \chi_k(x_k, s, 1)$. Now suppose that x_k lies on the boundary of Ω_c. Let $d \in \mathbb{R}^{n_c}$ be the vector that minimizes the problem $\chi_k(x_k, s, 1)$. Consider the coordinates i for which $x_{k,i} = x_{lb,i}$; then $d_i \geq 0$ and $s_i d_i \geq \min\{0, s_i\} d_i = \mathbf{red}(s, x_k)_i d_i$. Similarly, for the coordinates i such that $x_{k,i} = x_{ub,i}$, then $d_i \leq 0$ and

$$s_i d_i \geq \max\{0, s_i\} d_i = \mathbf{red}(s, x_k)_i d_i.$$

Altogether, this implies that $0 \geq s^\top d \geq (\mathbf{red}(s, x_k)^\top d)$. Thus, $\chi_k(x_k, \mathbf{red}(s, x_k), 1) \geq |(\mathbf{red}(s, x_k))^\top d| \geq |s^\top d| = \chi_k(x_k, s, 1)$.
For the second statement, suppose that $\bar{d} \in \mathbb{R}^{n_c}$ is the vector that minimizes the problem $\chi_k(x_k, \mathbf{red}(s, x_k), 1)$; then:

$$\chi_k(x_k, \mathbf{red}(s, x_k), 1) = |(\mathbf{red}(s, x_k)^\top \bar{d})| \leq \|\bar{d}\| \cdot \|\mathbf{red}(s, x_k)\| \leq \|\mathbf{red}(s, x_k)\|.$$

□

Lemma 4.3. *Suppose $x_k \in \Omega_c$ and $t > 0$. Then $p_k(x_k, l_k^c, t) = p_k(x_k, \boldsymbol{red}(l_k^c, x_k), t)$.*

Proof. Suppose first that $x_{lb,i} < x_{k,i} < x_{ub,i} \ \ \forall i \in \{1, \dots, n_c\}$; then $l_k^c = \mathbf{red}(l_k^c, x_k)$ and the condition holds. Then, suppose $x_{k,i} = x_{ub,i}$ for some i; in this case

$$[p_k(x_k, l_k^c, t)]_i = \max\{x_{lb_i}, x_{ub,i} - t \max\{0, l_{k,i}^c\}\}$$

and

$$\begin{aligned}[p_k(x_k, \mathbf{red}(l_k^c, x_k), t)]_i &= \max\{x_{lb_i}, x_{ub,i} - t \max\{0, \max\{0, l_{k,i}^c\}\}\} \\ &= \max\{x_{lb_i}, x_{ub,i} - t \max\{0, l_{k,i}^c\}\}.\end{aligned}$$

Thus, $[p_k(x_k, l_k^c, t)]_i = [p_k(x_k, \mathbf{red}((l_k^c, x_k), t)]]_i$. We can similarly prove the case $x_{k,i} = x_{ub,i}$:

$$[p_k(x_k, l_k^c, t)]_i = \min\{x_{ub_i}, x_{lb,i} - t\min\{0, l_{k,i}^c\}\}$$

and

$$\begin{aligned}[p_k(x_k, \mathbf{red}((l_k^c, x_k), t)]_i &= \min\{x_{ub_i}, x_{lb,i} - t\min\{0, \min\{0, l_{k,i}^c\}\}\} \\ &= \min\{x_{ub_i}, x_{lb,i} - t\min\{0, l_{k,i}^c\}\},\end{aligned}$$

which concludes the proof. □

Theorem 4.3. *Suppose $x_k \in \Omega_c$. If $\|\boldsymbol{red}(l_k^c, x_k)\| > 0$, then $\|x_k^C - x_k\| > 0$.*

Proof. This statement is proven by contradiction. Suppose that $\|\mathbf{red}(l_k^c, x_k)\| > 0$ and $x_k^C = x_k$. Note that $x_{k,i} > x_{lb,i}\ \forall i \in supp^+(\mathbf{red}(l_k^c, x_k))$, $x_{k,i} < x_{ub,i}\ \forall i \in supp^-(\mathbf{red}(l_k^c, x_k))$ and $|supp^+(\mathbf{red}(l_k^c, x_k)) \cup supp^-(\mathbf{red}(l_k^c, x_k))| > 0$. Now we determine the maximum scalar $\bar{t} > 0$ for which $x_k - \bar{t} \cdot \mathbf{red}(l_k^c, x_k) \in \Omega_c$. Let

$$\bar{t}^+ = \min_{i \in supp^+(\mathbf{red}(l_k^c, x_k))} \left\{ \frac{(x_{k,i} - x_{lb_i})}{[\mathbf{red}(l_k^c, x_k)]_i} \right\} > 0$$

and

$$\bar{t}^- = \min_{i \in supp^-(\mathbf{red}(l_k^c, x_k))} \left\{ \frac{(x_{x_k,i} - x_{ub_i})}{[\mathbf{red}(l_k^c, x_k)]_i} \right\} > 0.$$

Above, if $|supp^+(\mathbf{red}(l_k^c, x_k))| = 0$ or $|supp^-(\mathbf{red}(l_k^c, x_k))| = 0$ then we assume $\bar{t}^+ = \infty$ or $\bar{t}^- = \infty$, respectively. Let $\bar{t} = \min\{\bar{t}^+, \bar{t}^-\}$. From Lemma 4.3, $x_k - \bar{t} \cdot \mathbf{red}(l_k^c, x_k) = p_k(x_k, l_k^c, \bar{t})$, so that $p_k(x_k, l_k^c, \bar{t})$ is a feasible solution of the problem used in the definition of the *Cauchy step* (Definition 4.7). Thus, $m_k(x_k - \bar{t} \cdot \mathbf{red}(l_k^c, x_k)) \geq m_k(x_k^C)$. However,

$$m_k(x_k - \bar{t} \cdot \mathbf{red}(l_k^c, x_k)) = -\bar{t}\|\mathbf{red}(l_k^c, x_k)\|^2 > 0 \geq m_k(x_k^C) = 0,$$

which contradicts the hypothesis that $x_k^C = x_k$. □

The following two corollaries summarize the relationship between the three stationarity criteria.

Corollary 4.1. *Suppose $x_k \in \Omega_c$. If $\|\boldsymbol{red}(l_k^c, x_k)\| > 0$, there exists a constant $\kappa_{frd} \in (0,1)$ such that $\|x_k^C - x_k\| \geq \Delta_k^c \kappa_{frd}$ (the subindex frd stands for fraction of delta).*

Proof. Since $\|\mathbf{red}(l_k^c, x_k)\| > 0$, then $\Delta_k^c > \|x_k^C - x_k\| > 0$ by Theorem 4.3. Then,

$$0 < \kappa_{frd} \leq \frac{\|x_k^C - x_k\|}{\Delta_k^c} < 1.$$

□

Corollary 4.2. *Suppose $x_k \in \Omega_c$. If $\|\boldsymbol{red}(l_k^c, x_k)\| > 0$, there exists a constant $\kappa_{cri} \in (0,1)$ such that $\bar{\chi}_k \geq \kappa_{cri} \|\boldsymbol{red}(l_k^c, x_k)\|$ (the subindex cri stands for criticality).*

Proof. Since $\|\mathbf{red}(l_k^c, x_k)\| > 0$, then $\Delta_k^c > \|x_k^C - x_k\| > 0$ by Theorem 4.3. Since Ω_c is convex and $x_k, x_k^C \in \Omega_c$, any point on the line $x_k + (x_k^C - x_k)h \in \Omega_c, \ \forall h \in [0,1]$. Then the point $x_k + (x_k^C - x_k)/\max\{1.2, \|x_k^C - x_k\|\} \in \hat{B}_c(x_k, 1) \cap \Omega_c$, thus

$$\bar{\chi}_k \geq \frac{|(x_k^C - x_k)^\top l_k^c|}{\max\{1.2, \|x_k^C - x_k\|\}} > 0.$$

Finally, from Lemma 4.2 we have $\|\mathbf{red}(l_k^c, x_k)\| \geq \bar{\chi}_k$, thus

$$0 < \kappa_{cri} \leq \frac{\bar{\chi}_k}{\|\mathbf{red}(l_k^c, x_k)\|} < 1.$$

□

4.2.2 Conditions for Mixed-Integer Stationarity

In this section we complete the proof of convergence of Algorithm 4.1 to a first-order mixed-integer stationary point. Note that we use the global error constants κ_{ef}, κ_{eg} described in Section 3.2.4. We first show that if the current iterate is not a first-order critical point, Algorithm 4.2 converges in a finite number of iterations:

Lemma 4.4. *Let $d_1, d_2 \in \mathbb{R}^{n_c}$ and $x_k \in \Omega_c$. The following holds:*

$$\|\boldsymbol{red}(d_1, x_k) - \boldsymbol{red}(d_2, x_k)\| \leq \|d_1 - d_2\|.$$

Proof. This statement is proven by looking at the element-wise squared difference of entries between $\mathbf{red}(d_1, x_k)$ and $\mathbf{red}(d_2, x_k)$. If x_k is not at the boundary of Ω_c, then

$\mathbf{red}(d_1, x_k) = d_1$ and $\mathbf{red}(d_2, x_k) = d_2$, thus the condition holds. Now suppose that $x_{k,i} = x_{ub,i}$ then

$$([\mathbf{red}(d_1, x_k)]_i - [\mathbf{red}(d_2, x_k)]_i)^2 = (\max\{d_{1,i}, 0\} - \max\{d_{2,i}, 0\})^2$$

for which we have 3 possible cases: (1) $d_{1,i}, d_{2,i} \geq 0$, (2) $d_{1,i}, d_{2,i} \leq 0$ and (3) $d_{1,i} > 0 > d_{2,i}$. In the first case it is clear that $([\mathbf{red}(d_1, x_k)]_i - [\mathbf{red}(d_2, x_k)]_i)^2 = (d_{1,i} - d_{2,i})^2$. In the second case we have that $([\mathbf{red}(d_1, x_k)]_i - [\mathbf{red}(d_2, x_k)]_i)^2 = 0 \leq (d_{1,i} - d_{2,i})^2$. In case (3) we have that $([\mathbf{red}(d_1, x_k)]_i - [\mathbf{red}(d_2, x_k)]_i)^2 = (d_{1,i} - 0)^2 \leq (d_{1,i} - d_{2,i})^2$. The same analysis can be done for the case $x_{k,i} = x_{lb,i}$ with the same result, thus $([\mathbf{red}(d_1, x_k)]_i - [\mathbf{red}(d_2, x_k)]_i)^2 \leq (d_{1,i} - d_{2,i})^2 \ \forall i \in \{1, \dots n_c\}$, which completes the proof. □

Lemma 4.5. *If $\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| > 0$ then the **CriticalityTest** (Algorithm 4.2) terminates in a finite number of iterations.*

Proof. Suppose that $\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| > 0$ and the **CriticalityTest** iterates indefinitely. Then, the following relationship between the gradient of the model and the trust-region radius holds:

$$\tilde{\Delta}_k = \omega^{i-1}\Delta_k^{icb} > \mu\|\mathbf{red}(\tilde{l}_k^c, x_k)\| \ \ \forall i > 0,$$

and $\lim_{i\to\infty}\|\mathbf{red}(\tilde{l}_k^c, x_k)\| = 0$. We show that this would imply $\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| = 0$. To see this, notice that:

$$\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| \leq \|\mathbf{red}(\nabla_x f(x_k, y_k), x_k) - \mathbf{red}(\tilde{l}_k^c, x_k))\| + \|\mathbf{red}(\tilde{l}_k^c, x_k))\|. \tag{4.6}$$

Taking into account that for every iteration $i > 0$ the resulting model $\tilde{m}_k$ is fully-linear with respect to $\tilde{\Delta}_k$, and using Lemma 4.4, we have $\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k) - \mathbf{red}(\tilde{l}_k^c, x_k))\| \leq \kappa_{eg}\omega^{i-1}\Delta_k^{icb}$. Thus, (4.6) implies:

$$\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| \leq (\kappa_{eg} + \mu^{-1})\omega^{i-1}\Delta_k^{icb}$$

so that $\lim_{i\to\infty}\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| = 0$, which contradicts the initial assumption. Thus, the **CriticalityTest** must converge in a finite number of iterations. □

This result implies that unless $\mathbf{red}(\nabla_x f(x_k, y_k), x_k) = 0$, $\|\mathbf{red}(l_k^c, x_k)\| > 0$ and we can improve the objective function by searching in the continuous manifold. With this result we present the minimum expected improvement for the *Generalized Cauchy step*:

Theorem 4.4. *Let m_k be the surrogate model and x^C the Generalized Cauchy Point. Then $m_k(x_k, y_k) - m_k(x_k^C, y_k) \geq \kappa_{frd}\bar{\chi}_k \min\{\Delta_k^c, 1\}$.*

Proof. Recall that $m_k(x_k + s_c, y_k) = m_k(x, k) + s_c^\top l_k^c$ due to the lack of continuous quadratic terms in 3.4. From Theorem 4.1 we have that

$$m_k(x_k^C, y_k) - m_k(x_k, y_k) = |(x_k^C - x_k)^\top l_k^c| = \chi_k(x_k, l_k^c, \|x_k^C - x_k\|).$$

First we consider the case $\|x_k^C - x_k\| \geq 1$. From Lemma 4.1, $\chi_k(x_k, l_k^c, \|x_k^C - x_k\|) \geq \bar{\chi}_k$, thus $m_k(x_k^C, y_k) - m_k(x_k, y_k) \geq \bar{\chi}_k \geq \kappa_{frd}\bar{\chi}_k$. Next, we consider the case $\|x_k^C - x_k\| < 1$. From Lemma 4.1 we have $\chi_k(x_k, l_k^c, \|x_k^C - x_k\|) \geq \|x_k^C - x_k\|\bar{\chi}_k$. Recalling Corollary 4.1, we then obtain:

$$m_k(x_k^C, y_k) - m_k(x_k, y_k) = \chi_k(x_k, l_k^c, \|x_k^C - x_k\|) \geq \bar{\chi}_k\|x_k^C - x_k\| \geq \kappa_{frd}\bar{\chi}_k\Delta_k^c.$$

□

It is not required to compute the *Cauchy step* in every iteration to evaluate continuous stationarity. It is enough to relate the improvement of *Generalized Cauchy step* and the solution of optimization subproblems in the **CandidateComputation** and **RescueProcedure**:

Lemma 4.6. *At every iteration $k > 0$, the predicted improvement is bounded by a fraction of the mixed-integer Cauchy step: $m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z) \geq \kappa_{frd}\bar{\chi}_k \min\{\Delta_k^c, 1\}$.*

Proof. From line 8 of Algorithm 4.3 the following inequalities hold: $m_k(x_k + s_c, y_k + s_z) \leq m_k(x_k^C, y_k)$ and $m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z) \geq m_k(x_k, y_k) - m_k(x_k^C, y_k)$. From Theorem 4.4 we get $m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z) \geq \kappa_{frd}\bar{\chi}_k \min\{\Delta_k^c, 1\}$. This concludes the proof. □

Lemma 4.7. *At every iteration $k > 0$ the predicted improvement is bounded by the partial mixed-integer stationary parameter Θ_k: $m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z) \geq \Theta_k$.*

Proof. In case $\Theta_k > 0$ the vectors (x, y) that yield its value corresponds to a feasible solution of optimization problem computed in Algorithm 4.3- Line 8. On the other hand, when $\Theta_k \leq 0$ it corresponds to an inactive lower-bound for $m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z)$ as the predicted improvement is always nonnegative. □

Now, we relate the properties of the model approximation in Algorithm 4.6 with the convergence of Algorithm 4.1.

Lemma 4.8. *If the model m_k is mixed-integer fully linear and the continuous trust-region radius satisfies*

$$\Delta_k^c \leq \min\left\{1, \frac{\kappa_{frd}\Phi_k(1-\eta_1)}{\kappa_{ef}}\right\},$$

then iteration k is very-successful, successful *or* acceptable.

Proof. First consider the *Cauchy step*. As $\Delta_k^c \leq 1$, the expected improvement is bounded by:

$$m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z) \geq \kappa_{frd}\bar{\chi}_k \min\{\Delta_k^c, 1\} = \kappa_{frd}\bar{\chi}_k\Delta_k^c.$$

Now, we consider the minimization of m_k in the region $(\hat{B}_v(x_k, y_k, \Delta_k^c, 1) \cap \Omega_m$ if function is LQMI (Line 5, Algorithm 4.3), or, $\hat{B}_c(x, \Delta_k^c) \times \Omega_d^G(y_k, M_k)) \cap \Omega_m$ otherwise (**RescueProcedure**, Algorithm 4.1). A bound on the update parameter ρ resulting from this optimization can be computed as follows:

$$|\rho - 1| \leq \left|\frac{m_k(x_k + s_c, y_k + s_z) - f(x_k + s_c, y_k + s_z)}{m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z)}\right|.$$

Recalling the fully-linearity condition and the bound on Δ_k^c, we get:

$$|\rho - 1| \leq \frac{\kappa_{ef}(\Delta_k^c)^2}{\kappa_{frd}\bar{\chi}_k\Delta_k^c} \leq \frac{\|\mathbf{red}(l_k^c, x_k)\|(1-\eta_1)}{\bar{\chi}_k} \leq \frac{\Phi_k(1-\eta_1)}{\bar{\chi}_k}.$$

From Lemma 4.2 we have that $\bar{\chi}_k \leq \|\mathbf{red}(l_k^c, x_k)\| \leq \Phi_k$ thus $|\rho - 1| \leq (1 - \eta_1)$. We remark that it is possible that such improvement is not accepted in Algorithm 4.3 if

the model reduction is lower than ϵ_a. Nonetheless, as model m_k is fully-linear the **RescueProcedure** is invoked and a solution on $\hat{B}_c(x_k, \Delta_k^c) \times \{y_k\}$ is identified.

Now, suppose that $\Phi_k = \bar{\Theta}_k$. It implies that $\bar{\Theta}_k > 0$, thus $\bar{\Theta}_k = \Theta_k \geq \epsilon_a$. The expected improvement given by the partial mixed-integer parameters is bounded by:

$$m_k(x_k, y_k) - m_k(x_k + s_c, y_k + s_z) \geq \bar{\Theta}_k \geq \epsilon_a.$$

Then, a bound in the update parameter is given by $|\rho - 1| \leq \frac{\kappa_{ef}(\Delta_k^c)^2}{\bar{\Theta}_k}$ during the optimization procedure related to the **RescueProcedure**. As $\Delta_k^c \leq 1$ we have that $(\Delta_k^c)^2 \leq \Delta_k^c$ and:

$$|\rho - 1| \leq \frac{\kappa_{ef}\Delta_k^c}{\bar{\Theta}_k} \leq \frac{\kappa_{ef}\Delta_k^c}{\kappa_{frd}\bar{\Theta}_k} \leq \frac{\Phi_k(1-\eta_1)}{\bar{\Theta}_k} \leq (1-\eta_1).$$

In consequence, $\rho \geq \eta_0$ and $f(x_k + s_c, y_k + s_z) < f(x_k, y_k)$, thus iteration k is either *very-successful*, *successful* or *acceptable*. □

Next, we prove that Algorithm 4.1 converges with respect to the continuous variables. Let $\mathcal{S}_{imp}$ be the set of *very-successful*, *successful* and *acceptable* iterations.

Lemma 4.9. *The number of* successful *and* acceptable *iterations for which the new iterate is selected from previous samples (Lines 4 and 14 in Algorithm 4.3 or Line 1 in Algorithm 4.5) is finite.*

Proof. Let N_{IS} be the number of *successful* or *acceptable* iterations where the candidate is retrieved by samples. Let us consider the objective function improvement obtained by the iterates of Algorithm 4.1:

$$\lim_{k\to\infty} f(x_o, y_o) - f(x_k, y_k) = \sum_{j\in\mathcal{S}_{imp}} f(x_j, y_j) - f(x_{j+1}, y_{j+1}) \geq N_{IS}\epsilon_a.$$

By Assumption 1.1, $\lim_{k\to\infty} f(x_o, y_o) - f(x_k, y_k)$ is bounded, therefore N_{IS} must be finite. □

Lemma 4.10. *There exists a constant $\eta_{res} > 0$ such that*

$$f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \eta_{res}(m_k(x_k, y_k) - m_k(x_{k+1}, y_{k+1}))$$

for every acceptable *iteration that is selected from the solution of the Problem 4.1-4.4.*

Proof. A new candidate is only accepted during **RescueProcedure** if it yields an improvement of the objective function $f(x,y)$. As $m_k(x_k,y_k) - m_k(x_{k+1},y_{k+1}) \geq 0$ for Problem 4.1-4.4 and $f(x_k,y_k) - f(x_{k+1},y_{k+1}) > 0$, we have that

$$\frac{f(x_k,y_k) - f(x_{k+1},y_{k+1})}{m_k(x_k,y_k) - m_k(x_{k+1},y_{k+1})} \geq \eta_{res} > 0.$$

□

Lemma 4.11. $\lim_{k\to\infty} \Delta_k^c = 0$.

Proof. For every *very-successful*, *successful* or *acceptable* iteration, the predicted improvement is given by:

$$f(x_k,y_k) - f(x_{k+1},y_{k+1}) \geq \eta_{bnd}(\min\{m_k(x_k,y_k) - m_k(x_{k+1},y_{k+1}), \epsilon_a\}), \tag{4.7}$$

where $\eta_{bnd} = \min\{\eta_{res}, \eta_0\}$. Note that the term ϵ_a corresponds to the iterations in $\mathcal{S}_{imp}$ where the point (x_{k+1},y_{k+1}) is selected from samples. From Lemmas 4.6 and 4.7, we have:

$$\begin{aligned} m_k(x_k,y_k) - m_k(x_{k+1},y_{k+1}) &\geq \kappa_{frd}\bar{\chi}_k \min\{\Delta_k^c, 1\} \geq \\ &\kappa_{frd}\kappa_{cri}\|\mathbf{red}(l_k^c, x_k)\| \min\{\Delta_k^c, 1\} \end{aligned}$$

and

$$m_k(x_k,y_k) - m_k(x_{k+1},y_{k+1}) \geq \Theta_k \geq \kappa_{frd}\kappa_{cri}\Theta_k.$$

As a consequence, the bound 4.7 can be expressed as

$$\begin{aligned} &f(x_k,y_k) - f(x_{k+1},y_{k+1}) \\ &\geq \eta_{bnd} \min\{\kappa_{frd}\kappa_{cri} \max\{\|\mathbf{red}(l_k^c, x_k)\| \min\{\Delta_k^c, 1\}, \Theta_k\}, \epsilon_a\}. \end{aligned} \tag{4.8}$$

Now, we study the behavior of bound 4.8 with respect to the parameter Φ_k. We remark that Algorithm 4.2 guarantees that $\Phi_k \geq \min\{\epsilon_c, \mu^{-1}\Delta_k^c\}$. First, consider the set of iterations $\{\ell_j\}$ such that $\Phi_{\ell_j} = \bar{\Theta}_{\ell_j}$, then $\Theta_{\ell_j} = \bar{\Theta}_{\ell_j}$ and $\Phi_{\ell_j} \geq \|\mathbf{red}(l_{\ell_j}^c, x_{\ell_j})\| \geq$

$\|\mathbf{red}(l^c_{\ell_j}, x_{\ell_j})\| \min\{\Delta^c_{\ell_j}, 1\}$. As a consequence, for every ℓ_j the bound 4.8 is equivalent to:

$$f(x_{\ell_j}, y_{\ell_j}) - f(x_{\ell_j+1}, y_{\ell_j+1}) \geq \eta_{bnd} \min\{\kappa_{frd}\kappa_{cri}\Phi_{\ell_j}, \epsilon_a\} \geq \eta_{bnd} \min\{\kappa_{frd}\kappa_{cri} \min\{\mu^{-1}\Delta^c_{\ell_j}, \epsilon_c\}, \epsilon_a\}.$$

Now, consider the iterations $\{t_j\}$ where $\Phi_{t_j} = \|\mathbf{red}(l^c_{t_j}, x_{t_j})\| \geq \min\{\mu^{-1}\Delta^c_{t_j}, \epsilon_c\}$. In these cases, the bound 4.8 is equivalent to:

$$f(x_{t_j}, y_{t_j}) - f(x_{t_j+1}, y_{t_j+1}) \geq \eta_{bnd} \min\{\{\kappa_{frd}\kappa_{cri} \max\{\min\{\mu^{-1}\Delta^c_{t_j}, \epsilon_c\} \min\{\Delta^c_{t_j}, 1\}, \Theta_{t_j}\}, \epsilon_a\} \geq \eta_{bnd} \min\{\{\kappa_{frd}\kappa_{cri} \min\{\mu^{-1}\Delta^c_{t_j}, \epsilon_c\} \min\{\Delta^c_{t_j}, 1\}, \epsilon_a\}.$$

We highlight that S_{imp} corresponds to the union of both sequences $\{\ell_j\}$ and $\{t_j\}$. From Assumption 1.1 the series $\sum_{k \in S_{imp}} f(x_k, y_k) - f(x_{k+1}, y_{k+1})$ is convergent, thus the limiting values of $f(x_k, y_k) - f(x_{k+1}, y_{k+1})$ are bounded by 0. The latter condition is only attained if $\lim_{k\to\infty} \Delta^c_k = 0$, completing the proof. □

Finally, we prove that $\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$, and there exists some iteration number after which y_k is fixed to a value y^*. The first part consists in proving that there exists one accumulation point (x^*, y^*) such that $\Phi_k = 0$.

Lemma 4.12. $\liminf_{k\to\infty} \Phi_k = 0$.

Proof. Assume, for establishing a contradiction that there exists a $\kappa_1 > 0$ such that $\Phi_k > 0, \forall k \geq 0$. For Φ_k to be bounded by κ_1 it is necessary that either $\|\mathbf{red}(l^c, x_k)\| > \kappa_1$ or $\bar{\Theta}_k > \kappa_1, \ \forall k > 0$. We use the *Criticality Step* to derive a relationship between κ_1, Φ_k and Δ^c_k. There exists two possible scenarios:

- $\Delta^c_k \geq \min\{\Delta^{icbc}_k, \beta\|\mathbf{red}(l^c, x_k)\|\}$ if the *CriticalityTest* is called.
- $\Delta^c_k = \Delta^{icbc}_k$ otherwise.

By Lemma 4.8 and the assumption that $\Phi_k > \kappa_1$, whenever Δ^c_k falls bellow $\bar{\kappa}_2 = \min\left\{1, \frac{\kappa_{frd}\ \kappa_1(1-\eta_1)}{\kappa_{ef}}\right\}$, the iteration kth cannot be *unsuccessful*. Thus $\Delta^{icbc}_{k+1} \geq \Delta_k$ and $\Delta^{icbc}_k \geq \min\{\gamma_0\bar{\kappa}_2, \Delta^{icbc}_0\} \ \forall k > 0$.

We first consider the case where $\bar{\Theta}_k > \kappa_1, \ \forall k > 0$. As $\bar{\Theta}_k$ is either 0 or greater than $\max\{\epsilon_c, \epsilon_a\}$ (Definition 4.1), we have that $\bar{\Theta}_k \geq \max\{\epsilon_a, \epsilon_c, \kappa_1\} \geq \epsilon_c \ \forall k > 0$, thus $\Phi_k \geq \epsilon_c \ \forall k > 0$. As a consequence, the **CriticalityTest** is never invoked and

$$\Delta_k^c \geq \min\{\Delta_0^{icbc}, \gamma_0 \bar{\kappa}_2\}.$$

On the other hand, if $\|\mathbf{red}(l^c, x_k)\| > \kappa_1$ we have that for every iteration k, whether the **CriticalityTest** is invoked or not, the following condition holds:

$$\Delta_k^c \geq \min\{\Delta_k^{icbc}, \beta\|\mathbf{red}(l^c, x_k)\| \geq \min\{\Delta_k^{icbc}, \beta\kappa_1\}.$$

As a result, Δ_k^c must be bounded: $\Delta_k^c \geq \min\{\Delta_0^{icbc}, \beta\kappa_1, \gamma_0\bar{\kappa}_2\}, \ \forall k > 0$, which contradicts Lemma 4.11.

□

Lemma 4.13. *For a subsequence $\{k_i\}$ such that*

$$\lim_{i\to\infty} \Phi_{k_i} = 0,$$

it also holds that

$$\lim_{i\to\infty} \|\boldsymbol{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| = 0$$
$$\textit{and} \quad \lim_{i\to\infty} \chi_{k_i}(x_{k_i}, \nabla_x f(x_{k_i}, y_{k_i}), 1) = 0.$$

Proof. First, note that for a large k_i we have that $\Phi_{k_i} < \epsilon_c$ as the limit value of the subsquence is 0; thus, the model m_{k_i} is mixed-integer fully-linear and $\Delta_{k_i}^c \leq \mu\|\mathbf{red}(l_{k_i}^c, x_{k_i})\| = \mu\Phi_k$ (Algorithm 4.1, *Criticality Step*). From Lemma 4.4 we have that

$$\begin{aligned}\|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i}) - \mathbf{red}(l_{k_i}^c, x_{k_i}))\| &\leq \|\nabla_x f(x_{k_i}, y_{k_i}) - l_{k_i}^c)\| \\ &\leq \kappa_{eg}\Delta_{k_i}^c \leq \kappa_{eg}\mu\|\mathbf{red}(l_{k_i}^c, x_{k_i})\|.\end{aligned}$$

This bound can be used to compute an upper bound on the norm of the reduced gradient of f at the point x_{k_i}, y_{k_i}:

$$\begin{aligned} &\|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| \\ \leq \|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i}) - \mathbf{red}(l^c_{k_i}, x_{k_i}))\| &+ \|\mathbf{red}(l^c_{k_i}, x_{k_i})\| \\ &\leq (\kappa_{eg}\mu + 1)\|\mathbf{red}(l^c_{k_i}, x_{k_i})\|. \end{aligned} \tag{4.9}$$

As a consequence $\lim_{i\to\infty}\|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| = 0$.
Finally, from Lemma 4.2 $\|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| \geq \chi_k(x_{k_i}, \nabla_x f(x_{k_i}, y_{k_i}), 1)$, thus,

$$\lim_{i\to\infty} \chi_{k_i}(x_{k_i}\nabla_x f(x_{k_i}, y_{k_i}), 1) = 0.$$

□

Theorem 4.5. *The number of iterations for which $\|y_{k+1} - y_k\|_1 \geq 1$ is bounded, i.e., there exists an iteration number $k_{mis} > 0$ such that $y_k = y_{k_{mis}}$ for all $k \geq k_{mis}$.*

Proof. According to the **CandidateComputation** and **RescueProcedure**, a change in the integer coordinates can only be accepted when the solution of the surrogate algorithm yields $m_k(x_k, y_k) - m_k(x_{k+1}, y_{k+1}) \geq \epsilon_a$, or, the search on previously sampled points yields $f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \epsilon_a$. Thus, the objective reduction in each iteration in the subsequence such that the discrete coordinates change, $\mathcal{S}_{ych} \subset S_{imp}$, is bounded from below by $f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \eta_{bnd}\epsilon_a > 0$. If $|\mathcal{S}_{ych}| = \infty$, then $\lim_{k\to\infty} f(x_k, y_k) = -\infty$, but this would contradict Assumption 1.1. □

Now we prove that there exist at least one accumulation point (x^*, y^*) which is stationary with respect to the continuous variables, or $\|\mathbf{red}(\nabla_x f(x^*, y^*), x^*)\| = 0$.

Lemma 4.14. $\liminf_{k\to\infty}\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| = 0$.

Proof. Assume, for establishing a contradiction that there exists a bound $\kappa_1 > 0$ on the reduced gradient of f such that $\epsilon_c > \kappa_1$, $\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| > \kappa_1 \ \forall k \geq 0$. Now we establish a relationship between κ_1 and $\|\mathbf{red}(l^c_k, x_k)\|$. From Lemma 4.12 and its proof we know that there exists at least an iteration k_i for which $\Phi_{k_i} < \frac{\kappa_1}{2+\kappa_{eg}\mu}$. Considering that κ_1 is strictly smaller than ϵ_c and ϵ_a we have that:

- $\tilde{\Theta}_{k_i} = 0$ thus $\Phi_{k_i} = \|\mathbf{red}(l^c_{k_i}, x_{k_i})\|$,

- the model m_{k_i} is fully-linear, and
- the chain of inequalities (4.9) holds.

As a consequence for the iteration k_i we have that:

$$\kappa_1 \leq \|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| \leq (1+\mu\kappa_{eg})\|\mathbf{red}(l^c_{k_i}, x_{k_i})\| < \frac{1+\mu\kappa_{eg}}{2+\mu\kappa_{eg}}\kappa_1,$$

which contradicts the initial assumption. □

Theorem 4.6. *Under Assumptions 1.1 and 1.3 the sequence of iterates generated by Algorithm 4.1 satisfies:*

$$\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$$

Proof. Let k_{mis} be defined as in Theorem 4.5. Assume, for establishing a contradiction there exist a sequence of iterations $\{\ell_j\} \subset \mathcal{S}_{imp}$, such that $\ell_1 > k_{mis}$ and $0 < \epsilon_o < \chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) \leq \|\mathbf{red}(\nabla_x f(x_{\ell_j}, y_{\ell_j}), x_{\ell_j})\|$. The relationship between the criticality measure $\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1)$ and the norm of the reduced gradient of $\nabla_x f(x_{\ell_j}, y_{\ell_j})$ is given by Lemma 4.2.
Now we establish a relationship between ϵ_o and Φ_{ℓ_j}. From Lemma 4.13 we obtain that $\Phi_{\ell_j} \geq \epsilon$ for some $\epsilon > 0$ and a sufficiently large j. Without loss of generality, we select an ϵ such that $\epsilon < \min\left\{\frac{\epsilon_o}{(2+\mu\kappa_{eg})}, \epsilon_c\right\}$. In this way we have that

$$\Phi_{\ell_j} \geq \min\left\{\epsilon_c, \frac{\epsilon_o}{(2+\kappa_{eg}\mu)}\right\} > \epsilon. \tag{4.10}$$

We define $t_j \in \mathcal{S}_{imp}, t_j > \ell_j$ as the first iteration after ℓ_j for which $\Phi_{t_j} < \epsilon$. Such an iteration exists as consequence of Lemma 4.12. Let us define some sequences of iterations (indexed by j) $\mathcal{K}_j = \{k \in \mathcal{S}_{imp} \mid \ell_j \leq k < t_j\} \cup \{t_j\}$. Note that for every sequence $\mathcal{K}_j$ we have the following properties:

- For every $k \in [\ell_j, t_j)$ we that $\Phi_k \geq \epsilon$, and
- the last element is t_j.

We remark that in iteration t_j the model m_{t_j} is fully-linear, $\mu\|\mathbf{red}(l^c_{t_j}, x_{t_j})\| \geq \Delta^c_{t_j}$ and $\bar{\Theta}_k = 0$, as $\Phi_{t_j} < \epsilon_c$. For every $k \in \mathcal{K}_j$, $\ell_j \leq k < t_j$ there exists a bound on Δ^c_k given

by equation (4.7):

$$f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \eta_{bnd}(m_k(x_k, y_k) - m_k(x_{k+1}, y_{k+1}))$$
$$f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \eta_{bnd} \max\{\kappa_{frd}\kappa_{cri}\|\mathbf{red}(l^c_{\ell_j}, x_{\ell_j})\| \min\{\Delta^c_k, 1\}, \Theta_k\}.$$

As $\kappa_{frd}\kappa_{cri} \min\{\Delta^c_k, 1\} \in (0, 1]$ then

$$f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \eta_{bnd}\kappa_{frd}\kappa_{cri} \max\{\|\mathbf{red}(l^c_{\ell_j}, x_{\ell_j})\|, \Theta_k\} \min\{\Delta^c_k, 1\}.$$

The latter is equivalent to $f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \eta_{bnd}\Phi_k \min\{\Delta^c_k, 1\}$. For a large $\ell_j \to \infty$ we have that $\forall k \in [\ell_j, t_j)$ the trust-region radius is small $\Delta^c_k < 1$ and $\Delta^c_k \leq \frac{f(x_k,y_k)-f(x_{k+1},y_{k+1})}{\eta_{bnd}\kappa_{frd}\kappa_{cri}\epsilon}$. Thus, we can derive the following bound for $\|x_{t_j} - x_{\ell_j}\|$:

$$\|x_{t_j} - x_{\ell_j}\| \leq \sum_{k \in \mathcal{K}_j}^{t_j-1} \|x_k - x_{k+1}\| \leq \sum_{k \in \mathcal{K}_j}^{t_j-1} \Delta^c_k$$
$$\leq \sum_{k \in \mathcal{K}_j}^{t_j-1} \frac{f(x_k, y_k) - f(x_{k+1}, y_{k+1})}{\eta_{bnd}\kappa_{frd}\kappa_{cri}\epsilon} = \frac{f(x_{\ell_j}, y_{\ell_j}) - f(x_{t_j}, y_{t_j})}{\eta_{bnd}\kappa_{frd}\kappa_{cri}\epsilon}.$$

As $\lim_{\ell_j \to \infty} f(x_{\ell_j}, y_{\ell_j}) - f(x_{t_j}, y_{t_j}) = 0$ (from Assumption 1.1), $\lim_{\ell_j \to \infty} \|x_{\ell_j} - x_{t_j}\| = 0$. Now we derive an upper-bound for the stationarity measure of f at every iteration in the sequence $\{\ell_j\}$:

$$\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) \leq |\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) - \chi_{t_j}(x_{t_j}, \nabla_x f(x_{t_j}, y_{t_j}), 1)|$$
$$+\chi_{t_j}(x_{t_j}, \nabla_x f(x_{t_j}, y_{t_j}), 1).$$

From Lemma 4.2 $\|\mathbf{red}(\nabla_x f(x_{t_j}, y_{t_j}), x_{t_j})\| \geq \chi_{t_j}(x_{t_j}, \nabla_x f(x_{t_j}, y_{t_j}), 1)$. Considering Lemma 4.4 and the fully-linearity of t_j, the bound can be rewritten as:

$$\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) \leq |\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) - \chi_{t_j}(x_{t_j}, \nabla_x f(x_{t_j}, y_{t_j}), 1)|$$
$$+\|\mathbf{red}(\nabla_x f(x_{t_j}, y_{t_j}), x_{t_j}) - \mathbf{red}(l^c_{t_j}, x_{t_j})\| + \|\mathbf{red}(l^c_{t_j}, x_{t_j})\|. \tag{4.11}$$

Theorem 4.2 guarantees that

$$\lim_{j \to \infty} |\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) - \chi_{t_j}(x_{t_j}, \nabla_x f(x_{t_j}, y_{t_j}), 1)| = 0 \tag{4.12}$$

whenever $y_{t_j} = y_{\ell_j}$ and $\lim_{j\to\infty}\|x_{t_j} - x_{\ell_j}\| = 0$; thus the first term of the right-hand side of Equation 4.11 goes to 0. Now we derive an upper bound for the remaining terms of Equation 4.11. Since the model t_j is fully-linear we have $\|\mathbf{red}(\nabla_x f(x_{t_j}, y_{t_j}), x_{t_j}) - \mathbf{red}(l^c_{t_j}, x_{t_j})\| \leq \kappa_{eg}\Delta^c_{t_j}$. In addition $\Phi_{t_j} < \epsilon_c$, therefore $\Delta^c_{t_j} \leq \mu\|\mathbf{red}(l^c_{t_j}, x_{t_j})\|$. As a consequence $\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1)$ is bounded by:

$$\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) \leq \kappa_{eg}\Delta^c_{t_j} + \|\mathbf{red}(l^c_{t_j}, x_{t_j})\| \leq (1 + \mu\kappa_{eg})\epsilon.$$

Taking into account that $\epsilon < \frac{\epsilon_o}{(1+\mu\kappa_{eg})}$ (Equation 4.10) the upper bound is transformed into

$$\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) < \epsilon_o.$$

But this contradicts the assumption on a limiting bound for $\chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1)$. Therefore, for any given $\epsilon_o > 0$ there exists an $N \in \mathbb{N}$, $N \geq k_{mis}$ such that:

$$\chi_k(x_k, \nabla_x f(x_k, y_k), 1) < \epsilon_o \;\; \forall k > N,$$

which implies that $\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$. □

Now, having proved that Algorithm 4.1 is convergent to a stationary solution with respect to the continuous variables and the box Ω_c, we will show that it also attains a solution which is an ϵ local minimizer with respect to the integer variables and the set Ω_z. The first step is to show that there exist an infinite number of iterations where $\Delta^z = 1$, which facilitates the analysis over the sets $\Omega^G_d(y_k, M_k)$, $\Omega^{LQMI}_d(y_k)$ and $\Omega^Q_d(y_k)$:

Lemma 4.15. *There exists an infinite number of iterations for which* $\Delta^z_k = 1$.

Proof. This statement is proved by contradiction. Suppose there exists a $\hat{k}$ that is the last iteration such that $\Delta^z_k = 1$. From Lemma 4.6 we have that

$$\forall\epsilon > 0(\exists N \in \mathbb{N} \mid \Delta^c_k < \epsilon \;\; \forall k \geq N).$$

Let $N(\epsilon)$ be defined as $\min\{N \in \mathbb{N} \mid \Delta^c_k < \epsilon \;\; \forall k \geq N\}$. Function $N(\epsilon)$ has the following properties:

$$N(\epsilon_1) \geq N(\epsilon_2) \;\text{ if }\; \epsilon_1 < \epsilon_2, \tag{4.13}$$

$$\forall \epsilon_1, \epsilon_2 \in \mathbb{R}^+,\ \epsilon_1 < \epsilon_2,\ N(\epsilon_1) > N(\epsilon_2)\ (\exists k \in [N(\epsilon_2), N(\epsilon_1)) \mid \Delta_k^c \in (\epsilon_1, \epsilon_2]). \tag{4.14}$$

We define the sequences $\{\epsilon_i\}$ and $\{N_i\}$ as follows:

- $\epsilon_0 = \Delta_{max}^c + 1e^{-20}$ and $N_0 = 0$;
- $\epsilon_{i+1} = \text{argmin}_{\epsilon \in \mathbb{R}^+, \epsilon < \epsilon_i} \{N(\epsilon) \mid N(\epsilon) > N_i\}$ and $N_{i+1} = N(\epsilon_{i+1})$.

For the sequence ϵ_i we define the constant $\bar{\beta} = \sup_{i \geq 0} \frac{\epsilon_{i+1}}{\epsilon_i}$. As $\epsilon_i > \epsilon_{i+1}\ \forall i \geq 0$, then $\bar{\beta} < 1$. Consider an index $\hat{\imath} > 0$ such that $N_{\hat{\imath}} > \hat{k}$ and $\epsilon_{\hat{\imath}} < \Delta_{max}^c / \gamma_1$. From Equation 4.14 we know that there exists an iteration $\tilde{k} \in [N_{\hat{\imath}}, N_{\hat{\imath}+1})$ such that $\Delta_{\tilde{k}}^c \geq \epsilon_{\hat{\imath}+1}$. From the definition of the sequences $\{\epsilon_i\}$ and N_i we can derive the following inequality:

$$\Delta_k^c < \bar{\beta}^{i-\hat{\imath}} \Delta_{\tilde{k}}^c\ \forall k \geq N_i, i > \hat{\imath} + 1. \tag{4.15}$$

Now, we aim to compute a lower bound for $\Delta_k^c\ \forall k \geq N_{\hat{\imath}+1}$ and establish a relationship with Δ_k^z. Algorithm 4.1 states that the continuous trust-region radius can be modified in three particular circumstances:

1. Reduced as a consequence of the invocation of the **CriticalityTest** (Line 8). We define this reduction as $ct_k^{dec} = \frac{\Delta_k^c}{\Delta_k^{icbc}}$. If the **CriticalityTest** is not invoked $ct_k^{dec} = 1$.

2. Reduced as consequence of the *unsuccessful* iteration k, where model m_k is fully-linear (Line 31). The reduction is equivalent to $\Delta_{k+1}^c = \Delta_k^c \gamma_0 ct_{k+1}^{dec}$.

3. Extended as a consequence of the *very-successful* iteration k (Line 18). As $\Delta_k^c \leq \Delta_{max}^c / \gamma_1$ then $\Delta_{k+1}^c = \gamma_1 \Delta_k^c ct_{k+1}^{dec}$.

From points (1), (2) and (3) we rewrite Δ_k^c as:

$$\Delta_k^c = \left(\prod_{j=\tilde{k}}^{k} ct_j^{dec} \right) \Delta_{\tilde{k}}^c \gamma_0^{redu(k,\tilde{k})} \gamma_1^{ext(k,\tilde{k})}\ \forall k > \tilde{k} > \hat{k}, \tag{4.16}$$

where $ext(k, \tilde{k})$ is the number of iterations where Δ_k^c is extended, and $redu(k, \tilde{k})$ the number of iterations between $\tilde{k}, k$ where Δ_k^c is reduced. From Algorithm 4.1 we can compute an upper bound for Δ_k^z in terms of $redu(k, \tilde{k})$ and $ext(k, \tilde{k})$:

- If iteration k is *very-successful* then

$$\Delta_{k+1}^z = \min\{\Delta_{max}^z, \max\{\gamma_1 \Delta_k^z c t_{k+1}^{dec}, 1\}\} \leq \max\{\gamma_1 \Delta_k^z c t_{k+1}^{dec}, 1\}. \tag{4.17}$$

  As $\Delta_k^z > 1 \ \forall k > \hat{k}$ then $c t_{k+1}^{dec} \Delta_k^z \gamma_1 > 1$ and therefore we have that $\Delta_{k+1}^z \leq c t_{k+1}^{dec} \Delta_k^z \gamma_1$ as a result from 4.17.

- If iteration k is *unsuccessful* and model m_k is fully linear, then

$$\Delta_{k+1}^z = \max\{1, c t_{k+1}^{dec} \max\{\gamma_0 \Delta_k^z, 1\}\},$$

  Similarly to the previous case, the fact that $\Delta_k^z > 1 \ \forall k > \hat{k}$ implies that $\Delta_{k+1}^z = \gamma_0 \Delta_k^z c t_{k+1}^{dec}$.

Considering the above points we have that

$$\Delta_k^z \leq \left(\prod_{j=\tilde{k}}^{k} c t_j^{dec} \right) \Delta_{\tilde{k}}^z \gamma_0^{redu(k,\tilde{k})} \gamma_1^{ext(k,\tilde{k})} \ \ \forall k > \tilde{k} > \hat{k}. \tag{4.18}$$

Combining Equations (4.15), (4.16) and (4.18) we obtain the following relationship:

$$\begin{aligned} \frac{\Delta_k^z}{\Delta_{\tilde{k}}^z} &\leq \left(\prod_{j=\tilde{k}}^{k} c t_j^{dec} \right) \gamma_0^{redu(k,\tilde{k})} \gamma_1^{ext(k,\tilde{k})} \\ &= \frac{\Delta_k^c}{\Delta_{\tilde{k}}^c} < \bar{\beta}^{i-\hat{\imath}-1} \ \ \forall k \geq N_i, i > \hat{\imath} + 1. \end{aligned}$$

The latter would imply that $\lim_{k\to\infty} \Delta_k^z = 0$, which contradicts the initial assumption and concludes the proof. □

Theorem 4.7. *Under Assumptions 1.1, 1.2 and 1.3, the sequence of iterates generated by Algorithm 4.1 satisfies:*

$$\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$$
$$\text{and } \ f(x_k, y_k) \leq f(x, y) + \epsilon_a \quad \forall (x, y) \in \{x_k\} \times \Omega_d^Q(y_k),$$

where $\Omega_d^Q(y_k) = \bar{Z}^k$ is defined on Line 10, Algorithm 4.6.

Proof. This statement is proven by contradiction. Suppose there exists $\tilde{y} \in \Omega_d^Q(y_k)$ for which $f(x_k, y_k) - f(x_k, \tilde{y}) \geq \epsilon_a$, $k > k_{mis}$, $k \in \{j \in \mathbb{N} \mid \Delta_j^z = 1\}$ and model m_k is fully-linear. We highlight that as $k \to \infty$, then:

- $\Delta_k^c \to 0$.
- $|f(x, y) - f(x_k, y)| \approx 0 \ \forall x \in \hat{B}_c(x_k, \Delta_k^c), y \in \hat{B}_z(y_k, 1) \cap \Omega_z$ from Assumption 1.2.

We now show that the above remarks imply that $\tilde{y}$ or a solution in a different integer manifold among those in $\Omega_d^Q(y_k)$ must be identified in the **CandidateComputation** or **RescueProcedure**. Note that $m_k(x_k, \tilde{y}) = f(x_k, \tilde{y})$. Let $\tilde{x} = \text{argmin}_{x \in \hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c} m_k(x, \tilde{y})$. Then $m_k(x_k, y_k) - m_k(\tilde{x}, \tilde{y}) \geq m_k(x_k, y_k) - m_k(\tilde{x}, y_k) \geq \epsilon_a$ as $f(\tilde{x}, \tilde{y}) \approx m(\tilde{x}, \tilde{y}) \leq m(\tilde{x}, y_k) \approx f(\tilde{x}, y_k)$. Therefore any solution $(\bar{x}, \bar{y})$ of the surrogate optimization problem in Line 5 of Algorithm 4.3 must improve by at least ϵ_a. However, note that:

$$|m(x_k, y_k) - m(x, y_k)| = |(x - x_k)^\top l_k^c| \leq \|l_k^c\| \cdot \Delta_k^c = 0, \ \ \forall x \in \hat{B}_c(x_k, \Delta_k), \tag{4.19}$$

thus the improvement cannot be due to a change in the continuous variables. Then, the integer component of the iterate must change. If $\frac{f(x_k, y_k) - f(\hat{x}, \hat{y})}{m_k(x_k, y_k) - m_k(\hat{x}, \hat{y})} \geq \eta_0$, the new solution is accepted and the discrete components of the iterate change. If this condition is not met, the **RescueProcedure** is invoked. Because the surrogate model is accurate over $\Omega_d^G(y_k, M_k)$, then either: (i) the point $(x_k, \tilde{y})$ is selected as output of Algorithm 4.5, as it yields an improvement of at least ϵ_a on the objective, or (ii) a different point in the neighborhood is selected, with a larger improvement. In either case, the output (x', y') of the **RescueProcedure** cannot have the same integer components as y_k, because, due to Assumption 1.2, we have:

$$f(x_k, y_k) - f(x', y') \geq f(x_k, y_k) - f(x_k, \tilde{y}) \geq \epsilon_a \geq$$
$$|f(x_k, y_k) - f(x, y_k)| \approx 0 \ \ \forall x \in \hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c.$$

The above discussion shows that the integer components of the iterate necessarily change. By Theorem 4.5, this can only happen a finite a number of times. Thus, there cannot be an infinite sequence of iterates converging to (x_k, y_k), a contradiction. □

Theorem 4.8. *Under Assumptions 1.1, 1.3, 3.1 and 3.2, the sequence of iterates generated by Algorithm 4.1 satisfies:*

$$\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$$
$$\text{and } f(x_k, y_k) \leq f(x, y) + \epsilon_a \quad \forall (x, y) \in \{x_k\} \times \Omega_d^{LQMI}(y_k).$$

Proof. The proof follows the same structure as for Theorem 4.7; the only difference is that, due to Assumptions 3.1 and 3.2, the model m_k is accurate for all points $(x, y) \in \hat{B}_v(x_k, y_k, \Delta_k^c, 1)$, i.e., $m_k(x, y) \approx f_k(x, y)$. Thus, if a better integer candidate $\tilde{y}$ in the neighborhood $\hat{B}_z(y_k, 1)$ exists, **CandidateComputation** identifies a point that improves the objective function by at least ϵ_a. Since Δ_k^c is sufficiently small, the iteration is then *successful* or *very-successful*, and similarly to Theorem 4.7, this contradicts Theorem 4.5.

□

Now, having proved that Algorithm 4.1 is capable of converging to a ϵ_a mixed-integer separate local minimum (SLM), we will show that it convergences to a strong separate minimum (StLM). The radius of the region over which the point is a StLM is dependent on the ϵ_a parameter and the global Lipschitz κ_f constant of the function f.

Theorem 4.9. *If Assumptions 1.1, 1.2 and 1.3 hold, then*

$$\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$$
$$\text{and } f(x_k, y_k) \leq f(x, y) + 2\epsilon_a \quad \forall (x, y) \in (\hat{B}_c(x_k, \bar{\Delta}) \times \Omega_d) \cap \Omega_m$$

*where $\bar{\Delta} = \frac{\epsilon_a}{\kappa_f}$, for some discrete neighborhood Ω_d. The discrete neighborhood Ω_d is equal to $\Omega_d^{LQMI}(y_k)$ for an LQMI function, and it is the output of the **QuadraticInterpolationSet** procedure on Line 10 Algorithm 4.6 ($\Omega_d^{Q}(y_k)$) for functions that are not LQMI.*

Proof. For every point $(x, y) \in (\hat{B}_c(x_k, \bar{\Delta}) \times \Omega_d) \cap \Omega_m$ the following inequality holds, as a direct consequence of Theorems 4.7 and 4.8:

$$f(x_k, y_k) - f(x, y) = f(x_k, y_k) - f(x, y) - f(x_k, y) + f(x_k, y) \leq \\ \epsilon_a + f(x_k, y) - f(x, y).$$

Considering that $|f(x_k, y) - f(x, y)| \leq \kappa_f \|x_k - x\|$ and $\|x_k - x\| \leq \frac{\epsilon_a}{\kappa_f}$; then,

$$f(x_k, y_k) - f(x, y) \leq \epsilon_a + \kappa_f \frac{\epsilon_a}{\kappa_f} \leq 2\epsilon_a,$$

which concludes the proof. □

Remark 2. *We have proved algorithmic convergence to a strong separate local minimum on the sets $\Omega_d^Q(y_k)$ and $\Omega_d^{LQMI}(y_k)$. We have proved that the number of iterations for which $\Delta_k^z = 1$ is infinite; nonetheless, in particular cases there might exist an infinite number of iterations for which $\Delta_k^z > 2$. In consequence, for particular instances it is possible to achieve convergence to a StLM with respect to the larger discrete set $\Omega_d \subset \hat{B}_z(y_k, \Delta)$ with $\Delta \geq 2$.*

4.3 Conclusions and Future Work

In this chapter we introduced an LQMI-based algorithm for the solution of a box-constrained mixed-integer derivative-free problem. Algorithm 4.1 is an adaptation of the trust-region algorithm that uses the surrogate model framework introduced in Chapter 3. We presented new features that allow to overcome the problems that arise from dealing with a mixed-integer function:

- We introduced the mixed-integer stationarity parameter, Θ_k, that measures possible objective improvement in surrounding manifolds.
- We presented the combined stationarity parameter, Φ_k, that serves as an indicator of mixed-integer stationarity and triggers the convergence test.
- We adapted the *Criticality Step* to be used in the mixed-integer setting. It incorporates the **ManifoldSearch** procedure that allows to avoid early convergence to first-order stationary points.

- We introduced the **RescueProcedure** (Algorithm 4.5) that allows to use the *fully-linearity* of LQMI surrogates in the general mixed-integer setting.

Furthermore, we proved that Algorithm 4.1 is globally convergent to a separate local minimum (SLM) and a strong local minimum (StLM). In the future, we expect this methodology can be used as a base for further developments on mixed-integer derivative-free applications. We highlight this framework has the potential to be used to tackle problems with more complicated constraints, both algebraic and simulated-based (black-box). Adapting this framework to incorporate *penalty* [77], [80], the *augmented Lagrangian* function [91], [92] or progressive-barrier [39] is natural to deal with general non-convex constraints.

Chapter 5

Hybrid DCA-DFO Optimization

In this chapter, we present a methodology for solving the mixed-integer optimization problem 1.1 under Assumptions 1.1, 1.2 and the following combinatorial property with respect to the integer variables:

Assumption 5.1. *Function $f(x, y)$ is $M^\natural$ discrete convex with respect to the discrete variables y at fixed values of the vector of continuous variables x.*

In the next section, we introduce the generalities of M and $M^\natural$ discrete convex functions. Let us now recall some notions that will be used in the solution approach.

Definition 5.1. *For a convex function g the vector $w \in \mathbb{R}^{n_c}$ is said to be a subgradient of f at the point x_o if:*

$$g(x) \geq g(x_o) + w^\top (x - x_o), \ \ \forall x \in \boldsymbol{dom}_g.$$

The set of all the subgradients of g at x_o is called the subdifferential *and it represented by $\partial g(x_o)$.*

Definition 5.2. *For $\epsilon > 0$, a vector $w \in \mathbb{R}^{n_c}$ is said to be a* ϵ-subgradient *for the convex function g at the point x_o if:*

$$g(x) \geq g(x_o) - \epsilon + w^\top (x - x_o) \ \ \forall x \in \boldsymbol{dom}_g.$$

The set of all ϵ-subgradients of g at x_o is ϵ-subdiferential and is denoted by $\partial_\epsilon g(x_o)$.

We illustrate the difference between subgradients and ϵ-subgradients in Figure 5.1, that represents the function $f(x) = x^2$ and its first-order information. Figure 5.1a shows the derivative at the point $x_o = 0$. Note that no point on the graph lies above

the gradient line. We remark that as $f(x) = x^2$ is convex and differentiable, its subdifferential only contains one element, $\partial f(x_o) = \{\nabla_x f(x_o)\}$. Figures 5.1b and 5.1c show the linear interpolation of f between the points (x_o, x_1) and (x_o, x_2), respectively. Note that the line between the points $x_o, f(x_o)$ and $x_1, f(x_1)$ lies above $f(x)$ in the segment (x_o, x_1); the same happens with the linear interpolation between x_o and x_2. It indicates that the coefficients $\frac{f(x_1)-f(x_o)}{x_1-x_o} \notin \partial f(x_o)$ and $\frac{f(x_2)-f(x_o)}{x_2-x_o} \notin \partial f(x_o)$; however, it is evident that the slope of the linear interpolation is an inexact subgradient of $f(x_o)$ with error bounded by $\epsilon = \max_{x\in\mathbb{R}} \frac{f(x_1)-f(x_o)}{x_1-x_o}(x - x_o) - f(x)$. We highlight that the subgradient error of the linear interpolation of a convex function depends on how distant its samples are located. For example, the subgradient error in Figure 5.1b, ϵ_1, is equal to 0.25; on the other hand, the subgradient error in Figure 5.1c, ϵ_2, is equal to 1. Therefore, $\frac{f(x_1)-f(x_o)}{x_1-x_o} \in \partial_{\epsilon_2} f(x_o)$ but $\frac{f(x_2)-f(x_o)}{x_2-x_o} \notin \partial_{\epsilon_1} f(x_o)$.

Finally, we introduce the following notations:

Definition 5.3. *Let $I_y : \mathbb{Z} \to \mathbb{Z}^{n_z}$ be defined as the map of one integer index to one element of the discrete set Ω_z.*

Definition 5.4. *Let $\Gamma : \mathbb{R}^{n_c} \to \mathbb{R}$ be defined as*

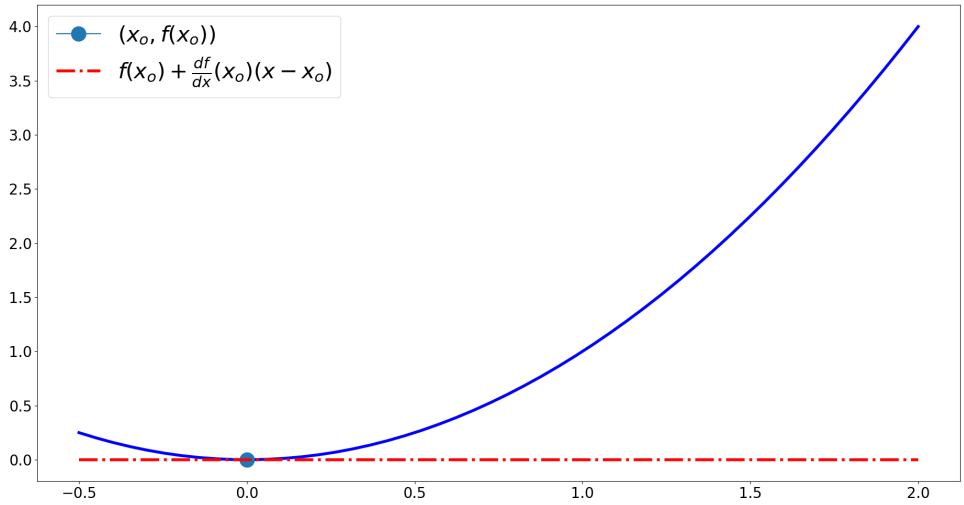
$$\Gamma(x) = \min_{y\in\Omega_z} f(x, y).$$

5.1 Discretely Convex Functions

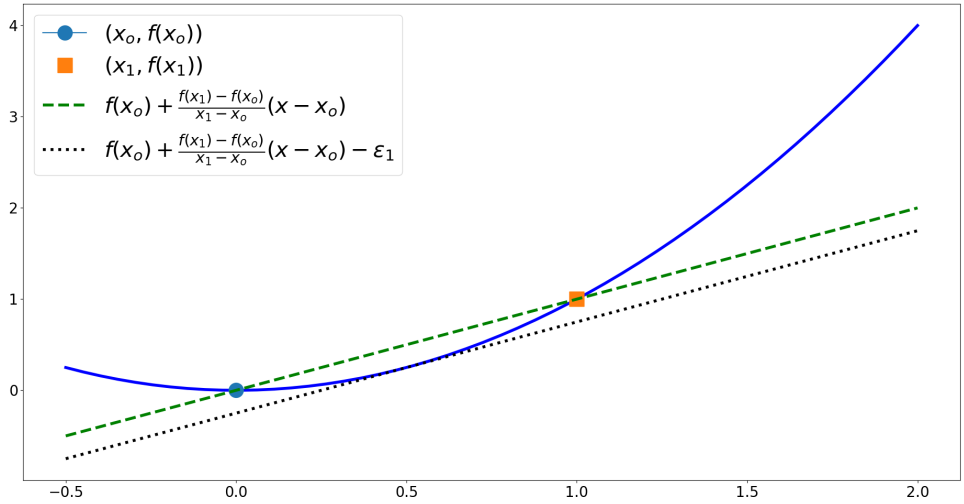
In this section we describe the M and $M^\natural$ discrete convex functions. Both are part of the discrete convex analysis [93], a theoretical framework that combines the concepts of convex analysis and combinatorial mathematics. Discrete convex analysis has been extensively used in the operations research as a tool for solving storage and inventory problems [94], [95]; moreover, it plays an important role in the design and analysis of auction algorithms [96]. The M and $M^\natural$ functions are defined as follows:

Definition 5.5. *A function $f : \mathbb{Z}^{n_z} \to \mathbb{R}$ with $\boldsymbol{dom}_f \neq \emptyset$ is said to be* M discrete convex function *if it accomplishes the following exchange axiom: for every pair $z^1, z^2 \in \boldsymbol{dom}_f$ and $u \in supp^+(z^2 - z^1)$ there exists a $v \in supp^-(z^2 - z^1)$ such that*

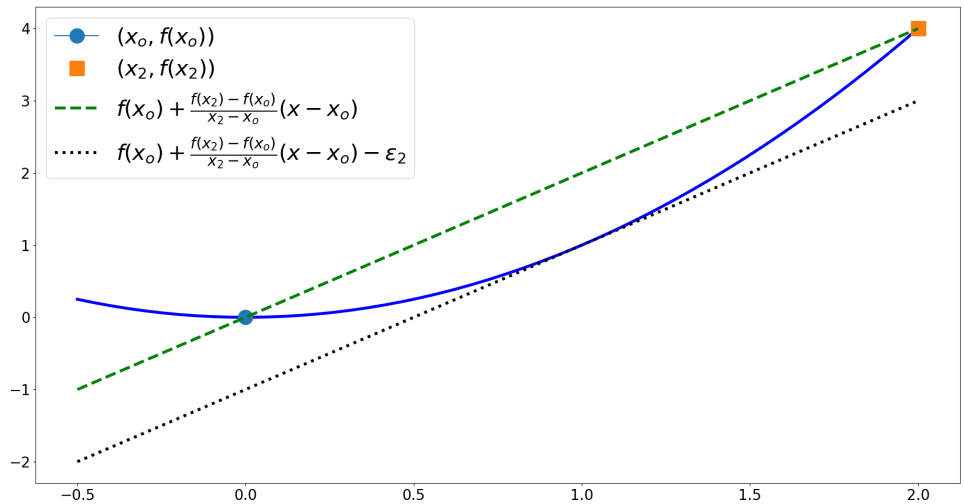
$$f(z^1) + f(z^2) \geq f(z^2 - e_u + e_v) + f(z^1 + e_u - e_v).$$



(A) Exact gradient at $x = 0$



(B) Linear interpolation between the points $x_o = 0$ and $x_1 = 1$



(C) Linear interpolation between the points $x_o = 0$ and $x_2 = 2$

FIGURE 5.1: Subgradients and ϵ-subgradients

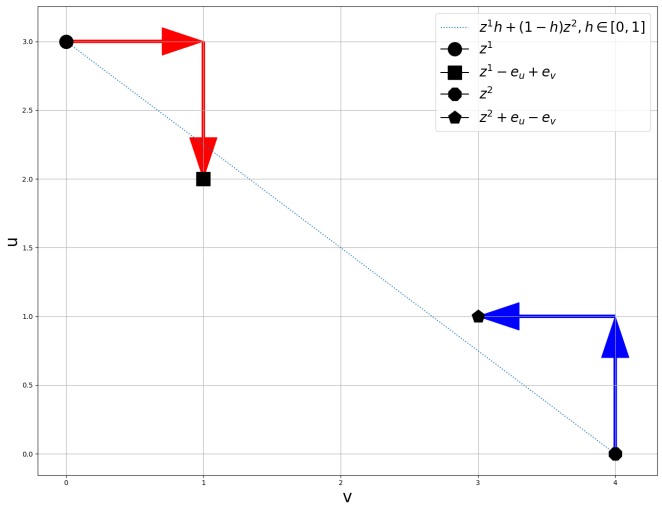


FIGURE 5.2: Exchange property in M convex functions

Figure 5.2 shows the exchange axiom in Definition 5.5. Note that the points $z^1 - e_v + e_u$ and $z^2 - e_u + e_v$ are the closest grid points to the line segment between z^1 and z^2. M convexity mimics the relationship

$$f(x^1) + f(x^2) \geq f(x^1 - h(x^1 - x^2)) + f(x^2 + h(x^1 - x^2)) \;\; \forall h \in [0,1]$$

that holds for every convex function $f : \mathbb{R}^{n_c} \to \mathbb{R}$.

Definition 5.6. *A function* $f : \mathbb{Z}^{n_z} \to \mathbb{R}$ *with* $\boldsymbol{dom}_f \neq \emptyset$ *is said to be* $M^\natural$ discrete convex function *if the function* $\tilde{f} : \mathbb{Z}^{n_z+1} \to \mathbb{R}$ *defined by:*

$$\tilde{f}(y_o, y) = \begin{cases} f(y) & \textbf{\textit{if}} \;\; y_o = -\sum_{j=1}^{n_z} y_j \\ +\infty & \textit{otherwise} \end{cases}$$

is M discrete convex.

Note that $M^\natural$-convex functions are defined in terms of M-convex functions. It is easy to prove that an M-convex function is a special case of the $M^\natural$ set of functions, for which its domain is restricted to the hyperplane $\{y \in \mathbb{Z}^{n_z} \mid \sum_{i=1}^{n_z} y_i = a\}, \;\; a \in \mathbb{Z}$. One of the most interesting features displayed by the the two classes of discrete functions is the equivalence of local and global optimality, similar to the case of the continuous convex functions:

Theorem 5.1. ***(Murota, [93] , Theorem 6.26)*** *Global and local optimality in descretly convex functions:*

1. *For an* M convex function *and* $z \in \boldsymbol{dom}_f$, *we have*

$$f(z) \leq f(y) \ \ (\forall y \in \mathbb{Z}^{n_z}) \iff f(z) \leq f(z + e_u - e_v) \ \ (\forall u, v \in \{1, \ldots, n_z\}).$$

2. *For an* $M^\natural$-convex function *and* $z \in \boldsymbol{dom}_f$, *we have*

$$f(z) \leq f(y) \ \ (\forall y \in \mathbb{Z}^{n_z}) \iff \begin{cases} f(z) \leq f(z + e_u - e_v) \ \ (\forall u, v \in \{1, \ldots, n_z\}) \\ f(z) \leq f(z \pm e_u) \ \ (\forall u \in \{1, \ldots, n_z\}). \end{cases}$$

We emphasize that the optimality criterion given by Theorem 5.1 only relies on samples, this condition is convenient in the integer black-box setting. These functions also present discrete versions of subgradients and subdifferentials, as well as descent directions. All of these features allow the development of tailored algorithms that converge to global minimizers. Such algorithms only consider zero-th order information. We now show two examples of said methodologies: the steepest descent algorithm (Algorithm 5.1) and the domain reduction algorithm (Algorithm 5.2).

Algorithm 5.1 Steepest Descent Algorithm

Input: Point $y \in \mathbf{dom}_f$
Output: Global minimizer y

loop
  Set $(\bar{u}, \bar{v}) = \operatorname{argmin}_{u,v \in \{1,\ldots,n_z\} | u \neq v} \{f(y + e_u - e_v)\}$
  if $f(y_k) \leq f(y + e_u - e_v)$ **then**
    break
  else
    Set $y = y + e_u - e_v$
  end if
end loop

We remark that both algorithms are designed to solve only M-convex functions; however, they are easily adapted to solve $M^\natural$-convex instances by transforming the

Algorithm 5.2 Domain Reduction Algorithm

Input: Variable bounds y_{lb} and y_{ub}
Output: Global minimizer y

Set $D = \mathbf{dom}_f$
loop
  For all $i \in \{1, \ldots, n_z\}$ set $lb_i = \min_{y \in D} e_i^\top y$ and $ub_i = \max_{y \in D} e_i^\top y$
  Set $\bar{lb} = \left\lfloor \left(1 - \frac{1}{n_z}\right) lb + \frac{1}{n_z} ub \right\rfloor$ and $\bar{ub} = \left\lceil \frac{1}{n_z} lb + \left(1 - \frac{1}{n_z}\right) ub \right\rceil$
  Set $\bar{D} = \{z \in \mathbb{Z}^{n_z} \mid \bar{lb} \leq z \leq \bar{ub}\}$
  Select a $y \in \bar{D}$
  Set $(\bar{u}, \bar{v}) = \operatorname{argmin}_{u,v \in \{1,\ldots,n_z\} | u \neq v} \{f(y + e_u - e_v)\}$
  if $f(y) \leq f(y + e_u - e_v)$ **then**
    break
  else
    Set $D = D \cap \{z \in \mathbb{Z}^{n_z} \mid z_u \leq x_u - 1, z_v \geq y_v + 1\}$
  end if
end loop

objective function into

$$\tilde{f}(y_o, y) = \begin{cases} f(y) & \textbf{if } y_o = -\sum_{j=1}^{n_z} y_j \\ +\infty & \text{otherwise.} \end{cases}$$

Note that the term y_o accounts for the univariate translations $y \pm e_i \ \forall i \in \{1, \ldots, n_z\}$. Let $K_f = \max\{\|y^1 - y^2\|_1 \mid y^1, y^2 \in \mathbf{dom}_f\}$ and $K_\infty = \max\{\|y^1 - y^2\| \mid y^1, y^2 \in \mathbf{dom}_f\}$. Algorithm 5.2 finds the global optima of an M-convex function f in $\mathcal{O}(n_z^4 (\log_2 K_\infty)^2)$ evaluations of f [93]. On the other hand, if a vector in the domain is given Algorithm 5.1 finds the global optima of an M-convex function f in $\mathcal{O}(n_z^2 K_f)$ function evaluations when a special tie-breaking rule is applied on f [97].

There exist additional efficient methodologies for the optimization of M-convex and $M^\natural$ functions based on the combination of the (scaled) steepest descend and domain reduction algorithm, for example Shioura's *fast-scaling algorithm* [98] that finds the global optima in $\mathcal{O}(n_z^3 log_2(K_\infty / n_z))$ function evaluations. In the next section, we will detail a methodology that takes advantage of the polynomial time algorithms for the solution of M-convex functions to optimize the mixed-integer function $f(x, y)$ over the set Ω_m.

5.2 Solution by the Difference of Convex Algorithm (DCA)

We aim to solve the problem 1.1 under Assumption 5.1 by performing the following reformulation:

$$\min_{x\in\Omega_c, y\in\Omega_z} f(x,y) = \min_{x\in\Omega_c} \Gamma(x).$$

Observation 5.1. *Function* $\Gamma(x)$ *is equivalent to* $\psi(x) = -\sup_{y\in\Omega_z} -f(x,y)$.

If Assumption 5.1 holds, the Observation 5.1 implies that evaluating Γ at a given point x is equivalent to maximizing an $M^{\natural}$-concave function. In addition, the function $\Gamma(x)$ is equivalent to the pointwise operation $\Gamma(x) = -\max_{i\in\{1,\dots,|\Omega_z|\}} -\hat{f}_{I_y(i)}(x)$ (see Definition 5.3).

We highlight that function $\Gamma(x)$ can be represented as the difference of two functions

$$\Gamma(x) = \frac{\lambda_k}{2}\|x\|^2 - \left[\frac{\lambda_k}{2}\|x\|^2 - \Gamma(x)\right],$$

where the term $\frac{\lambda_k}{2}\|x\|^2 - \Gamma(x)$ can be rewritten as

$$\begin{aligned}\frac{\lambda_k}{2}\|x\|^2 - \Gamma(x) &= \frac{\lambda_k}{2}\|x\|^2 - \left(-\max_{i\in\{1,\dots,|\Omega_z|\}} -\hat{f}_{I_y(i)}(x)\right)\\ &= \max_{i\in\{1,\dots,|\Omega_z|\}} \left(\frac{\lambda_k}{2}\|x\|^2 - \hat{f}_{I_y(i)}(x)\right).\end{aligned}$$

Moreover, if $\lambda_k > \kappa_g$ then $\frac{\lambda_k}{2}\|x\|^2 - \Gamma(x)$ is a convex function, as we now prove.

Definition 5.7. *Let* $\Psi_k(x) = \max_{i\in\{1,\dots,|\Omega_z|\}} \frac{\lambda_k}{2}\|x\|^2 - \hat{f}_{I_y(i)}(x)$.

Lemma 5.1. (Zhou, [99], Lemma 4.) *Let* $\hat{f}_y(x)$ *be differentiable with* κ_g-Lipschitz *gradient for every* $y \in \Omega_z$. *Then* $\frac{\kappa_g}{2}\|x\|^2 - \hat{f}_y(x)$ *is convex. It also means that* $\frac{\lambda_k}{2}\|x\|^2 - \hat{f}_y(x)$ *is convex if* $\lambda_k > \kappa_g$.

Theorem 5.2. *If Assumption 1.3 holds and* $\lambda_k > \kappa_g$, *then* $\Psi_k(x)$ *is a convex function.*

Proof. Having that Assumption 1.3 holds, every function $\frac{\lambda_k}{2}\|x\|^2 - \hat{f}_y(x)$ is convex. We recall that the pointwise maximum of convex functions is also convex [100]. □

Remark 3. $\Psi_k(x)$ *is a convex subdifferentiable function, with a subdifferential denoted by:*

$$\partial\Psi_k(x) = \boldsymbol{Co}\left\{\lambda_k x - \nabla_x \hat{f}_y(x) \mid \frac{\lambda_k}{2}\|x\|^2 - \hat{f}_y(x) = \Psi_k(x),\ \ y\in\Omega_z\right\}.$$

To summarize, Problem 1.1 can be rewritten as $\min_{x\in\Omega_c} \frac{\lambda_k}{2}\|x\| - \Psi_k(x)$. Theorem 5.2 and Remark 3 allow to view the function $\Gamma(x)$ as the difference of two convex functions, $\frac{\lambda_k}{2}\|x\|$ and $\Psi_k(x)$. We can therefore solve it using the Difference of Convex Algorithm (DCA) [101] (see Algorithm 5.3).

Algorithm 5.3 DCA method for problem 1.1

Input: Point $x_0 \in \Omega_c$, regularization term $\bar{\lambda} > \kappa_g$
1: Set $k = 0$
2: **repeat**
3: Set $\lambda_k = \bar{\lambda}$
4: Compute a subgradient $w_k \in \partial\Psi_k(x_k)$
5: Set $x_{k+1} = \operatorname{argmin}_{x\in\Omega_c}\{\frac{\lambda_k\|x\|}{2} - w_k^\top(x - x_k)\}$
6: **until** Convergence of x_k

There exist two main issues that arise from the use of Algorithm 5.3. First, we lack the first-order information to compute the subdifferential $\partial\Psi_k$. Second, we lack a proper estimation of the κ_g constant. The estimation of an accurate value for λ_k is therefore crucial: if the value of λ_k is too large it can affect the rate of convergence, making the algorithm slow. In contrast, if λ_k is small, there is a risk that $\Psi_k(x)$ is not convex. To overcome these issues we introduce elements from the surrogate-based derivative-free optimization to develop an algorithm with similar convergence properties to the DCA.

5.3 Hybrid DCA-DFO Algorithm

The standard DCA requires at each iteration the computation of $w_k \in \partial\Psi_k(x_k)$. However, this computation does not need to be exact. It is possible to overcome the lack of first-order information by considering accurate approximations of the subgradients (often called ϵ-subgradients) which can be estimated using fully-linear or fully-quadratic surrogates of Ψ_k. DC algorithms have been proved to be convergent also in the case where ϵ-subgradients are used [102], [103]. The only condition required is that $\lim_{k\to\infty}\epsilon_k = 0$.

Note that the computation of the ϵ-subgradients requires sampling inside the domain $\hat{B}_c(x_k, \Delta_k^c))$, that is a function of a given trust-region radius Δ_k^c. It adds at least one degree of freedom to Algorithm 5.3.

In this section, we introduce the hybrid DCA-DFO that incorporates three modifications from the DCA 5.3:

- We replace the subdifferential $\partial\Psi_k(x)$ with the ϵ_k subdifferential $\partial_{\epsilon_k}\Psi_k(x)$ computed from the fully-linear approximation $m_k(x)$ of $\hat{f}_{y_k}(x)$ on $\hat{B}_c(x_k, \Delta_k^c)$.
- We do not consider a fixed λ_k but one that is function of the model $m_k(x)$ and Δ_k^c. In this way we eliminate the additional degrees of freedom that come from the sampling and the model construction process.
- We relax the optimality conditions of $\Psi_k(x)$. We show that is possible to retain convergence considering a suboptimal y_k such $y_k \notin \operatorname{argmax}_{y\in\Omega_z} -f(x_k, y)$.

5.3.1 Algorithm Description

In this subsection we show how the terms λ_k can be related to Δ_k^c under the principles of first-order derivative-free trust-region methods. The following theorem explores the properties of the DCA outer-linearization subproblem (Line 5, Algorithm 5.3):

Theorem 5.3. *Let m_k be a given surrogate approximation of $\hat{f}_{y_k}(x)$ on $\hat{B}_c(x_k, \Delta_k^c)$. Let $x' = \operatorname{argmin}_{x\in\Omega_c}\left\{\frac{\lambda_k\|x\|^2}{2} - (\lambda_k x_k - \nabla_x m_k(x_k))^\top (x - x_k)\right\}$. For every iteration $k > 0$ the following items hold:*

1. $\|x' - x_k\| \leq \frac{\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|}{\lambda_k}$.
2. $x' = p_k(x_k, \nabla_x m_k(x_k), 1/\lambda_k)$, *and*
3. *if* $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| > 0$, *then* $\|x' - x_k\| > 0$ *and* $|(x' - x_k)^\top \nabla_x m_k(x_k)| > 0$.

Proof. The first-order stationarity conditions related to this optimization problem indicate that

$$u^+ - u^- + \lambda_k x' - \lambda_k x_k + \nabla_k m_k(x_k) = 0,$$

where $u^+, u^- \in \mathbb{R}^{n_c}$ are the dual variables related to the box constraints $x \leq x_{ub}$ and $x_{lb} \leq x$, respectively. In addition, we have that $u^+, u^- \geq 0$, $(u^+)^\top u^- = 0$, $(u^+)^\top (x' - x_{ub}) = 0$, $(u^-)^\top (x_{lb} - x') = 0$ and

$$x_i' - x_{k,i} = \frac{1}{\lambda_k}\left(u_i^- - u_i^+ - \frac{\partial m_k(x_k)}{\partial x_i}\right).$$

We have three possible scenarios, depending whether the variable $x_{k,i}$ lays on a corner of Ω_c:

1. For the set of variables such $x_{k,i} = x_{ub,i}$ we have that $x'_i - x_{k,i} \leq 0$, then:

   - If $\frac{\partial m_k(x_k)}{\partial x_i} < 0$ we that $u_i^- = 0$ and $u_i^+ > 0$, as a consequence $x'_i = x_{ub,i}$ and $x'_i - x_{k,i} = 0$.
   - If $\frac{\partial m_k(x_k)}{\partial x_i} > 0$ we have that if $x'_i = x_{lb,i}$ then $u_i^- \geq 0$; otherwise, $u_i^- = 0$. As a consequence
   $$x'_i = \max\left\{x_{lb_i}, x_k - \lambda_k \frac{\partial m_k(x_k)}{\partial x_i}\right\} \tag{5.1}$$
   and
   $$|x'_i - x_{k,i}| = \min\left\{x_{ub,i} - x_{lb,i}, \frac{1}{\lambda_k}\frac{\partial m_k(x_k)}{\partial x_i}\right\} \leq \frac{1}{\lambda_k}\frac{\partial m_k(x_k)}{\partial x_i}.$$
   - If $\frac{\partial m_k(x_k)}{\partial x_i} = 0$ then $x'_i = x_{ub,i}$.

   We highlight in this case that if $\frac{\partial m_k(x_k)}{\partial x_i} < 0$, then $\mathbf{red}(\nabla_x m_k(x_k), x_k)_i = 0$, otherwise $\mathbf{red}(\nabla_x m_k(x_k), x_k)_i = \frac{\partial m_k(x_k)}{\partial x_i}$. Therefore $|x'_i - x_{k,i}| \leq \frac{1}{\lambda_k}\mathbf{red}(\nabla_x m_k(x_k), x_k)_i$.It is also evident that if $\frac{\partial m_k(x_k)}{\partial x_i} > 0$, then $\frac{\partial m_k(x_k)}{\partial x_i}(x'_i - x_{k,i}) < 0$; otherwise $\frac{\partial m_k(x_k)}{\partial x_i}(x'_i - x_{k,i}) = 0$.

2. For the set of variables such $x_{k,i} = x_{lb,i}$ we can similarly prove that $|x'_i - x_{k,i}| \leq \frac{1}{\lambda_k}|\mathbf{red}(\nabla_x m_k(x_k), x_k)_i|$ with
   $$x'_i = \min\left\{x_{ub_i}, x_k - \lambda_k \min\left\{0, \frac{\partial m_k(x_k)}{\partial x_i}\right\}\right\} \tag{5.2}$$
   and
   $$|x'_i - x_{k,i}| = \min\left\{x_{ub,i} - x_{lb,i}, -\frac{1}{\lambda_k}\min\left\{0, \frac{\partial m_k(x_k)}{\partial x_i}\right\}\right\}.$$
   We can derive as well that if $\frac{\partial m_k(x_k)}{\partial x_i} > 0$, then $\frac{\partial m_k(x_k)}{\partial x_i}(x'_i - x_{k,i}) < 0$; otherwise $\frac{\partial m_k(x_k)}{\partial x_i}(x'_i - x_{k,i}) = 0$.

3. For the set of variables such $x_{lb,i} < x_{k,i} < x_{ub,i}$ we have that

- If $\frac{\partial m_k(x_k)}{\partial x_i} > 0$, then

$$x'_i = \max\left\{x_{lb,i}, x_{k,i} - \frac{1}{\lambda_k}\frac{\partial m_k(x_k)}{\partial x_i}\right\} \tag{5.3}$$

and

$$|x'_i - x_{k,i}| = \min\left\{x_{k,i} - x_{lb,i}, \frac{1}{\lambda_k}\frac{\partial m_k(x_k)}{\partial x_i}\right\} \leq \frac{1}{\lambda_k}\frac{\partial m_k(x_k)}{\partial x_i}.$$

- If $\frac{\partial m_k(x_k)}{\partial x_i} \leq 0$, then

$$x'_i = \min\left\{x_{ub,i}, x_k - \frac{1}{\lambda_k}\frac{\partial m_k(x_k)}{\partial x_i}\right\} \tag{5.4}$$

and

$$|x'_i - x_{k,i}| = \min\left\{x_{ub,i} - x_{k,i}, -\frac{1}{\lambda_k}\frac{\partial m_k(x_k)}{\partial x_i}\right\} \leq -\frac{1}{\lambda_k}\frac{\partial m_k(x_k)}{\partial x_i}.$$

- If $\frac{\partial m_k(x_k)}{\partial x_i} = 0$, then $x'_i = 0$.

We remark in this case that $\mathbf{red}(\nabla_x m_k(x_k), x_k)_i = \frac{\partial m_k(x_k)}{\partial x_i}$, thus $|x'_i - x_{k,i}| \leq \frac{1}{\lambda_k}|\mathbf{red}(\nabla_x m_k(x_k), x_k)_i|$. Furthermore, if $\left|\frac{\partial m_k(x_k)}{\partial x_i}\right| > 0$ then $\frac{\partial m_k(x_k)}{\partial x_i}(x'_i - x_{k,i}) < 0$; otherwise $\frac{\partial m_k(x_k)}{\partial x_i}(x'_i - x_{k,i}) = 0$.

As a result we have that $\|x_k - x'\| \leq \|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|/\lambda_k$.
Finally, the chain of Equations 5.1, 5.2, 5.3 and 5.4 indicates that $x' = [P_{\Omega_c}(x_k - \frac{1}{\lambda_k}\nabla_x m_k(x_k)] = p_k(x_k, \nabla_x m(x_k), 1/\lambda_k)$. It also shows that for every coordinate $i \in \{1, \ldots, n_c\}$ such that $0 < |\mathbf{red}(\nabla_x m_k(x_k), x_k)_i|$ the value of $|x'_i - x_{k,i}| > 0$. Hence, if $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|$ is bounded away from 0 the same happens to $\|x' - x_k\|$ and $|\nabla_x m_k(x_k)^\top (x' - x_k)|$, concluding the proof. □

Theorem 5.3 shows that the meaningful search domain for iteration k is determined by the ratio between $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|$ and λ_k. It is therefore natural to define $\lambda_k = \frac{\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|}{\Delta_k^c}$. Under this principle we construct our algorithm.

The hybrid DCA-DFO algorithm is presented in Algorithm 5.4. For a given iteration k the current solution is denoted by (x_k, y_k). Y^k indicates the set of points

sampled up to the iteration k. The basic idea is to compute at every iteration a continuous surrogate $m_k(x)$ that approximates the manifold $\hat{f}_{y_k}(x)$ and use its gradient (evaluated at the point x_k) as a substitute of the subdiferential $\partial\Psi_k(x_k)$.

The inputs of this algorithm are: the black-box function $f(x,y)$, the variable lower bounds (x_{lb}, y_{lb}) and upper bounds (x_{ub}, y_{ub}), the initial candidate solution $(x_0, y_0) \in \Omega_m$, the starting and maximum size of the trust-region Δ_0^{icb} and Δ_{max}, the parameters used to update the trust-region γ_0 and γ_1, the parameters used to evaluate the quality of an iteration η_0 and η_1, the scaling parameters of criticality test μ and β, the reduction factor of the trust-region ω and the pivoting tolerance $\xi \in (0, 1/4]$. Finally, we consider the parameter PO that determines if the function $\Psi_k(x)$ is computed to optimality, or we use a suboptimal solution such that $y' \notin \text{argmax}_{y\in\Omega_z} \frac{\lambda_k}{2}\|x_k\| - f(x_k, y)$.

The initialization (Lines 1-2) correspond to the computation of the first surrogate approximation m_k^{icb}. Note that this model is constructed on the integer coordinate y_0; nonetheless, a preliminary integer local search can be performed to accelerate the rate of objective improvement without affecting algorithmic convergence. The main part of the algorithm consists of a loop (Lines 3-42) that is repeated until a stopping criterion is reached. The loop starts verifying if the norm of the reduced gradient $\|\mathbf{red}(\nabla_x m_k^{icb}(x_k), x_k)\|$ of the incumbent model is smaller than threshold ϵ_c (Line 4), implying convergence to a first-order stationary point. In this case, the *Criticallity Step* (Lines 6-12) is invoked and the variable $CRIT$ is set to 1. The **CriticalityTest** evaluates at the same time if the incumbent model is fully-linear on $\hat{B}_c(x_k, \Delta_k^{icb}) \cap \Omega_c$ and if there exists certain relationship between the reduced gradient $\mathbf{red}(\nabla_x m_k^{icb}(x_k), x_k)$ and the continuous trust-region radius. If m_k^{icb} does not accomplish both conditions, the **CriticalityTest** (Algorithm 4.2) is used to generate a new model for which $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| \geq \Delta_k^c \mu^{-1}$. However, if the model m_k^{icb} is fully-linear and the desired relationship is satisfied, the model m_k^{icb} is accepted (Line 10) and subsequently used to compute a new candidate solution.

After the *Criticality Step* a new candidate solution (x', y') is computed together with the update parameter ρ_k. This part of the algorithm is called *Candidate Computation* (Lines 16-24), that consists in the following operations: first, the regularization term λ_k and the subgradient w_k are computed from the reduced gradient and the

trust-region radius Δ_k^c. Next, we compute x' by optimizing the outer linearization of $\Gamma(x)$ (Line 18). The candidate y' is obtained via local search heuristics when $PO = 0$ and $CRIT = 0$, or by using one of the global optimization procedures tailored for the optimization of M and $M^\natural$-convex functions ($CRIT = 1$).

If $\rho_k \in (\eta_0, \eta_1)$ iteration k is said to be *successful*, when the new solution yields a sufficiently large improvement with respect to the previous solution. Furthermore, if $\rho_k > \eta_1$ the iteration k is said to be *very-successful*. If we have a *successful* or *very-successful* iteration (i.e., $\rho_k > \eta_0$), we accept (x', y') as the current solution and we generate the model to be used in the next iteration (Line 32). If we have a *very-successful* iteration (i.e., $\rho_k \geq \eta_1$), we increase the size of the trust-region (Line 23).

In case $\rho_k \leq \eta_0$ we check if the model used is fully-linear (Line 27). If the model m_k is fully-linear the iteration k is called *unsuccessful* and the trust-region radius Δ_k^c is reduced. Note that the model is not modified, thus $m_{k+1}^{icb}(x) = m_k(x)$. On the other hand, if the fully-linearity condition is not satisfied, we have a *model-improving* iteration, where the **LinearInterpolationModel** procedure is used to construct a fully-linear surrogate on $\hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c$.

In the following subsection we detail the **CriticalityTest** and **LinearInterpolationModel** procedures used in Algorithm 5.4.

5.3.2 Auxiliary Procedures

The most important task in the development of Algorithm 5.4 is the construction of continuous-fully linear surrogates to compute approximate subgradients. This is done via linear-interpolation. The process to select the proper samples to retrieve a linear model is detailed in Algorithm 5.5 - **LinearInterpolationModel**. This algorithm is based on the LU decomposition and aims to generate a set of linearly independent samples inside the trust-region $(\hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c) \times \{y_k\}$. It takes into account the set of previous samples (Lines 2 and 3) and selects the fittest n_c points. In case the existing set is not *well-poised* or the number of samples is insufficient this algorithm is able to select a new set of points by maximizing the absolute value of the pivotal polynomials $u_i(x)$ inside the domain. Note that the pivotal polynomials are linear, therefore, the optimal solution of the subproblem in Line 7 is computed without large computational effort using a non-linear solver.

Algorithm 5.4 Hybrid sequential DFO-DCA Algorithm

Input: Black-box function $f(x,y)$, variable bounds (x_{lb}, y_{lb}) and (x_{ub}, y_{ub}), initial point $(x_0, y_0) \in \Omega_m$. Initial and maximum trust-region radii Δ_0^{icb} and Δ_{max}, γ_0, γ_1 such that $0 < \gamma_0 < 1 < \gamma_1$; η_0, η_1 such that $0 < \eta_0 < \eta_1 < 1$; μ, β such that $\mu > \beta > 0$; $\omega \in (0,1)$; $\xi \in (0, 0.25]$;$\epsilon_c > 0$ and $PO = \{0, 1\}$.

1: Set $k = 0$
2: Set $(m_0^{icb}, Y^0) =$ **LinearInterpolationModel**$(x_0, x_{lb}, x_{ub}, y_0, \emptyset, \Delta_0^{icb}), \xi)$
3: **repeat**
4: **if** $\|$**red**$(\nabla_x m_k^{icb}(x_k), x_k)\| < \epsilon_c$ **then**
5: Set $CRIT = 1$
6: **if** m_k^{icb} is not fully-linear or $\|$**red**$(\nabla_x m_k(x_k), x_k)\| < \mu^{-1}\Delta_k^{icb}$ **then**
7: $(m_k, Y^k\tilde{\Delta}_k) =$ **CriticalityTest**$(x_k, x_{lb}, x_{ub}, y_k, \omega, \mu, Y^k, \xi)$
8: Set $\Delta_k = \min\{\max\{\tilde{\Delta}_k, \beta\|$**red**$(\nabla_x m_k(x_k), x_k)\|\}, \Delta_k^{icb}\}$
9: **else**
10: GoTo line 18
11: **end if**
12: **else**
13: $CRIT = 0$
14: Set $m_k = m_k^{icb}, \Delta_k^c = \Delta_k^{icb}$
15: **end if**
16: Set $\lambda_k = \|$**red**$(\nabla_x m_k(x_k), x_k)\| / \Delta_k^c$
17: Set $w_k = \lambda_k x_k - \nabla_x m_k(x_k)$
18: Set $x' = \text{argmin}_{x \in \Omega_c}\{\frac{\lambda_k}{2}\|x\|^2 - (x - x_k)^\top w_k\}$
19: **if** $PO = 1$ or $CRIT = 1$ **then**
20: Set $y' = \text{argmin}_{y \in \Omega_z} f(x', y)$
21: **else**
22: Compute y' by partially optimizing $f(x', y)$ in Ω_z
23: **end if**
24: Set $\rho_k = \frac{f(x_k, y_k) - f(x', y')}{(x' - x_k)^\top \nabla_x m_k(x_k)}$
25: **if** $\rho_k > \eta_0$ **then**
26: Set $x_{k+1} = x'$, $y_{k+1} = y'$
27: **if** $\rho > \eta_1$ **then**
28: Set $\Delta_{k+1}^{icb} = \min\{\gamma^{inc}\Delta_k, \Delta_{max}\}$
29: **else**
30: Set $\Delta_{k+1}^{icb} = \Delta_k^c$
31: **end if**
32: Set $(m_{k+1}^{icb}, Y^{k+1}) =$ **LinearInterpolationModel**$(x_{k+1}, x_{lb}, x_{ub}, y_{k+1}, Y^k\Delta_{k+1}^{icb}), \xi)$
33: **else**
34: **if** m_k is fully-linear with respect $(\hat{B}_c(x_k, \Delta_k) \cap \Omega_c) \times \{y_k\}$ **then**
35: Set $\Delta_{k+1}^{icb} = \gamma\Delta_k$, $m_{k+1}^{icb} = m_k$ and $Y^{k+1} = Y^k$
36: **else**
37: Set $(m_{k+1}^{icb}, Y^{k+1}) =$ **LinearInterpolationModel**$(x_k, x_{lb}, x_{ub}, y_k, Y^k, \Delta_k^c, \xi)$
38: **end if**
39: Set $x_{k+1} = x_k$, $y_{k+1} = y_k$
40: **end if**
41: Set $k = k + 1$
42: **until** Convergence is proven

These second auxiliary procedure of Algorithm 5.4 is the **CriticalityTest** (Algo-

Algorithm 5.5 LinearInterpolationModel

Input: Current continuous solution x_k, variable bounds x_{lb}, x_{ub}, current integer solution y_k, set of samples $\mathcal{Y}_k$, trust-region radius Δ_k^c and pivoting tolerance ξ
Output: Fully-linear model $m_k(x)$ and updated set of samples $\mathcal{Y}^k$

Set $u_i(x) = x_i \ \ \forall i \in \{1, \ldots, n_c\}$
Set $X_k = \{x \in \hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c \mid (x, y_k) \in \mathcal{Y}^k\}, \tilde{X}_k = \{\}$
Set $\hat{X}_k = \{(x - x_k)/\Delta_k^c \ \ \forall x \in X_k\}$
for $i \in \{1, \ldots, n_c\}$ **do**
  Set $\bar{x} = \text{argmax}_{x \in \hat{X}_k} |u_i(x)|$
  if $|u_i(\bar{x})| < \xi$ or $\hat{X}_k = \emptyset$ **then**
    Set $\bar{x} = \text{argmax}_{x \in \hat{B}_c(0,1) | x_k + x\Delta_k^c \in \Omega_c} |u_i(x)|$
    Set $\mathcal{Y}^k = \mathcal{Y}^k \cup \{x_k + \bar{x}\Delta_k^c\}$
  else
    Set $\hat{X}_k = \hat{X}_k \setminus \{\bar{x}\}$
  end if
  Set $\tilde{X}_k = \tilde{X}_k \cup \{\bar{x}\}$
  for $j \in \{i+1, \ldots, n_c\}$ **do**
    Set $u_j(x) = u_j(x) - \frac{u_j(\bar{x})}{u_i(\bar{x})} u_j(x)$
  end for
  Solve the system of equations $f(x_k + \bar{x}\Delta_k^c, y_k) - f(x_k, y_k) = g^\top \bar{x}\Delta_k^c \ \ \forall \bar{x} \in \tilde{X}_k$ to compute $g \in \mathbb{R}^{n_c}$
  Set $m_k(x) = f(x_k, y_k) + g^\top (x - x_k)$
end for

rithm 5.6). It is used to evaluate the convergence into a first-order stationary point. The goal of Algorithm 5.6 is to generate a surrogate approximation $\tilde{m}_k$ of $f(x_k, y_k)$ such that the following conditions are satisfied: the model m_k is fully-linear and $\|\textbf{red}(\nabla_x m_k(x_k), x_k)\| \geq \Delta_k^c \mu^{-1}$. We highlight that if $\|\textbf{red}(\nabla_x f(x_k, y_k), x_k)\| > 0$ this procedure converges in a finite number of iterations [66].

Algorithm 5.6 CriticalityTest

Input: Current continuous solution x_k, variable bounds x_{lb}, x_{ub}, current integer solution y_k, parameters ω, μ,set of samples $\mathcal{Y}^k$ and pivoting tolerance ξ
Output: Model m_k, updated set of samples $\mathcal{Y}^k$ and trust-region radius $\tilde{\Delta}_k$

Set $i = 1$
repeat
  Set $\tilde{\Delta}_k = \omega^{i-1} \Delta_k^{icb}$
  Set $(\tilde{m}_k, \mathcal{Y}^k) =$ **LinearInterpolationModel**$(x_k, x_{lb}, x_{ub}, y_k, \mathcal{Y}^k, \tilde{\Delta}_k, \xi)$
  Set $i = i + 1$
until $\tilde{\Delta}_k < \mu \|\textbf{red}(\nabla_x \tilde{m}_k(x_k), x_k)\|$

5.4 Convergence of Algorithm 5.4

In this section we prove that Algorithm 5.4 is globally convergent to a first-order stationary point, in a similar way to the scheme presented by Conn et al [66]. We use several elements introduced in Section 4.2, such as the function $\chi_k(x_k, g, t)$ and the *criticality measure* $\bar{\chi_k} = \chi_k(x_k, \nabla_x m_k(x_k), 1)$. The proof consists in the following steps. First, we detail the relationship between the reduced gradient $\mathbf{red}(\nabla_x m_k(x_k), x_k)$ and the solution of the outer linearization problem described in Line 18-Algorithm 5.4. Then, we present the notion of *minimum decrease condition* which is theoretically equivalent to the improvement related to *generalized Cauchy point* on the trust-region method. Next, we show that unless a point is stationary, model update procedures compute a model for which the gradient $\mathbf{red}(\nabla_x m_k(x_k), x_k)$ and the trust-region radius Δ_k^c diverge from zero, and that objective improvement is always possible. Finally, we prove by contradiction that the sequence $\{x_k, y_k\}$ is convergent, and its limiting value $(\tilde{x}, \tilde{y})$ is first-order stationary with respect to the continuous variables and a global optima with respect to the integer set Ω_z. For the remainder of this section we denote $x' = \operatorname{argmin}_{x \in \Omega_c} \left\{ \frac{\lambda_k \|x\|^2}{2} - (\lambda_k x_k - \nabla_x m_k(x_k))^\top (x - x_k) \right\}$ and $\mathcal{S}_{imp}$ as the collection of all the *successful* and *very-successful* iterations.

Now, we introduce the building blocks of this proof. First, we remark that there exists a pair of global constants κ_{ef}, κ_{eg} that bound the error of the interpolation for every iteration for which m_k is fully-linear:

Proposition 5.1. *For any given function f that satisfies Assumption 1.3 and the class of discrete sets $\bar{\Omega} = \{ \{y\} \mid \forall y \in \Omega_z \}$, we guarantee there exists a fully-linear class of models $\mathcal{M}$ (Definition 3.2) with suitable positive global constants κ_{ef}, κ_{eg} such that, for any given $\Delta \in (0, \Delta_{max}]$, $(x, y) \in \Omega_m$ and $\Omega_d \in \bar{\Omega}$, the error in the function and gradient approximation is bounded. Moreover, we can obtain a fully-linear model from this class in a finite, uniformly bounded operations and function evaluations.*

Algorithm 5.4 considers $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|$ as a measure of algorithmic stationarity with respect to the set Ω_c. The following two corollaries summarize the relationship of x', x_k and Δ_k^c with the reduced gradient:

Corollary 5.1. *Let $x_k \in \Omega_c$. If $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| > 0$ there exists a global constant $\kappa_{frd} \in (0, 1)$ such that $\|x' - x_k\| \geq \kappa_{frd} \Delta_k^c$.*

Proof. Since $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| > 0$, then $\Delta_k^c = \frac{\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|}{\lambda_k} \geq \|x' - x_k\| > 0$ by Theorem 5.3, then the ratio between $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|$ and Δ_k^c must be bounded from bellow

$$0 < \kappa_{frd} \leq \frac{\|x' - x_k\|}{\Delta_k^c} \leq 1.$$

□

Corollary 5.2. *Let $x_k \in \Omega_c$. If $\|\boldsymbol{red}(\nabla_x m_k(x_k), x_k)\| > 0$ there exists a global constant $\kappa_{cri} \in (0,1)$ such that $\bar{\chi}_k \geq \kappa_{cri} \|\boldsymbol{red}(\nabla_x m_k(x_k), x_k)\|$.*

Proof. Since $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| > 0$, then $\|x' - x_k\| > 0$ by Theorem 5.3. As $x_k, x' \in \Omega_c$, any point on the line $x_k + (x' - x_k)h \in \Omega_c, \ \forall h \in [0,1]$. As a consequence, the point $x_k + (x' - x_k)/\max\{2, \|x' - x_k\|\} \in \hat{B}_c(x_k, 1) \cap \Omega_c$, thus

$$\bar{\chi}_k \geq \frac{|(x' - x_k)^\top \nabla_x m_k(x_k)|}{\max\{2, \|x' - x_k\|\}} > 0.$$

Finally, from Lemma 4.2 we have $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| \geq \bar{\chi}_k$, therefore

$$0 < \kappa_{cri} \leq \frac{\bar{\chi}_k}{\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|} < 1.$$

□

Next, we show that if the current iterate is not a first-order critical point, Algorithm 5.6 converges in a finite number of iterations:

Lemma 5.2. *If $\|\boldsymbol{red}(\nabla_x f(x_k, y_k), x_k)\| > 0$ then the **CriticalityTest** (Algorithm 5.6) terminates in a finite number of iterations.*

Proof. This proof is equal to the proof of Lemma 4.5. □

Lemma 5.2 implies that unless $\mathbf{red}(\nabla_x f(x_k, y_k), x_k) = 0, \ \|\mathbf{red}(\nabla_x m_k(x_k) x_k)| > 0$ and it is possible to attain objective improvement by exploring the current manifold y_k. With this result, we present the minimum decrease condition for Algorithm 5.4.

Lemma 5.3. *At every iteration $k \geq 0$ there exists a minimum decrease condition given by*

$$(x_k - x')^\top \nabla_x m_k(x_k) \geq \kappa_{frd} \bar{\chi}_k \min\{\Delta_k^c, 1\}.$$

Proof. From Theorems 4.1 and 5.3 we have that $x' = p_k(x_k, \nabla_x m_k(x_k), \lambda_k)$ and

$$\chi_k(x_k, \nabla_x m_k(x_k), \|x' - x_k\|) = |(x' - x_k)^\top \nabla_x m_k(x_k)|.$$

First we consider the case $\|x' - x_k\| \geq 1$. From Lemma 4.1, $\chi_k(x_k, \nabla_x m_k(x_k), \|x' - x_k\|) \geq \bar{\chi}_k$, thus $(x_k - x')^\top \nabla_x m_k(x_k) \geq \bar{\chi}_k \geq \kappa_{frd}\bar{\chi}_k$. Next, we consider the case $\|x' - x_k\| < 1$. From Lemma 4.1 we have $\chi_k(x_k, \nabla_x m_k(x_k), \|x' - x_k\|) \geq \|x' - x_k\|\bar{\chi}_k$. Recalling Corollary 5.1, we then obtain:

$$(x_k - x')^\top \nabla_x m_k(x_k) = \chi_k(x_k, \nabla_x m_k(x_k), \|x' - x_k\|) \geq \bar{\chi}_k \|x' - x_k\| \geq \kappa_{frd}\bar{\chi}_k \Delta_k^c.$$

It completes the proof. □

Now, we relate the minimum decrease condition given by Lemma 5.3 with the convergence of Algorithm 5.4:

Lemma 5.4. *If the model m_k is fully-linear and the trust-region radius satisfies*

$$\Delta_k^c \leq \min\left\{1, \frac{\kappa_{frd}\|\boldsymbol{red}(\nabla_x m_k(x_k), x_k)\|(1-\eta_1)}{\kappa_{ef}}\right\}$$

then iteration k is very-successful.

Proof. First, consider the minimum decrease condition described in Lemma 5.3. As $\Delta_k^c \leq 1$ we have that $(x_k - x')^\top \nabla_x m_k(x_k) = m_k(x_k) - m_k(x') \geq \kappa_{frd}\bar{\chi}_k \Delta_k^c$. For every iteration $k > 0$ we can describe the update parameter as the addition of two separate terms ρ_k^1 and ρ_k^2. Let $\rho_k^1 = \frac{f(x_k,y_k)-f(x',y_k)}{m_k(x_k)-m_k(x')}$, $\rho_k^2 = \frac{f(x',y_k)-f(x',y')}{m_k(x_k)-m_k(x')}$ and

$$\rho_k = \frac{f(x_k, y_k) - f(x', y')}{m_k(x_k) - m_k(x')} = \frac{f(x_k, y_k) - f(x', y_k) + (f(x', y_k) - f(x', y'))}{m_k(x_k) - m_k(x')} = \rho_k^1 + \rho_k^2.$$

The term ρ_k^1 is a measure of objective improvement related to the local continuous search on the current integer manifold y_k. On the other hand, the term ρ_k^2 is a measure of possible objective decrease by the (partial) optimization in integer domain.

It is evident that $\rho_k^2 \geq 0$ as $f(x', y') \leq f(x', y_k)$. Now, taking into account the fully-linearity of the model m_k we establish the bounds of ρ_k^1:

$$\rho_k^1 - 1 = \frac{f(x_k, y_k) - m_k(x_k) + (m_k(x') - f(x', y_k))}{m_k(x_k) - m_k(x')}$$

with $f(x_k, y_k) - m_k(x_k) = 0$ from model interpolation and $\|m_k(x') - f(x', y_k)\| \leq \kappa_{ef}(\Delta_k^c)^2$ from the fully-linear error bounds, then

$$|\rho_k^1 - 1| \leq \frac{\kappa_{ef}(\Delta_k^c)^2}{\kappa_{frd}\bar{\chi}_k\Delta_k^c} \leq \frac{\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|(1-\eta_1)}{\bar{\chi}_k}.$$

From Lemma 4.2 we have that $\bar{\chi}_k \leq \|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|$ thus $|\rho_k^1 - 1| \leq (1 - \eta_1)$. Therefore, $\rho_k^1 \geq \eta_1$ and $\rho_k \geq \eta_1$, and iteration k is *very-successful*. □

Lemma 5.5. $\lim_{k\to\infty} \Delta_k^c = 0$.

Proof. For any *successful* or *very-successful* iteration we have that

$$f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \eta_0(x_k - x')^\top \nabla_x m_k(x_k) \geq \eta_0\kappa_{frd}\bar{\chi}_k \min\{\Delta_k^c, 1\}.$$

Recalling the conditions given by the **CriticalityTest** procedure this bound becomes

$$f(x_k, y_k) - f(x_{k+1}, y_{k+1}) \geq \eta_0(x_k - x')^\top \nabla_x m_k(x_k) \geq$$
$$\eta_0\kappa_{frd}\kappa_{cri}\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| \min\{\Delta_k^c, 1\}.$$

From Assumption 1.1 the term

$$\lim_{k\to\infty} f(x_0, y_0) - f(x_k, y_k) = \sum_{k \in \mathcal{S}_{imp}} f(x_k, y_k) - f(x_{k+1}, y_{k+1})$$

is bounded, condition that is only met if $\lim_{k\to\infty} \Delta_k^c = 0$. □

Lemma 5.6. $\liminf_{k\to\infty} \bar{\chi}_k = 0$.

Proof. Assume, for establishing a contradiction that there exists a $\kappa_1 > 0$ such that $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| \geq \bar{\chi}_k > \kappa_1, \forall k \geq 0$. We use the *Criticality Step* to derive a relationship between κ_1, $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|$ and Δ_k^c. There exists two possible scenarios:

- $\Delta_k^c \geq \min\{\Delta_k^{icb}, \beta\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|\}$ if the *CriticalityTest* is called.
- $\Delta_k^c = \Delta_k^{icb}$ otherwise.

By Lemma 5.4 and the assumption that $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| > \kappa_1$, whenever Δ_k^c falls bellow $\bar{\kappa}_2 = \min\left\{1, \frac{\kappa_{frd}\ \kappa_1(1-\eta_1)}{\kappa_{ef}}\right\}$, the iteration kth cannot be *unsuccessful*. Thus $\Delta_{k+1}^{icb} \geq \Delta_k$ and $\Delta_k^{icb} \geq \min\{\gamma_0\bar{\kappa}_2, \Delta_0^{icb}\}\ \ \forall k > 0$.
As $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| > \kappa_1$ we have that for every iteration k, whether the **CriticalityTest** is invoked or not, the following condition holds:

$$\Delta_k^c \geq \min\{\Delta_k^{icb}, \beta\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| \geq \min\{\Delta_k^{icb}, \beta\kappa_1\}.$$

As a result, Δ_k^c must be bounded by the factor $\Delta_k^c \geq \min\{\Delta_0^{icbc}, \beta\kappa_1, \gamma_0\bar{\kappa}_2\},\ \ \forall k > 0$, which contradicts Lemma 5.5.

□

Lemma 5.7. *For a subsequence $\{k_i\}$ such that*

$$\lim_{i\to\infty}\|\boldsymbol{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\| = 0$$

it also holds that

$$\lim_{i\to\infty}\|\boldsymbol{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| = 0$$
$$\textit{and}\quad \lim_{i\to\infty}\chi_{k_i}(x_{k_i}, \nabla_x f(x_{k_i}, y_{k_i}), 1) = 0.$$

Proof. First, note that for a large k_i we have that $\|\mathbf{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\| < \epsilon_c$ as the limit value of the subsquence is 0; thus, the model m_{k_i} is mixed-integer fully-linear and $\Delta_{k_i}^c \leq \mu\|\mathbf{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\|$ (Algorithm 5.4, *Criticality Step*). From Lemma 4.4 we have that

$$\begin{aligned}\|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i}) - \mathbf{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\| &\leq \|\nabla_x f(x_{k_i}, y_{k_i}) - \nabla_x m_{k_i}(x_{k_i})\| \\ &\leq \kappa_{eg}\Delta_{k_i}^c \leq \kappa_{eg}\mu\|\mathbf{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\|.\end{aligned}$$

This bound can be used to compute an upper bound on the norm of the reduced gradient of f at the point x_{k_i}, y_{k_i}:

$$\begin{aligned} &\|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| \\ \leq \|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i}) - \mathbf{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\| &+ \|\mathbf{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\| \\ &\leq (\kappa_{eg}\mu + 1)\|\mathbf{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\|. \end{aligned} \tag{5.5}$$

As a consequence $\lim_{i\to\infty}\|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| = 0$.
Finally, from Lemma 4.2 $\|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| \geq \chi_k(x_{k_i}, \nabla_x f(x_{k_i}, y_{k_i}), 1)$, thus,

$$\lim_{i\to\infty} \chi_{k_i}(x_{k_i}\nabla_x f(x_{k_i}, y_{k_i}), 1) = 0.$$

□

We now prove that there exists at least one accumulation point (x^*, y^*) which is stationary with respect to the continuous variables, or $\|\mathbf{red}(\nabla_x f(x^*, y^*), x^*)\| = 0$.

Lemma 5.8. $\liminf_{k\to\infty}\|\boldsymbol{red}(\nabla_x f(x_k, y_k), x_k)\| = 0$.

Proof. Assume, for establishing a contradiction there exists a bound $\kappa_1 > 0$ on the reduced gradient such that $\epsilon_c > \kappa_1$, $\|\mathbf{red}(\nabla_x f(x_k, y_k), x_k)\| > \kappa_1 \ \forall k \geq 0$. Now we establish a relationship between κ_1 and $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|$. From Lemma 5.6, there exists an iteration k_i such that $\bar{\chi}_k \leq \|\mathbf{red}(\nabla_x(m_{k_i}(x_{k_i}), x_{k_i})\| \leq \frac{\kappa_1}{2+\kappa_{eg}\mu}$. Considering that κ_1 is strictly smaller than ϵ_c, the model m_{k_i} is fully-linear and the chain of inequalities (5.5) holds. In consequence for iteration k_i we have that:

$$\kappa_1 \leq \|\mathbf{red}(\nabla_x f(x_{k_i}, y_{k_i}), x_{k_i})\| \leq (1 + \mu\kappa_{eg})\|\mathbf{red}(\nabla_x m_{k_i}(x_{k_i}), x_{k_i})\| < \frac{1 + \mu\kappa_{eg}}{2 + \mu\kappa_{eg}}\kappa_1,$$

which contradicts the initial assumption. □

In the remainder of this chapter we show that limiting values of x_k, y_k are stationary, or $\lim_{k\to\infty}\chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$. To complete the proof we use the potential convexity of the reformulation $\Psi_k(x)$. We remark that Algorithm 5.4 may generate a sequence of iterates $y_k \notin \operatorname{argmin}_{y\in\Omega_z} f(x_k, y)$. In those cases we can still proof convexity and generate valid inexact subgradients, as we now show:

Lemma 5.9. *Let* $m_k : \mathbb{R}^{n_c} \to \mathbb{R}$ *be a* fully-linear *approximation of* $\hat{f}_{y_k}(x)$ *on* $\hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c$, *where* $y_k \in \operatorname{argmax}_{y \in \Omega_z}\{-f(x_k, y)\}$. *The vector* $\lambda_k x_k - \nabla_x m_k(x_k) \in \partial_{\epsilon_k} \Psi_k(x_k)$ *with* $\epsilon_k = \kappa_{eg}(\Delta_k^c)^2$ *and* $\lambda_k \geq \kappa_g$.

Proof. For all $x \in \Omega_c$ we have that:

$$\Psi_k(x) \geq \Psi_k(x_k) + (x - x_k)^\top (\lambda_k x_k - \nabla_x \hat{f}_{y_k}(x_k))$$

which is equivalent to

$$\Psi_k(x) - \Psi_k(x_k) + (x - x_k)^\top (\nabla_x \hat{f}_{y_k}(x_k) - \nabla_x m_k(x_k)) \geq (x - x_k)^\top (\lambda_k x_k - \nabla_x m_k(x_k)).$$

As $\|\nabla_x m_k(x_k) - \hat{f}_{y_k}(x_k)\| \leq \kappa_{eg}\Delta_k$ and $\|x - x_k\| \leq \Delta_k^c$ then

$$(x - x_k)^\top (\nabla_x \hat{f}_{y_k}(x_k) - \nabla_x m_k(x_k)) \leq \|x - x_k\| \kappa_{eg}\Delta_k \leq \epsilon_k$$

and

$$\Psi_k(x) - \Psi_k(x_k) + \epsilon_k \geq (x - x_k)^\top (\lambda x_k - \nabla_x m_k(x_k)).$$

□

Lemma 5.10. *Let* $\tilde{\Psi}_k(x) = \frac{\lambda_k}{2}\|x\|^2 - \hat{f}_{y_k}(x)$ *with* $y_k \notin \operatorname{argmax}_{y \in \Omega_z}\{-f(x_k, y)\}$. *Let* $m_k : \mathbb{R}^{n_c} \to \mathbb{R}$ *be a* fully-linear *approximation of* $\hat{f}_{y_k}(x)$ *on* $\hat{B}_c(x_k, \Delta_k^c) \cap \Omega_c$. *The vector* $\lambda_k x_k - \nabla_x m_k(x_k) \in \partial_{\epsilon_k} \tilde{\Psi}_k(x_k)$ *with* $\epsilon_k = \kappa_{eg}(\Delta_k^c)^2$ *and* $\lambda_k \geq \kappa_g$.

Proof. As the parameter λ_k is larger than κ_g, the function $\tilde{\Psi}_k(x)$ is convex and differentiable, then for all $x \in \Omega_c$ we have that:

$$\tilde{\Psi}_k(x) \geq \tilde{\Psi}_k(x_k) + (x - x_k)^\top (\lambda_k x_k - \nabla_x \hat{f}_{y_k}(x_k))$$

which is equivalent to

$$\tilde{\Psi}_k(x) - \tilde{\Psi}_k(x_k) + (x - x_k)^\top (\nabla_x \hat{f}_{y_k}(x_k) - \nabla_x m_k(x_k)) \geq (x - x_k)^\top (\lambda_k x_k - \nabla_x m_k(x_k)).$$

As $\|\nabla_x m_k(x_k) - \hat{f}_{y_k}(x_k)\| \leq \kappa_{eg}\Delta_k$ and $\|x - x_k\| \leq \Delta_k^c$ then

$$(x - x_k)^\top (\nabla_x \hat{f}_{y_k}(x_k) - \nabla_x m_k(x_k)) \leq \|x - x_k\| \kappa_{eg}\Delta_k \leq \epsilon_k$$

and

$$\tilde{\Psi}_k(x) - \tilde{\Psi}_k(x_k) + \epsilon_k \geq (x - x_k)^\top (\lambda x_k - \nabla_x m_k(x_k)).$$

□

Theorem 5.4. *Under Assumptions 1.1 and 1.3 the sequence of iterates generated by Algorithm 5.4 satisfies:*

$$\lim_{k\to\infty} \|\textbf{\textit{red}}(\nabla_x m_k(x_k), x_k)\| = 0$$

and

$$\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$$

Proof. Assume, for establishing a contradiction there exist an infinite sequence of iterations $\{\ell_j\}$ such $0 < \epsilon_o < \chi_{\ell_j}(x_{\ell_j}, \nabla_x f(x_{\ell_j}, y_{\ell_j}), 1) \leq \|\textbf{red}(\nabla_x f(x_{\ell_j}, y_{\ell_j}), x_{\ell_j})\|$. From Lemma 5.7 we obtain that $\|\textbf{red}(\nabla_x m_{\ell_j}(x_{\ell_j}), x_{\ell_j})\| \geq \epsilon$ for some $\epsilon > 0$.
We now focus on the limiting value $\ell_j \to \infty$. Lemma 5.5 indicates that a subsequence of elements for which $\Delta_{\ell_j} \leq \min\left\{1, \frac{\epsilon}{2\kappa_g}, \frac{\gamma\kappa_{frd}\epsilon(1-\eta_1)}{\kappa_{ef}}\right\}$ exists. Note that as $\Delta_{\ell_j} \leq \epsilon/(2\kappa_g)$ the regularization parameter $\lambda_k > 2\kappa_g$. It indicates that:

- If $y_{\ell_j} \in \operatorname{argmin}_{y\in\Omega_z}\{f(x_{\ell_j}, y)\}$ the function Ψ_{ℓ_j} is convex (Lemma 5.1) and the vector $\lambda_{\ell_j} x_{\ell_j} - \nabla_x m_{\ell_j}(x_{\ell_j})$ computed from a fully-linear approximation $m_{\ell_j}(x)$ on $\hat{B}_c(x_{\ell_j}, \Delta_k^c) \cap \Omega_c$ is a proper subgradient of Ψ_{ℓ_j}, with error constant equal to $\kappa_{eg}(\Delta_{\ell_j}^c)^2$ (Lemma 5.9).
- If the current $y_{\ell_j} \notin \operatorname{argmin}_{y\in\Omega_z}\{f(x_{\ell_j}, y)\}$ the function $\tilde{\Psi}_k(x) = \frac{\lambda_{\ell_j}}{2}\|x\| - \hat{f}_{\ell_j}(x)$ is convex (Lemma 5.1) and the vector $\lambda_{\ell_j} x_{\ell_j} - \nabla_x m_{\ell_j}(x_{\ell_j})$ computed from a fully-linear approximation $m_{\ell_j}(x)$ on $\hat{B}_c(x_{\ell_j}, \Delta_k^c) \cap \Omega_c$ is a proper subgradient of $\tilde{\Psi}_{\ell_j}$, with error bounded by $\kappa_{eg}(\Delta_{\ell_j}^c)^2$ (Lemma 5.10).

We identify two possible scenarios depending on whether the model $m_{\ell_j}(x)$ is fully-linear or not. For the first case in which the model $m_{\ell_j}(x)$ is fully-linear on $\hat{B}_c(x_{\ell_j}, \Delta_{\ell_j}^c)$ we have that the iteration ℓ_j is *very-successful* (Lemma 5.4) and the following inequality holds (Lemma 5.9):

$$\frac{\lambda_{\ell_j}}{2}\|x_{\ell_j+1}\|^2 - f(x_{\ell_j+1}, y_{\ell_j+1}) \geq$$
$$\frac{\lambda_{\ell_j}}{2}\|x_{\ell_j}\|^2 - f(x_{\ell_j}, y_{\ell_j}) + (\lambda_{\ell_j} x_{\ell_j} - \nabla_x m_{\ell_j}(x_{\ell_j}))^\top (x_{\ell_j+1} - x_{\ell_j}) - \kappa_{eg}(\Delta_{\ell_j}^c)^2$$

that is equal to

$$\frac{\lambda_{\ell_j}}{2}\|x_{\ell_j+1}-x_{\ell_j}\|^2+(f(x_{\ell_j},y_{\ell_j})-f(x_{\ell_j+1},y_{\ell_j+1}))+\kappa_{eg}(\Delta^c_{\ell_j})^2 \geq (\nabla_x m_{\ell_j}(x_{\ell_j}))^\top(x_{\ell_j}-x_{\ell_j+1}). \tag{5.6}$$

We then define a proper upper bound for the left hand side of inequality 5.6. We have that $\|x_{\ell_j+1}-x_{\ell_j}\| \leq \Delta^c_{\ell_j}$ and $f(x_{\ell_j},y_{\ell_j})-f(x_{\ell_j+1},y_{\ell_j+1}) > \eta_1(\nabla_x m_{\ell_j}(x_{\ell_j}))^\top(x_{\ell_j}-x_{\ell_j+1})$, then

$$\frac{\lambda_{\ell_j}}{2}(\Delta^c_{\ell_j})^2+\eta_1(\nabla_x m_{\ell_j}(x_{\ell_j}))^\top(x_{\ell_j}-x_{\ell_j+1})+\kappa_{eg}(\Delta^c_{\ell_j})^2 \geq (\nabla_x m_{\ell_j}(x_{\ell_j}))^\top(x_{\ell_j}-x_{\ell_j+1}),$$

and reorganizing the previous inequality we obtain

$$\frac{\lambda_{\ell_j}}{2}(\Delta^c_{\ell_j})^2+\kappa_{eg}(\Delta^c_{\ell_j})^2 \geq (1-\eta_1)(\nabla_x m_{\ell_j}(x_{\ell_j}))^\top(x_k-x_{\ell_j+1}). \tag{5.7}$$

Taking into account that the minimum decrease condition given by 5.3 and the relationship that exist between $\bar{\chi}_k$ and $\|\mathbf{red}(\nabla_x m_k(x_k),x_k)\|$ (Corollary 5.2), we compute a proper lower bound for the right hand side of inequality 5.7:

$$\frac{\lambda_{\ell_j}}{2}(\Delta^c_{\ell_j})^2+\kappa_{eg}(\Delta^c_{\ell_j})^2 \geq (1-\eta_1)\kappa_{frd}\bar{\chi}_{\ell_j}\Delta^c_{\ell_j} \geq (1-\eta_1)\kappa_{frd}\kappa_{cri}\epsilon\Delta^c_{\ell_j}$$

thus

$$\Delta^c_{\ell_j} \geq \frac{(1-\eta_1)\kappa_{frd}\kappa_{cri}\epsilon}{\frac{\lambda_{\ell_j}}{2}+\kappa_{eg}} \geq \frac{(1-\eta_1)\kappa_{frd}\kappa_{cri}\epsilon}{2\kappa_g}.$$

For the second case consider that the model $m_{\ell_j}(x)$ is not fully-linear on $\hat{B}_c(x_{\ell_j},\Delta^c_{\ell_j})$. This case is only occurs in Algorithm 5.4 when the previous iteration was *unsuccessful* and $\|\mathbf{red}(\nabla_x m_{\ell_j}(x_{\ell_j}),x_{\ell_j})\| > \epsilon_c$. These two conditions indicate that the **CriticalityStep** was not summoned and the model was not updated from the previous iteration, thus $\Delta^c_{\ell_j} = \gamma\Delta^c_{\ell_j-1}$, $m_{\ell_j}(x) = m_{\ell_j-1}(x)$, $\|\mathbf{red}(\nabla_x m_{\ell_j-1}(x_{\ell_j-1}),x_{\ell_j-1})\| = \|\mathbf{red}(\nabla_x m_{\ell_j}(x_{\ell_j}),x_{\ell_j})\| \geq \epsilon$, and that the model $m_{\ell_j-1}(x)$ is fully-linear on $\hat{B}_c(x_{\ell_j-1},\Delta^c_{\ell_j-1})$. Nonetheless, this condition induces a contradiction. Since $\Delta^c_{\ell_j} \leq \frac{\gamma\kappa_{frd}\epsilon(1-\eta_1)}{\kappa_{ef}}$ we have that $\Delta^c_{\ell_j-1} \leq \frac{\kappa_{frd}\epsilon(1-\eta_1)}{\kappa_{ef}}$; therefore, iteration ℓ_j-1 must be *very-successful* (Lemma 5.4).

As a consequence we have that all the limiting values of the sequence $\{\ell_j\}$ are fully-linear and *very-successful*, then

$$\liminf_{\ell_j \to \infty} \Delta^c_{\ell_j} = \frac{(1-\eta_1)\kappa_{frd}\kappa_{cri}\epsilon}{2\kappa_g}. \tag{5.8}$$

This result contradicts Lemma 5.5. Therefore, for any given $\epsilon_0 > 0$ there exists an $N \in \mathbb{N}$ such that:

$$\chi_k(x_k, \nabla_x f(x_k, y_k), 1) < \epsilon_o \ \ \forall k > N.$$

It implies that $\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$. Moreover, Equation 5.8 indicates that $\lim_{k\to\infty} \|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| = 0$ as $\lim_{k\to\infty} \Delta^c_k = 0$. It concludes the proof. □

Theorem 5.5. *Under Assumptions 1.1 and 1.3 the sequence of iterates generated by Algorithm 5.4 satisfies:*

$$\liminf k \to \infty \lambda_k = \frac{1}{\mu}$$

Proof. From Theorem 5.4 we know that there exists a $N \in \mathbb{N}_+$ such that for every iteration $k > N$ the bound $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| < \epsilon_c$ holds. As a result, for the set of iterations such $k > N$ the relationship $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| \geq \Delta^c_k/\mu$ also holds. We remark that in Algorithm 5.4 the parameter $\lambda_k = \frac{\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|}{\Delta^c_k}$; therefore, as $k \to \infty$ the value of λ_k is bounded by the term $1/\mu$. □

Finally, having proved that Algorithm 5.4 is convergent to a first-order stationary point with respect to the vector x and the box Ω_c, we prove that the limiting values $\{y_k\}$ are global optima with respect to the discrete set Ω_z:

Theorem 5.6. *Under Assumptions 1.1, 1.2, 1.3 and 5.1 the sequence of iterates generated by Algorithm 5.4 satisfies:*

$$\lim_{k\to\infty} \chi_k(x_k, \nabla_x f(x_k, y_k), 1) = 0$$
$$\text{and} \ \ f(x_k, y_k) \leq f(x, y) \quad \forall (x, y) \in \{x_k\} \times \Omega_z.$$

Proof. From Theorem 5.4 we know that there exists a $N \in \mathbb{N}_+$ such that for every iteration $k > N$ the bound $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\| < \epsilon_c$ holds. As a result we have that the parameter $CRIT$ is set to 1 and the integer subproblem is solved to global

optimality (Line 19 - Algorithm 5.4). It implies that $y_k \in \operatorname{argmin}_{y \in \Omega_z} f(x_k, y) \ \forall k > N$. As a consequence we have that the limiting values of the sequence x_k, y_k are a SLM of the function f with respect to the discrete set Ω_z. □

Remark 4. *We have proved algorithmic convergence to a separate local minimum on the set Ω_z when the function exhibits M or $M^\natural$ properties. Moreover, the same results can be extended to mixed-integer functions that present combinatorial properties that allow a polynomial time solution of the partial minimization with respect to the set of discrete variables.*

Remark 5. *We highlight that Algorithm 5.4 can be used for the optimization of general mixed-integer functions that does not present combinatorial properties. In these cases, if the Assumptions 1.1, 1.2 and 1.3 hold we guarantee first-order stationarity with respect to the continuous variables, and weak local optimality with respect to the integer variables. The strength of the output of Algorithm 5.4 would depend on the method used for the optimization on the discrete domain.*

5.5 Conclusions and Future Work

In this chapter we introduced a hybrid difference of convex and trust-region algorithm for the solution of Problem 1.1 under the assumption of a combinatorial structure on the objective function. The solution scheme consists in the reformulation of the objective via partial discrete optimization and convex regularization with the weighted norm function. This method solves the problems that arise from the lack of first-order information by using error-bounded (sub)gradients. We proved that Algorithm 5.4 is globally convergent to strong local minimum (StLM) and can be used to address general mixed-integer derivative free problems.

In the future we expect this methodology can be used to address different types of combinatorial structures such as the globally convex functions or the L and $L^\natural$ discretly convex functions described by Murota [93], [104]. We also hope that our results motivate the research on additional structures on a mixed-integer function, like non-linear least squares, composite (non) convex functions and in particular general min-max problems.

Chapter 6

Methodology Benchmarking

In this chapter, we evaluate the computational performance of Algorithm 4.1 and Algorithm 5.4 for the optimization of structured instances -LQMI functions and $M^\natural$ discrete convex functions- and the solution of general mixed-integer black-box problems. To assess the rate of convergence and the quality of the solution obtained, we compare our results with the ones obtained with PyNomad-Beta. PyNomad-Beta is the Python interface of the solver NOMAD [38], the implementation of the MADS algorithm for the solution of black-box problems with continuous, integer and categorical variables.

To compare the different solution approaches we use *performance* and *data* profiles [42]. Let $\mathcal{S}$ be the set of solvers (or algorithms) that we want to evaluate and $\mathcal{P}$ the set of instances we want to use as a benchmark. A solver $s \in \mathcal{S}$ is said to be convergent for instance $p \in \mathcal{P}$ if it obtains the best objective improvement with respect to a starting evaluation point. More formally, let (x_0, y_0) be the starting evaluation point, (x_s, y_s) be the best solution found by solver s and $f_B = \min_{s\in\mathcal{S}} f(x_s, y_s)$ the best objective value for problem p obtained by all the solvers in $\mathcal{S}$. We have that the solver s is convergent up to a tolerance $\sigma \in (0, 1)$ if $f(x_o, y_o) - f(x_s, y_s) \geq (1 - \sigma)(f(x_o, y_o) - f_B)$.

The *performance ratio* $r^\sigma_{s,p}$ of a solver s with respect to the instance p is $r^\sigma_{s,p} = \frac{t^\sigma_{s,p}}{\min_{\hat{s}\in\mathcal{S}}\{t^\sigma_{\hat{s},p}\}}$, where $t_{s,p}$ is the total number of objective evaluations necessary to solver s to be convergent up to a tolerance σ for problem p. If a solver is not convergent, the performance ratio is considered as $r^\sigma_{s,p} = \infty$. The *performance* profile $per^\sigma_s(a)$ of a

solver s is a function of a required value a:

$$per_s^{\sigma}(a) = \frac{1}{|\mathcal{P}|}\left|\{p \in \mathcal{P} \mid \ r_{s,p}^{\sigma} \leq a\}\right|$$

For a given solver s, $per_s^{\sigma}(a)$ gives the fraction of convergent instances (up to a tolerance σ) with a performance ration lower than a. Performance profiles do not show how well a solver behaves in the case when the number of allowed function evaluations is limited. Therefore, we additionally use the *data* profile method. The data profile $dat_s(a)$ of a solver s is a function of a required value a and is defined as:

$$dat_s^{\sigma}(a) = \frac{1}{|\mathcal{P}|}\left|\{p \in \mathcal{P} \mid \ t_{s,p}^{\sigma}/(n_p+1) \leq a\}\right|$$

where $n_p = n_{c,p} + n_{z,p}$ is the total number of variables of problem p. For a given solver s, $dat_s^{\sigma}(a)$ represents the fraction of convergent instances with a number of normalized objective evaluation $t_{p,s}^{\sigma}/(n_p+1)$ lower than a.

6.1 Computational Results for Algorithm 4.1

In this section, we present the study of the computational performance of Algorithm 4.1. We evaluate two algorithmic versions that are described in Table 6.2: s_1 or *accurate*, and s_2 or *aggressive*.

6.1.1 Experimental Setting

We implemented Algorithm 4.1 in Python, using Pyomo [105], [106] to prototype the models used in Algorithms 4.3 and 4.5. The computational implementation also includes a co-domain re-scaling subroutine to prevent numerical issues related to large differences in the objective function. We follow a procedure similar to the ones used in [78], [107]: if the difference between the set of active samples is greater than 10^6 then the value $f(x_i, y_i)$ is replaced by $log(f(x_i, y_i))$ if its smallest value $f_{min} > 1$, or, $log(f(x_i, y_i) + 1 + |f_{min}|)$ otherwise. All the tests instances were evaluated in a single machine Lyon/hercule at GRID5000 cluster, with a Intel Xeon E5-2620 CPU and 32 GB of RAM. Finally, we consider algorithm termination when $\Delta_k^c < 10^{-5}$, $\Delta_k^z = 1$ and $\|\mathbf{red}(l_k^c, x_k)\| < \epsilon_c$, with $k_{max} = 700$.

TABLE 6.1: Fixed parameters for Algorithm 4.1

η_o	η_1	γ_0	γ_1	ϵ_c, ϵ_a	ω	β	μ
0.15	0.75	0.75	1.3	$1e^{-6}$	0.6	10^3	1500

$\Delta^c_{max} = \lceil \min_{i\in\{1,\ldots,n_c\}}\{x_{ub,i} - x_{lb,i}\}/2 \rceil$
$\Delta^c_o = \max\{\Delta^c_{max}/2, 0.1\}$
$\Delta^z_{max} = \lceil \min_{i\in\{1,\ldots,n_z\}}\{y_{ub,i} - y_{lb,i}\}/2 \rceil$
$\Delta^z_o = \max\{\Delta^z_{max}, 1\}$

TABLE 6.2: Variable parameters for Algorithm 4.1

	c_{f_r}	**IC**	**UP**	**SH**
s1-*accurate*	1	T	F	F
s2-*aggresive*	0.5	F	T	T

In Table 6.1 we present the values used for the fixed parameters. In Table 6.2 we resume the values of the variable parameters (c_{f_r}, IC, UP and SH) associated to the two variants of Algorithm 4.1. The variant *s1-accurate* uses at each iteration a fully-linear approximation ($c_{fr} = 1$ and IC=*TRUE*) and does not use the search heuristic (*SH=FALSE*), this allows to obtain a precise (but expensive to compute) model. On the other hand, the variant *s2-aggressive* aims at obtaining a model with a limited number of new sampled point, this is achieved by aggressively reusing previously generated point (*UP=TRUE*) and by activating the search heuristic (*SH=TRUE*). Our algorithm implementation uses the following **ManifoldSearch** procedure (Algorithm 6.1) in combination with the **CriticalityTest** (Algorithm 4.2) to avoid early convergence to stationary points.

Algorithm 6.1 attempts to compute a discrete set of directions M_k on way that $\Phi_k = \bar{\Theta}_k > \max\{\epsilon_c, \epsilon_a\}$. It is done by visiting a finite number of discrete neighborhoods (Line 3) to compute pure continuous fully-linear models that bound $\bar{\Omega}_k$. If the candidate $\tilde{\Omega}_k$ does not satisfy the condition $\tilde{\Omega}_k > \max\{\epsilon_c, \epsilon_a\}$ we keep the same model m^{icb}_k and transformation matrix M_k.

Finally, we perform an additional experiment of Algorithm 4.1 without the *criticality-test* to evaluate the effect of the stationarity measures on its computational performance. We evaluate the 290 instances of the `Generic` set with the *s1-accurate-h* and *s2-aggressive-h* variants, that are the heuristic versions of the variants of the *s1-accurate* *s2-aggressive* variants, respectively.

Algorithm 6.1 ManifoldSearch

Input: Point (x_k, y_k), variable bounds x_{lb}, x_{ub} and y_{lb}, y_{ub}, incumbent trust-region radii Δ_k^{icbc} and Δ_k^{icbz}, set of samples Y^k, transformation matrix M_k and criticality tolerance $\bar{\epsilon}$

Output: Model $\tilde{m}_k$, stationarity parameter $\tilde{\Theta}_k$ and transformation matrix M_k

Set $\tilde{\Theta}_k = 0$
Set $NS_k = (\hat{B}_z(y_k, \Delta_k^{icbz}) \cap \Omega_z) \setminus (Q(y_k, M_k) \cup \{y_k\})$
for $i = 1, \ldots, \min\{|NS_k|, n_z\}$ **do**
    Select a vector y^i from NS_k
    Compute a fully-linear surrogate $m_i(x)$ of $f(x, y^i)$ sampling $n_c + 1$ points $(x, y) \in (\hat{B}_c(x_k, \Delta_k^{icb_c}) \cap \Omega_c) \times \{y^i\}$
    if $f(x_k, y_k) - \min_{x \in \hat{B}_c(x_k, \Delta_k^{icb_c}) \cap \Omega_c} m_i(x) > \tilde{\Theta}_k$ **then**
        Set $\tilde{\Theta}_k = f(x_k, y_k) - \min_{x \in \hat{B}_c(x_k, \Delta_k^{icb_c}) \cap \Omega_c} m_i(x)$
        Set $\bar{y} = y^i$
    end if
    Set $NS_k = NS_k \setminus \{y^i\}$
end for
if $\tilde{\Theta}_k > \bar{\epsilon}$ **then**
    Compute the matrix M_k including the vector $\bar{y}$
    Set $\tilde{Q}_k = Q(y_k, \tilde{M}_k)$
    Set $(\tilde{m}_k, \bar{X}_o^k, \bar{X}_j^k, \bar{Z}^k) =$
    MixedIntegerModelComputation$(x_k, x_{lb}, x_{ub}, y_k, y_{lb}, y_{ub}, \Delta_k^{icbc}, \Delta_k^{icbz}, \tilde{Q}_k, Y^k, 1, 1)$
else
    Set $\tilde{m}_k = m_k^{icb}$ and keep the same M_k
end if

6.1.2 Test Instances

In the experimental section we use three sets of instances: `LQMI`, `Cutest` and `Generic` functions.

- `LQMI` instances.

  This set corresponds to a collection of 315 instances which are known to behave locally or globally as an LQMI function. `LQMI` instances can be split into three sub-classes: (1) globally quadratic functions, (2) quadratic + RBF functions, and, (3) quadratic + element-wise exponential functions:

$$
\begin{aligned}
&(1) \quad f(z) = f_o + c^\top (z - z_o) + \frac{1}{2}(z - z_o)^\top A (z - z_o) \\
&(2) \quad f(z) = f_o + c^\top (z - z_o) + \frac{1}{2}(z - z_o)^\top A (z - z_o) + \sum_{i=1}^{n} \left(\frac{f_{r,i}}{1 + \|x - x_{r,i}\|^2} \right) \\
&(3) \quad f(z) = f_o + c^\top (z - z_o) + \frac{1}{2}(z - z_o)^\top A (z - z_o) + \sum_{i=1}^{n_c} e^{v_i (x_i - x_{o,i})}
\end{aligned}
$$

  where $z^\top = [x^\top, y^\top]$. All the numeric coefficients are randomly generated as follows: $f_o, f_{r,i}$ are uniformly distributed in $[-1000, 1000]$, c is randomly distributed with mean 0 and standard deviation 1, A is normally distributed with mean 0 and standard deviation 1. Moreover, we have $x_{lb} = y_{lb} = -50$ $x_{ub} = y_{ub} = -50$ and the reference parameter z_o is uniformly distributed in $[-50, 50]$, and v_i is uniformly distributed in between $(-10^{-2}, 10^{-2})$. The dimension of the instances $n_c + n_z$ goes from 5 to 14 variables, with 5 different integer ratios $\frac{n_z}{n_c + n_z} = [0.2, 0.4, 0.6, 0.8]$.

- `Cutest` instances.

  This set consists of twice differentiable functions from the *CUTEr/st* testing environment [108], including the: *explin, explin2, mccormck* and *hadamals* instances. This set consists in 174 instances generated from 5 continuous functions from the *Cuter/st* environment. The dimension of these instances $(n_z + n_c)$ ranges from 8 to 12 variables, with 5 different integer ratios $(0.2, 0.4, 0.6, 0, 8)$ and 10 random starting points. We randomly select which of the variables are considered as integer.

TABLE 6.3: Set of `Generic` MINLP test functions

Instance	n_c	n_z	Lower bound	Upper bound
Davidon2	2	2	$[-15, 5]$	$[35, 15]$
Elattar	3	3	$[-8, -8, 3]$	$[12, 12, 17]$
Evd61	3	3	$[-8, -8, 3]$	$[12, 12, 17]$
Exp	3	2	$-[9.5, 10, 10]$	$[10.5, 10, 10]$
Filter	5	4	$-[10, 9, 10, 10.15, 10]$	$[10, 11, 10, 9.85, 10]$
Gamma	2	2	$-[9, 9]$	$[11, 11]$
Hs78	3	2	$[-12, -8.5, 8]$	$[8, 11.5, 12]$
Kowalik	2	2	$-[9.75, 9.61]$	$[10.25, 10.39]$
Maxquad	5	5	$-10 \times J_{5,1}$	$10 \times J_{5,1}$
Oet5	2	2	$-[9, 9]$	$[11, 11]$
Oet6	2	2	$-[9, 9]$	$[11, 11]$
Osbornne2	6	5	$-[8.7, 9.35, 9.35, 9.3, 9.4, 7]$	$[11.3, 10.65, 10.65, 10.7, 10.6, 13]$
Pbc1	2	3	$-[10, 11]$	$[10, 9]$
Prob10	3	2	$-100 \times J_{3,1}$	$100 \times J_{3,1}$
Prob102	3	2	$-100 \times J_{3,1}$	$100 \times J_{3,1}$
$*$ Prob107	5	5	$3 \times J_{10,1}$	$9 \times J_{10,1}$
$*$ Prob109	6	6	$-1 \times J_{12,1}$	$3 \times J_{12,1}$
$*$ Prob113	5	5	$3 \times J_{10,1}$	$99 \times J_{10,1}$
$*$ Prob115	6	6	$10 \times J_{12,1}$	$30 \times J_{12,1}$
$*$ Prob116	4	4	$-10 \times J_{8,1}$	$10 \times J_{8,1}$
$*$ Prob206	8	7	$-15 \times J_{15,1}$	$30 \times J_{15,1}$
$*$ Prob208	5	10	$-15 \times J_{15,1}$	$30 \times J_{15,1}$
Rosen-susuki	2	2	$-[10, 10]$	$[10, 10]$
Shelldual	8	7	$-9.9999 \times J_{8,1}$	$10.0001 \times J_{8,1}$
Shor	3	2	$-10 \times J_{3,1}$	$10 \times J_{3,1}$
Steiner2	6	6	$[-9.3333, -8.1111, -7.6666 -7, -6.3333, -4.9444]$	$[10.6666, 11.8888, 12.3333, 13, 13.6666, 15.0555]$
Transformer	3	3	$-[9.2, 8.5, 8.8]$	$[10.8, 11.5, 11.2]$
Wong1	4	3	$-[9, 8, 10, 6]$	$[11, 12, 10, 14]$
Wong2	5	5	$-[8, 7, 5, 5, 9]$	$[12, 13, 15, 15, 11]$

- `Generic` instances.

  This set consists of *non-smooth* and *discontinuous* problems proposed in [74], [109], accounting 29 test functions, each one of them tested with 10 random starting points. Table 6.3 reports the characteristics of all the `Generic` instances. For the instances which are not marked with $*$ the bounds on the discrete variables are $[0, 100]$, thus, only the continuous bounds are reported.

6.1.3 Discussion

Figures 6.1 and 6.2 display the profiles related to the `LQMI` instances with the convergence tolerances $\sigma \in \{0.01, 0.001\}$. The figures show that Algorithm 4.1 is better

suited than NOMAD for the solution of LQMI instances. Indeed, both *s1-accurate* and *s2-aggressive* are faster in the reduction of the objective function, both are able to provide better solutions and solve a larger number of instances. A possible explanation of the better results of both variants is the fact that the model provides an exact estimation of the integer part of the original function. The variant *s1-accurate* presents performance better than *s2-aggressive*, this is probably due to the fact that at each iteration this variant computes a fully-linear approximation of the model and does not "waste samples" in running heuristics. Therefore, the use of accurate model information accelerate the objective improvement when the function studied presents the LQMI structure. It is worth noting that none of these variants are able to solve 100% of the instances tested, this is related to the fact that Algorithm 4.1 might stop at a stationary point that does not correspond to a global optimum.

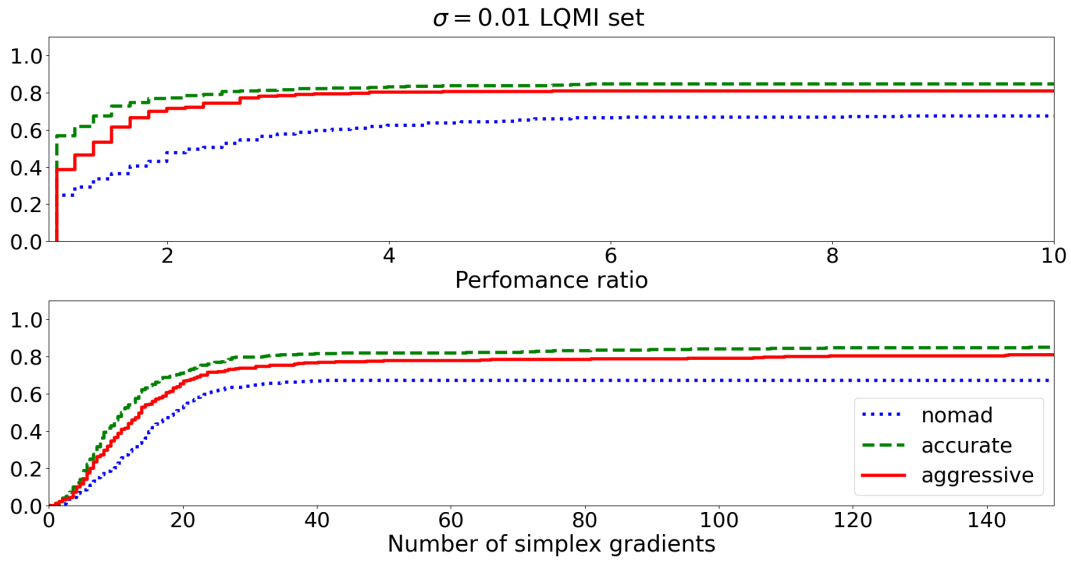


FIGURE 6.1: Performance and Data profiles on LQMI instances (Algorithm 4.1), $\sigma = 0.01$

Figures 6.3 and 6.4 show the profiles of the evaluation of the `Cutest` set instances with the convergence tolerances $\sigma \in \{0.01, 0.001\}$. From both performance and data profiles we observe that variant *s2-aggressive* is competitive with NOMAD in terms of instances solved and convergence rate. Moreover, in the $\sigma = 0.01$ case, *s2-aggressive* is able to solve a slightly larger fraction of test instances. In contrast to the `LQMI` set, variant *s2-aggressive* presents better performance than *s1-accurate* in terms of convergence rate and number of problems solved, indicating that quick model

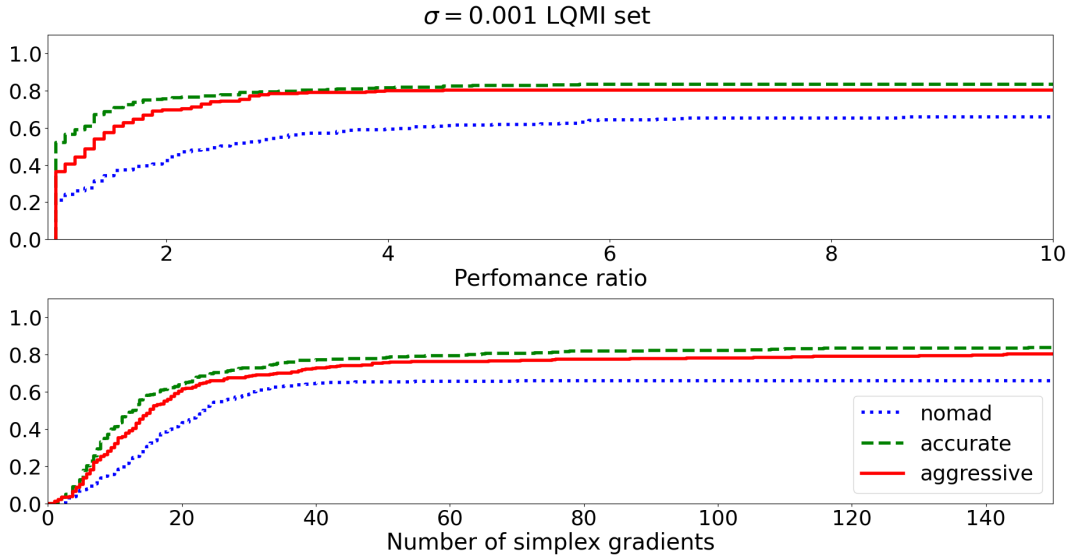


FIGURE 6.2: Performance and Data profiles on LQMI instances (Algorithm 4.1), $\sigma = 0.001$

update and the search heuristic might help to avoid early convergence to mixed-integer stationary points that could be potentially significantly far from the global optimum.

Figures 6.5 and 6.6 show the profiles of the evaluation of the `Generic` set of instances with the convergence tolerances $\sigma \in \{0.01, 0.001\}$. Both figures show that NOMAD is faster in attaining algorithmic convergence, while Algorithm 4.1 presents slower convergence in both variants. One possible explanation for the overall better performance of NOMAD is the fact that NOMAD is tuned with a series of heuristics that are focused on finding a good local optimum. On the other hand, Algorithm 4.1 is mainly tuned for a proved convergence to a stationary point. We also remark that Algorithm 4.1 uses a surrogate that is only globally accurate under the assumption of having an LQMI function.

Finally, Figures 6.7 and 6.8 show the profiles of the evaluation of the `Generic` set of instances using the heuristic variants of Algorithm 4.1 with tolerances $\sigma \in \{0.01, 0.001\}$. Both figures show that NOMAD is also faster than the heuristic versions of Algorithm 4.1 in attaining convergence; nonetheless, we observe that the variants *s1-accurate-h* and *s2-aggressive-h* solve a larger amount of instances with respect to the implementation that considers function stationarity. This improvement in the computational performance is noticed in the $\sigma = 0.001$ case (Figure 6.8), where

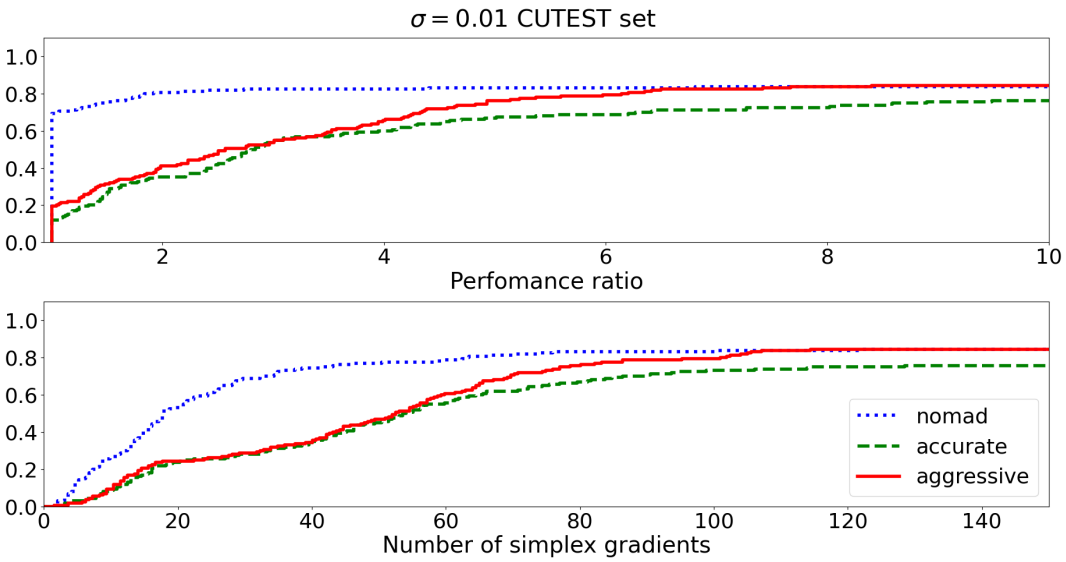


FIGURE 6.3: Performance and Data profiles on `Cutest` instances, $\sigma = 0.01$

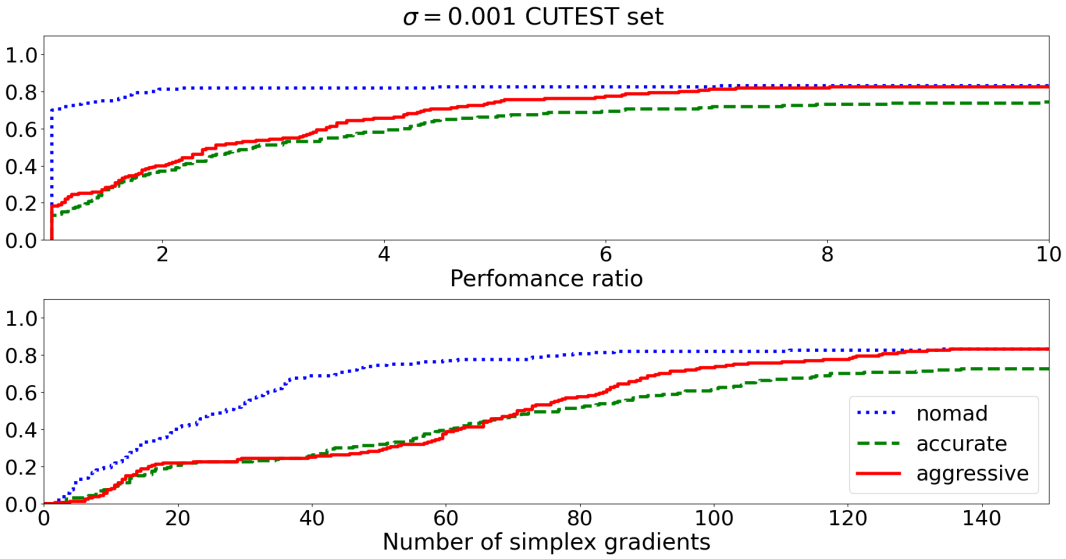


FIGURE 6.4: Performance and Data profiles on `Cutest` instances, $\sigma = 0.001$

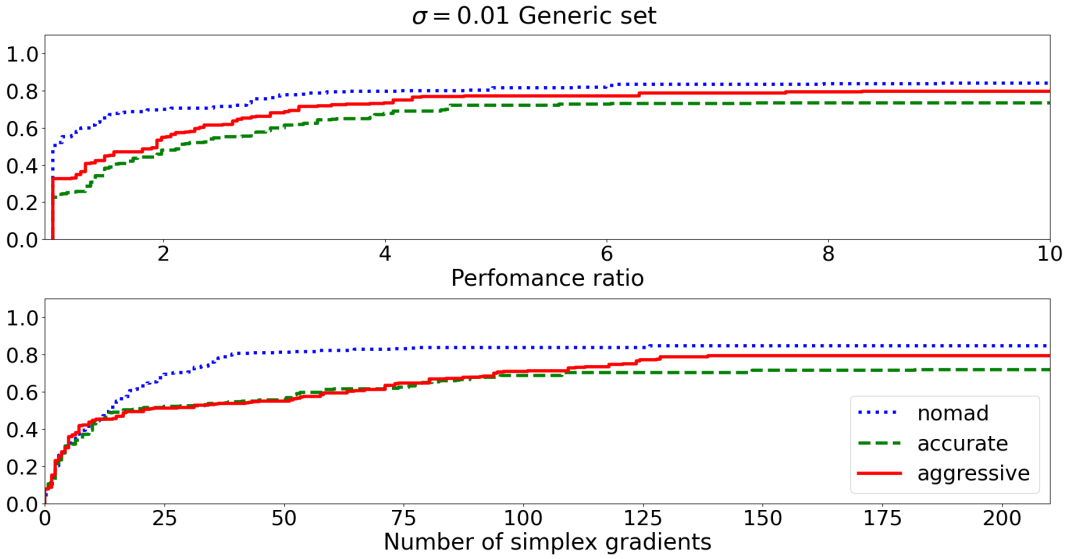


FIGURE 6.5: Performance and Data profiles on `Generic` instances, $\sigma = 0.01$

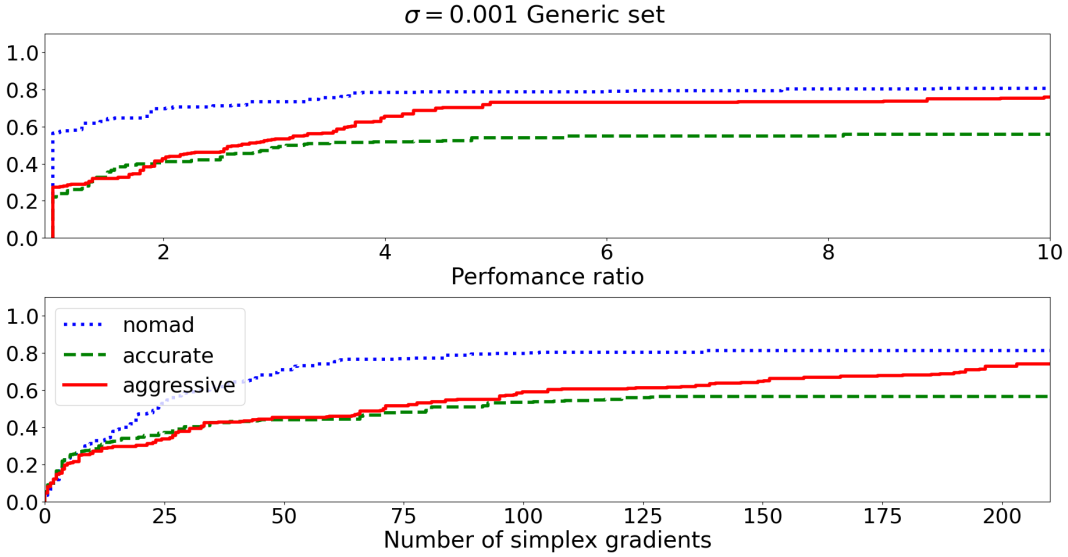


FIGURE 6.6: Performance and Data profiles on `Generic` instances, $\sigma = 0.001$

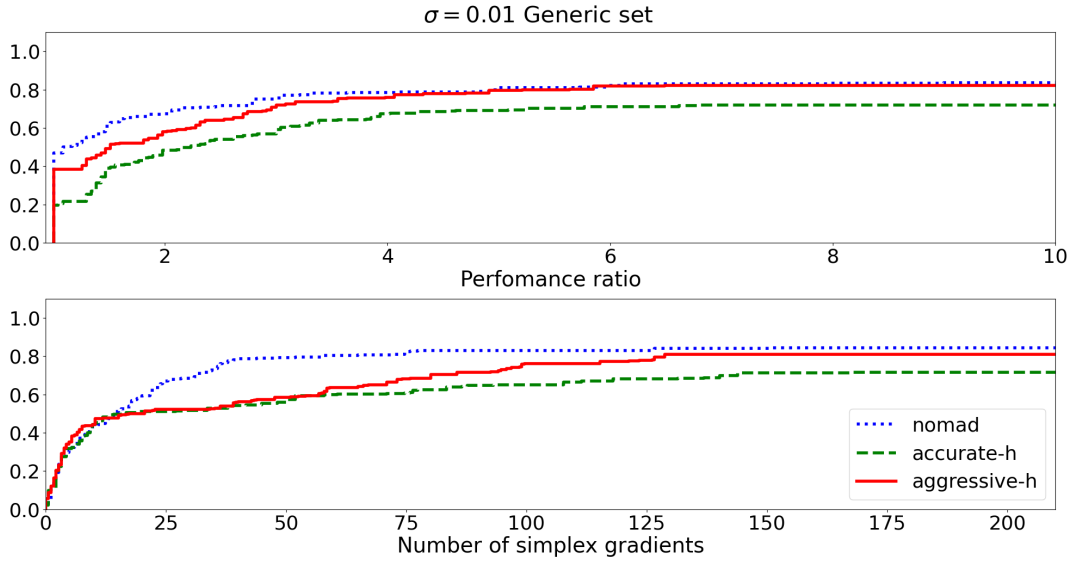


FIGURE 6.7: Performance and Data profiles on `Generic` instances on heuristic versions of Algorithm 4.1, $\sigma = 0.01$

the variant *s2-aggressive-h* solves a slightly larger fraction of problems than NOMAD does (79.5% vs 77.7%).

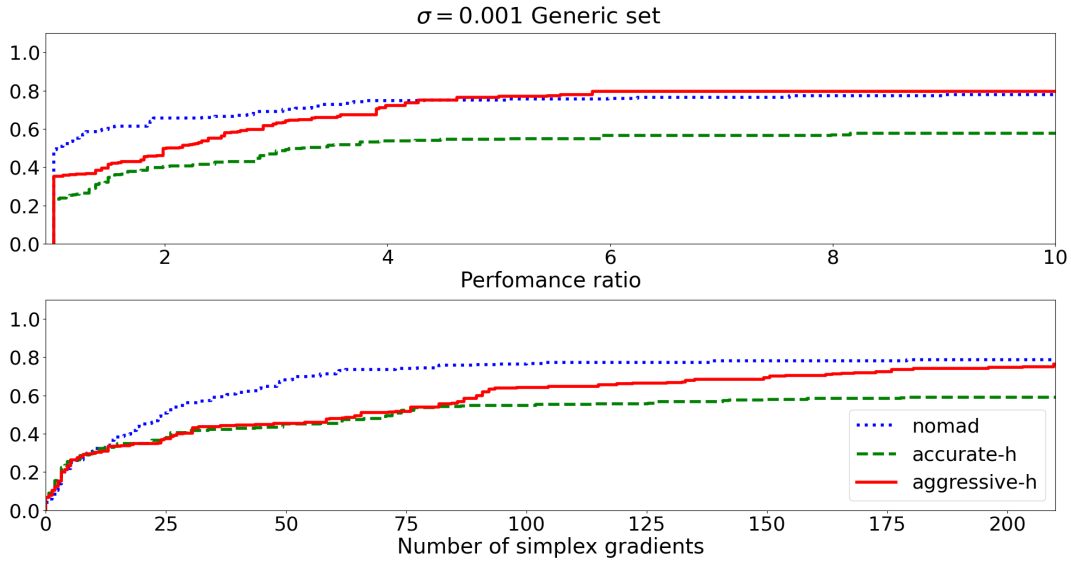


FIGURE 6.8: Performance and Data profiles on `Generic` instances instances on heuristic versions of Algorithm 4.1, $\sigma = 0.001$

The computational experiments presented in this section show that Algorithm 4.1 is well suited for solving LQMI functions and is competitive in the solution of functions that do not exhibit these type of properties. However, in the the general mixed-integer case it presents slower rate of objective decrease in comparison to a

general purpose DFO solver like NOMAD. This is the result of two features of Algorithm 4.1. First, it devotes a large number of objective evaluations to construct and certify a *fully-linear* approximation, specially with respect to the elements of the integer interpolation l^z and A^z, using $\mathcal{O}(n_z^2)$ samples. Second, despite incorporating tools that minimize the effect of stationarity such as the **MandifoldSearch** procedure (Algorithm 6.1), Algorithm 4.1 is still sensitive to stationary points. We observe this behaviour in Figure 6.9 and Table 6.4 that show the direct comparison of the variants *s1-accurate* and *s2-aggressive* with their heuristic counterparts. Figure 6.9 shows that the heuristic variants provide better performance in terms of objective reduction, solving to a strong degree of tolerance ($\sigma = 10^{-5}$) more than 20% of instances than the convergent variants. Table 6.4 displays the number of instances where the ratio between the best objective improvement yield by the proven convergent variants of Algorithm 4.1 and the best improvement attained by their respective heuristics is larger than $1 - \sigma$. At large values of σ we appreciate that for at least 30 of 290 instances *s1* and *s2* fail to attain 90% of the improvement reached by the heuristic variants, implying early convergence to suboptimal points. In addition, as the value of σ decreases a smaller number of instances satisfy the improvement condition with respect to the heuristics. It indicates that Algorithm 4.1 decreases its objective improvement rate as it gets close to a StLM.

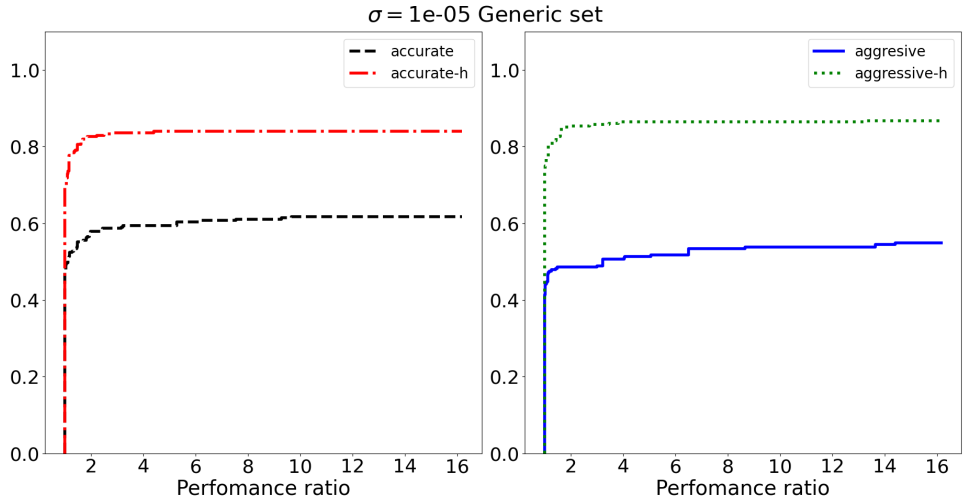


FIGURE 6.9: Performance profiles for convergent variants vs heuristics variants of Algorithm 4.1, $\sigma = 10^{-}5$

σ	*s1-accurate*	*s2-aggressive*
10^{-1}	269	247
10^{-2}	238	228
10^{-3}	221	210
10^{-4}	210	181
10^{-5}	183	163
10^{-10}	152	112

TABLE 6.4: Number of instances (out of 290 instances) such that the convergent variants presents objective improvement larger than $1-\sigma$ times the improvement yield by the heuristic versions of Algorithm 4.1

.

6.2 Computational Results for Algorithm 5.4

In this section, we study the computational performance of Algorithm 5.4. We evaluate three algorithmic versions that are described in Table 6.6: $s3$ or *opportunistic*, $s4$ or *extensive*, and, $s5$ or *global*.

6.2.1 Experimental Setting

We implemented Algorithm 5.4 in Python, using Pyomo [105], [106] to prototype the monomial pivotal optimization used in Algorithm 5.5. The algorithmic implementation uses two methodologies to address the (partial) integer optimization subproblems (Lines 18 and 22-Algorithm 5.4). For the global optimization of M and $M^{\natural} - convex$ functions we use Shioura's *fast-scaling* algorithm [98]. For the partial integer optimization we incorporate PyNomad. All the tests instances were evaluated in a single machine Lyon/hercule at GRID5000 cluster, with a Intel Xeon E5-2620 CPU and 32 GB of RAM. Finally, we consider algorithm termination when $\Delta_k^c < 10^{-5}$, with $k_{max} = 150$.

In Table 6.5 we present the values used for the fixed parameters. In Table 6.6 we resume the values of the variable parameters $(ISG, \mathbf{PO})$ associated to the three variants of Algorithm 5.4. The parameter ISG (*Integer Simplex Gradients*) controls the maximum number of function evaluations allowed in the partial integer optimization (Line 18 - Algorithm 5.4). The variant *s3-opportunistic* restricts the integer search to at most $4(n_z+1)$ points, unless the parameter $\|\mathbf{red}(\nabla_x m_k(x_k), x_k)\|$ is below the criticality tolerance ϵ_c. The variant *s4-extensive* allows larger exploration on the set

Ω_z and allows to NOMAD to use at most $20(n_z + 1)$ samples. On the other hand, the variant *s5-global* solves the integer subproblem to optimality at every iteration.

TABLE 6.5: Fixed parameters for Algorithm 5.4

η_o	η_1	γ_0	γ_1	ϵ_c	ω	β	μ
0.15	0.75	0.75	1.3	$1e^{-6}$	0.6	500	1000
$\Delta^c_{max} = \lceil \min_{i\in\{1,\dots,n_c\}}\{x_{ub,i} - x_{lb,i}\}/2 \rceil$							
$\Delta^c_o = \max\{\Delta^c_{max}/2, 0.1\}$							

TABLE 6.6: Variable parameters for Algorithm 5.4

	ISG	**PO**
s3 - *opportunistic*	4	1
s4 - *extensive*	20	1
s5 - *global*	∞	0

6.2.2 Test Instances

- $M^\natural$ `Quadratic` instances.

  This set corresponds to a collection of 411 instances which are known to behave globally as an $M^\natural$ function. $M^\natural$ instances can be split into two sub-classes: (1) globally quadratic functions and (2) quadratic + element-wise exponential functions:

$$(1) \quad f(z) = f_o + c^\top (z - z_o) + \frac{1}{2}(z - z_o)^\top A (z - z_o)$$

$$(2) \quad f(z) = f_o + c^\top (z - z_o) + \frac{1}{2}(z - z_o)^\top A (z - z_o) + \sum_{i=1}^{n_c+n_z} e^{v_i(z_i - z_{o,i})}$$

  where $z^\top = [y^\top, x^\top]$. All the numeric coefficients are randomly generated as follows: f_o are uniformly distributed in $[-1000, 1000]$, c is randomly distributed with mean 0 and standard deviation 1. The matrix A has the following properties:

  – If $i, j > n_z$ or $i = j \leq n_z$ the entry $a_{i,j}$ is normally distributed with mean 0 and standard deviation 1.

- If $i, j \leq n_z$ and $i \neq j$, then $a_{i,j} = b$ where b is uniformly distributed between 0 and $2 \min_{i \in \{1,\ldots,n_z\}} a_{i,i}$. This condition guarantees that the function f is $M^\natural$-convex after fixing the vector of continuous variables [93].

Moreover, we have $x_{lb} = y_{lb} = -50$, $x_{ub} = y_{ub} = -50$ and the reference parameter z_o is uniformly distributed in $[-50, 50]$, and v_i is uniformly distributed in between $(-10^{-2}, 10^{-2})$. The dimension of the instances $n_c + n_z$ goes from 10 to 25 variables with $n_c \in \{5, \ldots, 10\}$ and $n_z \in \{5, \ldots, 15\}$.

- `Generic` instances

  The same set of 29 general mixed-integer instances described in Section 6.1.2.

6.2.3 Discussion

Figures 6.10 and 6.11 show the profiles of the evaluation of the $M^\natural$-`Quadratic set` with the convergence tolerance $\sigma \in \{0.5, 0.001\}$. Both figures show that the *s3-opportunistic* variant presents the best performance in terms of objective improvement and rate of objective decrease. The variants *s4-extensive* and *s5-global* solve approximately the same number of instances as *s3*; however, they use significantly more samples to achieve the desired objective improvement. This is explained by the constraint on the number of samples used on the partial discrete optimization. For example, the variant *s5-global* uses around $\mathcal{O}(n_z^2)$ function evaluations to certify global optimality; this process is computationally expensive and does not yield any objective improvement. Moreover, Figure 6.10 indicates that NOMAD presents the worst performance for this set of instances as it solves less than 5% of problems to the desired degree of optimality. The global integer optimization included in Algorithm 5.4 allows it converge to a strong SLM that NOMAD cannot identify using local information.

Figures 6.12 and 6.13 show the profiles of the evaluation of the `Generic set` with the convergence tolerance $\sigma \in \{0.01, 0.0001\}$. In this experiment we did not include the *s5-global* variant as it displays slower convergence than *s3* and *s4*. Both figures show that NOMAD solves a larger number of instances than the two variants of Algorithm 5.4. A possible explanation for the overall better performance of NOMAD

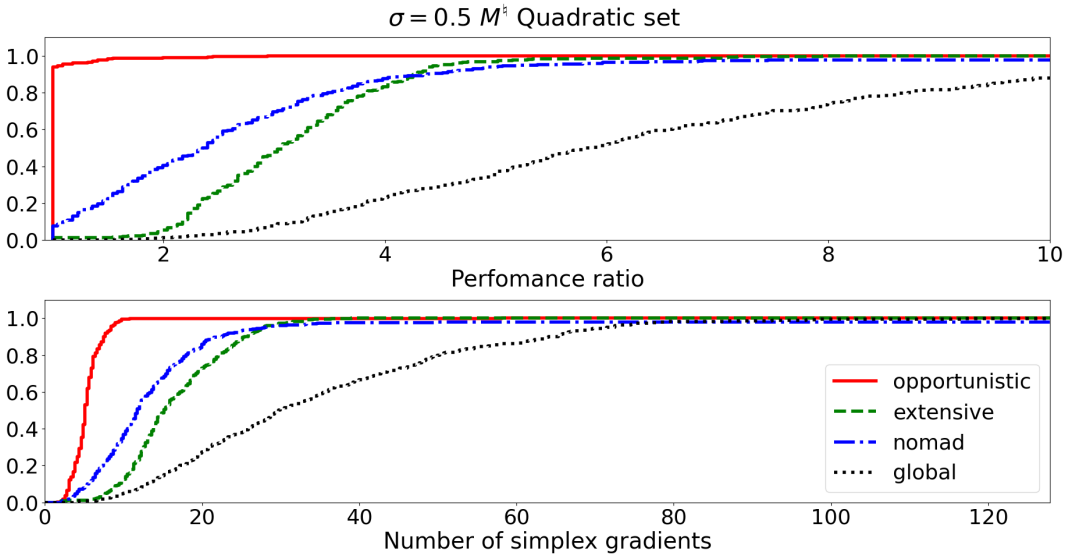


FIGURE 6.10: Performance and Data profiles on $M^\natural$ `Quadratic` instances, $\sigma = 0.5$

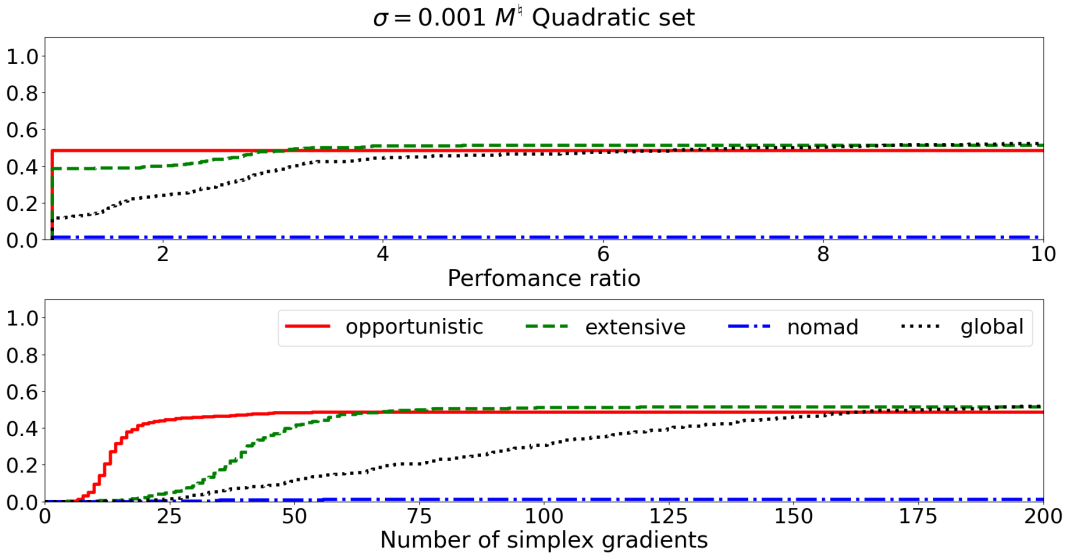


FIGURE 6.11: Performance and Data profiles on $M^\natural$ `Quadratic` instances, $\sigma = 0.001$

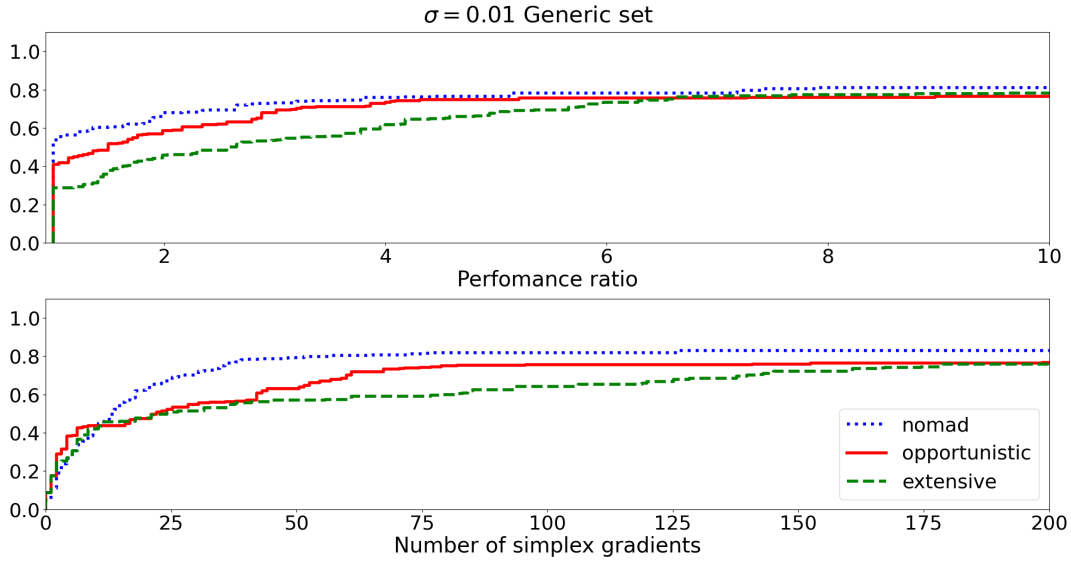


FIGURE 6.12: Performance and Data profiles on `Generic` instances (Algorithm 5.4), $\sigma = 0.01$

is the fact that Algorithm 5.4 relies only on first-order information, ignoring the interaction between integer and continuous variables, for that reason it converges to worst stationary points. Similar to the $M^\natural$ `Quadratic` set of instances, the variant *s3-opportunistic* is faster than *s4-extensive*, this is probably due to the fact that at each iteration this variant is more efficient in solving the integer subproblem. It is highly possible that for two or more consecutive iterations the output of the integer optimization subproblem remains the same, specially when the trust-region radius becomes small. Therefore, if the parameter IGS is smaller, not many samples would be devoted to improve the objective when no point on Ω_z yields better objective value than the current integer candidate.

6.3 Comparison between Algorithms 4.1 and 5.4

In this section we compare the performance of Algorithm 4.1 and 5.4 with respect to the `Generic` set of instances. Figures 6.14 and 6.15 show the performance and data profiles at $\sigma \in \{0.01, 0.001\}$. Both figures show that the variant that yields the largest number of instances solved is *s2-aggressive*, while the one that presents the best overall performance is the variant *s3-opportunistic*. Note that *s2* is the variant of Algorithm 4.1 that uses the least amount of samples to construct a surrogate approximation and the variant *s3* is the variant of Algorithm 5.4 that uses least objective

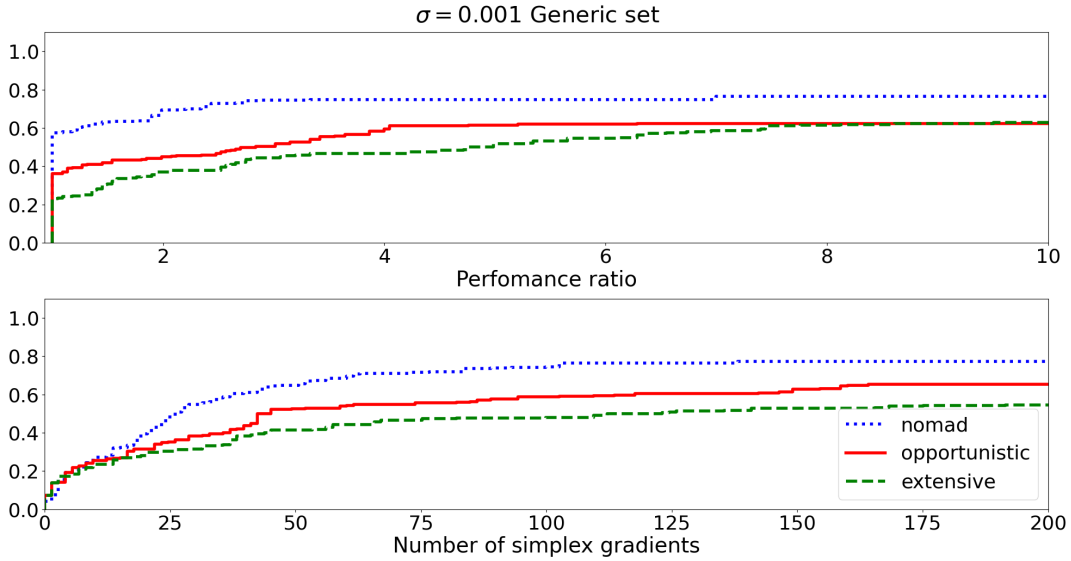


FIGURE 6.13: Performance and Data profiles on `Generic` instances (Algorithm 5.4), $\sigma = 0.001$

evaluations to optimize over the integer domain. These results indicate that for general mixed-integer instances a better performance is attained with least expensive surrogates and an intermediate degree of global integer exploration. On the other hand, the use of *fully-linear* but computationally expensive surrogates decreases the rate of objective improvement and increments the probability of early convergence to stationary points. The same result is observed while using the variant *s4-extensive*.

Figures 6.14 and 6.15 suggest that in the future the principles behind the LQMI surrogate approximation (Chapter 3) and DC reformulation (Chapter 5) can be mixed to devise a general purpose derivative-free solver with better performance than Algorithms 4.1 and 5.4:

- It should consider *fully-linear* mixed-integer approximations to prove convergence to SLM and StLM. Nonetheless, it should reduce the number objective evaluations devoted to compute and evaluate *fully-linearity*. This can be done by only considering the set of points $\Omega_d^G(y_k, M_k)$ instead of $\Omega_d^Q(y_k)$, decreasing the time used to evaluate the $\mathcal{O}(n_z^2)$ samples required to compute the model $\tau(y - y_k)$.

- It should include the **RescueProcedure** (Algorithm 4.5) to preserve the convergence to StLM of Algorithm 4.1.

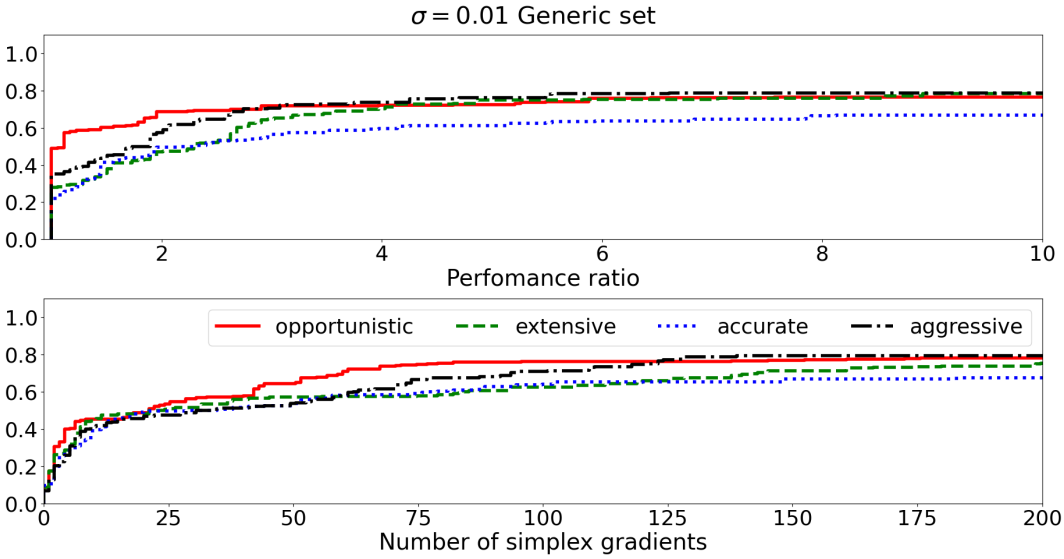


FIGURE 6.14: Performance and Data profiles on `Generic` instances, $\sigma = 0.01$

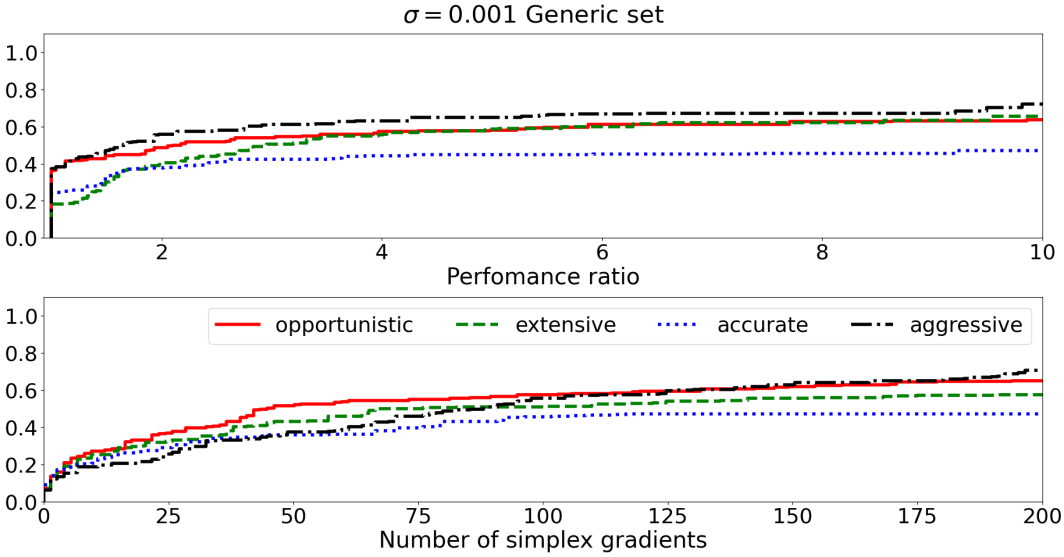


FIGURE 6.15: Performance and Data profiles on `Generic` instances, $\sigma = 0.001$

- It should incorporate a local/global search on the domain $\{x_k\} \times \Omega_z$ that accelerates the objective improvement and facilitates the prove of convergence to a SLM. However, the number of samples during this partial integer optimization should be bounded to prevent slow convergence.

- Finally, it should consider underdetermined surrogate approximations like *s2-aggressive*. It accelerates the rate of convergence and reduces the probability of early convergence to weak stationary points.

Chapter 7

Conclusions

In this project we have addressed the optimization of box-constrained mixed-integer problems when the objective function is computationally expensive to evaluate. This thesis focuses on the development of two algorithms that address the following open questions in the mixed-integer derivative-free setting: *How to extend the convergence of the model-based methods into the mixed integer domain?*, and, *How to take advantage of the structure of a mixed-integer function?*. The first algorithm is presented in Chapter 4 and is devised to solve problems that display local bilinear interaction between discrete and continuous variables. The second algorithm is introduced in Chapter 5 and is designed to solve functions that exhibit combinatorial discrete properties when the vector of continuous variables is fixed.

In Chapter 3 we introduced the concept of *locally-quadratic mixed-integer* (LQMI) functions. This class of functions exhibit a separable structure that allows us to overcome the lack of error bounds for the surrogate approximation of a mixed-integer function. With these results, we introduced the definition of mixed-integer *fully-linear* surrogate models and devised a general method for the computation of accurate models for the LQMI functions. Moreover, we researched on the methods for the fast computation and update of LQMI models. Finally, we proved that the framework for computing the models that accurately represent an LQMI function can be used for the approximation of a general mixed-integer function on a reduced domain.

Chapter 4 introduces a new model-based algorithm that uses the LQMI framework presented in Chapter 3. This mixed-integer algorithm adapts the trust-region method and presents some features that overcome the difficulties that arise from

dealing with mixed-integer variables, including new mixed-integer stationarity parameters and the mixed-integer version of the *criticality step*. Moreover, this algorithm is proved to be globally convergent to a separate local minimum (SLM) and a strong local minimum (StLM).

In Chapter 5 we explored the methods to optimize a function that exhibits combinatorial properties such as the M and $M^\natural$ discrete convexity. We developed a hybrid algorithm that combines the principles of the difference of convex algorithm (DCA) and the trust-region method. This algorithm is based on the reformulation of the objective function via partial discrete optimization and convex regularization. Furthermore, we proved that this algorithm is globally convergent to a SLM and can be used to solve general mixed-integer problems.

Finally, in Chapter 6 we studied the computational performance of the two algorithms for the solution of structured functions. The numerical experiments in Section 6.1 show that the algorithm introduced in Chapter 4 is well suited for the solution of LQMI instances. The tests performed in Section 6.2 show that the hybrid algorithm presented in Chapter 5 is efficient in the solution of $M^\natural$-convex instances. Moreover, both algorithms are competitive in the solution of general mixed-integer functions. We remark that both algorithms are two of the few model-based methodologies present in the literature that are able to converge to a SLM and a StLM.

We hope the results presented in this project motivate further research on mixed-integer derivative-free methods. Two main topics can be further explored. The first topic is the development of general mixed-integer *fully-linear* surrogates. In Chapter 3 we introduce a framework that uses quadratic polynomials to approximate a mixed-integer function; however, it is desirable to extend these results to other types of surrogate models like the radial basis functions (RBF). We believe this adaption can exploit the advantages of surrogate global models while preserving the convergence to local minimizers.

The second topic is the type of constraints that our algorithms can handle. Both algorithms were developed to tackle bound constraints; however, many real world problems include complicated non-linear constraints. We consider that our methodology can be adapted to deal with this type of constraints using the *penalty*, *augmented Lagrangian* or *progressive barrier* methods.

Bibliography

[1] A. L. Marsden, J. A. Feinstein, and C. A. Taylor, "A computational framework for derivative-free optimization of cardiovascular geometries," *Computer methods in applied mechanics and engineering*, vol. 197, no. 21-24, pp. 1890–1905, 2008.

[2] R. Oeuvray, "Trust-region methods based on radial basis functions with application to biomedical imaging," EPFL, Tech. Rep., 2005.

[3] C. Audet, V. Béchard, and J. Chaouki, "Spent potliner treatment process optimization using a mads algorithm," *Optimization and Engineering*, vol. 9, no. 2, pp. 143–160, 2008.

[4] M. C. Bartholomew-Biggs, S. C. Parkhurst, and S. P. Wilson, "Using direct to solve an aircraft routing problem," *Computational Optimization and Applications*, vol. 21, no. 3, pp. 311–323, 2002.

[5] J. Han, M. Kokkolaras, and P. Y. Papalambros, "Optimal design of hybrid fuel cell vehicles," *Journal of fuel cell science and technology*, vol. 5, no. 4, 2008.

[6] R. Yamamoto and N. Hibiki, "Optimal multiple pairs trading strategyusing derivative free optimizationunder actual investment management conditions," *Journal of the Operations Research Society of Japan*, vol. 60, no. 3, pp. 244–261, 2017.

[7] J. Larson, M. Menickelly, and S. M. Wild, "Derivative-free optimization methods," *Acta Numerica*, vol. 28, pp. 287–404, 2019.

[8] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, "Mixed-integer nonlinear optimization," *Acta Numerica*, vol. 22, pp. 1–131, 2013.

[9] E. Newby and M. M. Ali, "A trust-region-based derivative free algorithm for mixed integer programming," *Computational Optimization and Applications*, vol. 60, no. 1, pp. 199–229, 2015.

[10] T. T. Tran, D. Sinoquet, S. Da Veiga, and M. Mongeau, "Derivative-free mixed binary necklace optimization for cyclic-symmetry optimal design problems," *Optimization and Engineering*, pp. 1–42, 2021.

[11] H. Zhang, A. R. Conn, and K. Scheinberg, "A derivative-free algorithm for least-squares minimization," *SIAM Journal on Optimization*, vol. 20, no. 6, pp. 3555–3576, 2010.

[12] H. Zhang and A. R. Conn, "On the local convergence of a derivative-free algorithm for least-squares minimization," *Computational optimization and applications*, vol. 51, no. 2, pp. 481–507, 2012.

[13] C. Cartis and L. Roberts, "A derivative-free gauss–newton method," *Mathematical Programming Computation*, vol. 11, no. 4, pp. 631–674, 2019.

[14] B. Colson and P. L. Toint, "Optimizing partially separable functions without derivatives," *Optimization methods and software*, vol. 20, no. 4-5, pp. 493–508, 2005.

[15] C. J. Price and P. L. Toint, "Exploiting problem structure in pattern search methods for unconstrained optimization," *Optimisation Methods and Software*, vol. 21, no. 3, pp. 479–491, 2006.

[16] A. S. Bandeira, K. Scheinberg, and L. N. Vicente, "Computation of sparse low degree interpolating polynomials and their application to derivative-free optimization," *Mathematical programming*, vol. 134, no. 1, pp. 223–257, 2012.

[17] R. Garmanjani, D. Júdice, and L. N. Vicente, "Trust-region methods without using derivatives: Worst case complexity and the nonsmooth case," *SIAM Journal on Optimization*, vol. 26, no. 4, pp. 1987–2011, 2016.

[18] A. R. Conn and L. N. Vicente, "Bilevel derivative-free optimization and its application to robust optimization," *Optimization Methods and Software*, vol. 27, no. 3, pp. 561–577, 2012.

[19] J. Larson, S. Leyffer, P. Palkar, and S. M. Wild, *A method for convex black-box integer global optimization*, 2019. arXiv: 1903.11366 [math.OC].

[20] S. Lucidi and M. Sciandrone, "On the global convergence of derivative-free methods for unconstrained optimization," *SIAM Journal on Optimization*, vol. 13, no. 1, pp. 97–116, 2002. DOI: 10.1137/S1052623497330392. eprint: https://doi.org/10.1137/S1052623497330392. [Online]. Available: https://doi.org/10.1137/S1052623497330392.

[21] L. Grippo and F. Rinaldi, "A class of derivative-free nonmonotone optimization algorithms employing coordinate rotations and gradient approximations," *Computational Optimization and Applications*, vol. 60, no. 1, pp. 1–33, 2015.

[22] C. T. Kelley, *Implicit filtering*. SIAM, 2011.

[23] T. D. Choi and C. T. Kelley, "Superlinear convergence and implicit filtering," *SIAM Journal on Optimization*, vol. 10, no. 4, pp. 1149–1162, 2000.

[24] C. Cartis, N. I. M. Gould, and P. L. Toint, "On the oracle complexity of first-order and derivative-free algorithms for smooth nonconvex minimization," *SIAM Journal on Optimization*, vol. 22, no. 1, pp. 66–86, 2012. DOI: 10.1137/100812276. eprint: https://doi.org/10.1137/100812276. [Online]. Available: https://doi.org/10.1137/100812276.

[25] R. Hooke and T. A. Jeeves, ""direct search"solution of numerical and statistical problems," *Journal of the ACM (JACM)*, vol. 8, no. 2, pp. 212–229, 1961.

[26] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[27] D. J. Woods, "An interactive approach for solving multi-objective optimization problems," Tech. Rep., 1985.

[28] C. T. Kelley, "Detection and remediation of stagnation in the nelder–mead algorithm using a sufficient decrease condition," *SIAM journal on optimization*, vol. 10, no. 1, pp. 43–55, 1999.

[29] S. Singer and S. Singer, "Efficient implementation of the nelder–mead search algorithm," *Applied Numerical Analysis & Computational Mathematics*, vol. 1, no. 2, pp. 524–534, 2004.

[30] W. Spendley, G. R. Hext, and F. R. Himsworth, "Sequential application of simplex designs in optimisation and evolutionary operation," *Technometrics*, vol. 4, no. 4, pp. 441–461, 1962.

[31] K. I. McKinnon, "Convergence of the nelder–mead simplex method to a non-stationary point," *SIAM Journal on optimization*, vol. 9, no. 1, pp. 148–158, 1998.

[32] P. Tseng, "Fortified-descent simplicial search method: A general approach," *SIAM Journal on Optimization*, vol. 10, no. 1, pp. 269–288, 1999.

[33] Á. Bűrmen, J. Puhan, and T. Tuma, "Grid restrained nelder-mead algorithm," *Computational optimization and applications*, vol. 34, no. 3, pp. 359–375, 2006.

[34] L. M. Rios and N. V. Sahinidis, "Derivative-free optimization: A review of algorithms and comparison of software implementations," *Journal of Global Optimization*, vol. 56, no. 3, pp. 1247–1293, 2013.

[35] V. Torczon, "On the convergence of pattern search algorithms," *SIAM Journal on optimization*, vol. 7, no. 1, pp. 1–25, 1997.

[36] G. A. Gray and T. G. Kolda, "Appspack 4.0: Asynchronous parallel pattern search for derivative-free optimization.," Citeseer, Tech. Rep., 2004.

[37] R. M. Lewis and V. Torczon, "Pattern search algorithms for bound constrained minimization," *SIAM Journal on optimization*, vol. 9, no. 4, pp. 1082–1099, 1999.

[38] C. Audet and J. E. Dennis Jr, "Mesh adaptive direct search algorithms for constrained optimization," *SIAM Journal on optimization*, vol. 17, no. 1, pp. 188–217, 2006.

[39] ——, "A progressive barrier for derivative-free nonlinear programming," *SIAM Journal on optimization*, vol. 20, no. 1, pp. 445–472, 2009.

[40] M. A. Abramson, C. Audet, J. E. Dennis Jr, and S. L. Digabel, "Orthomads: A deterministic mads instance with orthogonal directions," *SIAM Journal on Optimization*, vol. 20, no. 2, pp. 948–966, 2009.

[41] M. A. Abramson and C. Audet, "Convergence of mesh adaptive direct search to second-order stationary points," *SIAM Journal on Optimization*, vol. 17, no. 2, pp. 606–619, 2006.

[42] J. J. Moré and S. M. Wild, "Benchmarking derivative-free optimization algorithms," *SIAM Journal on Optimization*, vol. 20, no. 1, pp. 172–191, 2009.

[43] A. Conn, K Scheinberg, and P. Toint, "Manual for fortran software package dfo v1. 2," 2000.

[44] M. Marazzi and J. Nocedal, "Wedge trust region methods for derivative free optimization," *Mathematical programming*, vol. 91, no. 2, pp. 289–305, 2002.

[45] F. V. Berghen and H. Bersini, "Condor, a new parallel, constrained extension of powell's uobyqa algorithm: Experimental results and comparison with the dfo algorithm," *Journal of computational and applied mathematics*, vol. 181, no. 1, pp. 157–175, 2005.

[46] M. J. Powell, "Uobyqa: Unconstrained optimization by quadratic approximation," *Mathematical Programming*, vol. 92, no. 3, pp. 555–582, 2002.

[47] ——, "The newuoa software for unconstrained optimization without derivatives," in *Large-scale nonlinear optimization*, Springer, 2006, pp. 255–297.

[48] M. Powell, "New developements of newuoa for minimization without derivatives," *Tech. Rep*, 2007.

[49] M. J. Powell, "Developments of newuoa for minimization without derivatives," *IMA journal of numerical analysis*, vol. 28, no. 4, pp. 649–664, 2008.

[50] ——, "The bobyqa algorithm for bound constrained optimization without derivatives," *Cambridge NA Report NA2009/06, University of Cambridge, Cambridge*, pp. 26–46, 2009.

[51] A. R. Conn, K. Scheinberg, and L. N. Vicente, "Geometry of interpolation sets in derivative free optimization," *Mathematical programming*, vol. 111, no. 1-2, pp. 141–172, 2008.

[52] P. G. Ciarlet and P.-A. Raviart, "General lagrange and hermite interpolation in rn with applications to finite element methods," *Archive for Rational Mechanics and Analysis*, vol. 46, no. 3, pp. 177–199, 1972.

[53] D Winfield, "Function minimization by interpolation in a data table," *IMA Journal of Applied Mathematics*, vol. 12, no. 3, pp. 339–347, 1973.

[54] A. R. Conn and P. L. Toint, "An algorithm using quadratic interpolation for unconstrained derivative free optimization," in *Nonlinear optimization and applications*, Springer, 1996, pp. 27–47.

[55] A. Verdério, E. Karas, L. Pedroso, and K. Scheinberg, "On the construction of quadratic models for derivative-free trust-region algorithms," *EURO Journal on Computational Optimization*, vol. 5, no. 4, pp. 501–527, 2017, ISSN: 2192-4406. DOI: https://doi.org/10.1007/s13675-017-0081-7. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2192440621000915.

[56] A. R. Conn, K. Scheinberg, and L. N. Vicente, "Introduction to derivative-free optimization introduction," *Introduction to derivative-free optimization*, vol. 8, 2009.

[57] M. Powell, "Beyond symmetric broyden for updating quadratic models in minimization without derivatives," *Mathematical Programming*, vol. 138, no. 1, pp. 475–500, 2013.

[58] M. J. D. Powell, "On the use of quadratic models in unconstrained minimization without derivatives," *Optimization Methods and Software*, vol. 19, no. 3-4, pp. 399–411, 2004. DOI: 10.1080/10556780410001661450. eprint: https://doi.org/10.1080/10556780410001661450. [Online]. Available: https://doi.org/10.1080/10556780410001661450.

[59] M. J. Powell, "Least frobenius norm updating of quadratic models that satisfy interpolation conditions," *Mathematical Programming*, vol. 100, no. 1, pp. 183–215, 2004.

[60] ——, "Recent research at cambridge on radial basis functions," *New developments in approximation theory*, pp. 215–232, 1999.

[61] H.-M. Gutmann, "A radial basis function method for global optimization," *Journal of global optimization*, vol. 19, no. 3, pp. 201–227, 2001.

[62] S. C. Billups, J. Larson, and P. Graf, "Derivative-free optimization of expensive functions with computational error using weighted regression," *SIAM Journal on Optimization*, vol. 23, no. 1, pp. 27–53, 2013.

[63] R. G. Regis and C. A. Shoemaker, "A stochastic radial basis function method for the global optimization of expensive functions," *INFORMS Journal on Computing*, vol. 19, no. 4, pp. 497–509, 2007.

[64] S. M. Wild and C. Shoemaker, "Global convergence of radial basis function trust-region algorithms for derivative-free optimization," *siam REVIEW*, vol. 55, no. 2, pp. 349–371, 2013.

[65] R. Oeuvray and M. Bierlaire, "Boosters: A derivative-free algorithm based on radial basis functions," *International Journal of Modelling and Simulation*, vol. 29, no. 1, pp. 26–36, 2009.

[66] 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," *SIAM Journal on Optimization*, vol. 20, no. 1, pp. 387–415, 2009.

[67] A. R. Conn, N. I. Gould, and P. L. Toint, *Trust region methods*. Siam, 2000, vol. 1.

[68] C. Audet and J. E. Dennis Jr, "Pattern search algorithms for mixed variable programming," *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 573–594, 2001.

[69] M. A. Abramson, C. Audet, J. W. Chrissis, and J. G. Walston, "Mesh adaptive direct search algorithms for mixed variable optimization," *Optimization Letters*, vol. 3, no. 1, p. 35, 2009.

[70] M. Porcelli and P. L. Toint, "Bfo, a trainable derivative-free brute force optimizer for nonlinear bound-constrained optimization and equilibrium computations with continuous and discrete variables," *ACM Transactions on Mathematical Software (TOMS)*, vol. 44, no. 1, pp. 1–25, 2017.

[71] M. A. Abramson, C. Audet, and J. Dennis Jr, "Filter pattern search algorithms for mixed variable constrained optimization problems," Tech. Rep., 2004.

[72] C. Audet, S. Le Digabel, V. Rochon Montplaisir, and C. Tribes, "NOMAD version 4: Nonlinear optimization with the MADS algorithm," Les cahiers du GERAD, Tech. Rep. G-2021-23, 2021. [Online]. Available: http://www.optimization-online.org/DB_HTML/2021/04/8351.html.

[73] K. Rashid, S. Ambani, and E. Cetinkaya, "An adaptive multiquadric radial basis function method for expensive black-box mixed-integer nonlinear constrained optimization," *Engineering Optimization*, vol. 45, no. 2, pp. 185–206, 2013.

[74] J. Müller, C. A. Shoemaker, and R. Piché, "So-i: A surrogate model algorithm for expensive nonlinear integer programming problems including global optimization applications," *Journal of Global Optimization*, vol. 59, no. 4, pp. 865–889, 2014.

[75] ——, "So-mi: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems," *Computers & Operations Research*, vol. 40, no. 5, pp. 1383–1400, 2013.

[76] J. Müller, "Miso: Mixed-integer surrogate optimization framework," *Optimization and Engineering*, vol. 17, no. 1, pp. 177–203, 2016.

[77] K. Holmström, N.-H. Quttineh, and M. M. Edvall, "An adaptive radial basis algorithm (arbf) for expensive black-box mixed-integer constrained global optimization," *Optimization and Engineering*, vol. 9, no. 4, pp. 311–339, 2008.

[78] A. Costa and G. Nannicini, "Rbfopt: An open-source library for black-box optimization with costly function evaluations," *Mathematical Programming Computation*, vol. 10, no. 4, pp. 597–629, 2018.

[79] G. Liuzzi, S. Lucidi, and F. Rinaldi, "Derivative-free methods for bound constrained mixed-integer optimization," *Computational Optimization and Applications*, vol. 53, no. 2, pp. 505–526, 2012.

[80] ——, "Derivative-free methods for mixed-integer constrained optimization problems," *Journal of Optimization Theory and Applications*, vol. 164, no. 3, pp. 933–965, 2015.

[81] G. Liuzzi, S. Lucidi, and M. Sciandrone, "Sequential penalty derivative-free methods for nonlinear constrained optimization," *SIAM Journal on Optimization*, vol. 20, no. 5, pp. 2614–2635, 2010.

[82] B. W. Wah, Y. Chen, and T. Wang, "Simulated annealing with asymptotic convergence for nonlinear constrained optimization," *Journal of Global Optimization*, vol. 39, no. 1, pp. 1–37, 2007.

[83] S. Lucidi, V. Piccialli, and M. Sciandrone, "An algorithm model for mixed variable programming," *SIAM Journal on Optimization*, vol. 15, no. 4, pp. 1057–1084, 2005.

[84] J. Larson, M. Menickelly, and S. M. Wild, "Manifold sampling for \ell_1 nonconvex optimization," *SIAM Journal on Optimization*, vol. 26, no. 4, pp. 2540–2563, 2016.

[85] J. Larson, M. Menickelly, and B. Zhou, "Manifold sampling for optimizing nonsmooth nonconvex compositions," *SIAM Journal on Optimization*, vol. 31, no. 4, pp. 2638–2664, 2021.

[86] M. Menickelly and S. M. Wild, "Derivative-free robust optimization by outer approximations," *Mathematical Programming*, vol. 179, no. 1, pp. 157–193, 2020.

[87] J. J. Moré and D. C. Sorensen, "Computing a trust region step," *SIAM Journal on Scientific and Statistical Computing*, vol. 4, no. 3, pp. 553–572, 1983.

[88] P. Alberto, F. Nogueira, H. Rocha, and L. N. Vicente, "Pattern search methods for user-provided points: Application to molecular geometry problems," *SIAM Journal on Optimization*, vol. 14, no. 4, pp. 1216–1236, 2004.

[89] W. Karush, "Minima of functions of several variables with inequalities as side conditions," in *Traces and Emergence of Nonlinear Programming*, Springer, 2014, pp. 217–245.

[90] A. R. Conn, N. Gould, A. Sartenaer, and P. L. Toint, "Global convergence of a class of trust region algorithms for optimization using inexact projections on convex constraints," *SIAM Journal on Optimization*, vol. 3, no. 1, pp. 164–221, 1993.

[91] R. M. Lewis and V. Torczon, "A direct search approach to nonlinear programming problems using an augmented lagrangian method with explicit treatment of linear constraints," *Technical Report of the College of William and Mary*, pp. 1–25, 2010.

[92] V. Picheny, R. B. Gramacy, S. Wild, and S. L. Digabel, "Bayesian optimization under mixed constraints with a slack-variable augmented lagrangian," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 1443–1451.

[93] K. Murota, "Discrete convex analysis," *Mathematical Programming*, vol. 83, no. 1, pp. 313–371, 1998.

[94] M. A. Begen and M. Queyranne, "Appointment scheduling with discrete random durations," *Mathematics of Operations Research*, vol. 36, no. 2, pp. 240–257, 2011. DOI: 10.1287/moor.1110.0489. eprint: https://doi.org/10.1287/moor.1110.0489. [Online]. Available: https://doi.org/10.1287/moor.1110.0489.

[95] P. Zipkin, "On the structure of lost-sales inventory models," *Operations research*, vol. 56, no. 4, pp. 937–944, 2008.

[96] K. Murota, A. Shioura, and Z. Yang, "Computing a walrasian equilibrium in iterative auctions with multiple differentiated items," in *International Symposium on Algorithms and Computation*, Springer, 2013, pp. 468–478.

[97] K. Murota, "On steepest descent algorithms for discrete convex functions," *SIAM Journal on Optimization*, vol. 14, no. 3, pp. 699–707, 2004.

[98] A. Shioura, "Fast scaling algorithms for m-convex function minimization with application to the resource allocation problem," *Discrete Applied Mathematics*, vol. 134, no. 1, pp. 303–316, 2004, ISSN: 0166-218X. DOI: https://doi.org/10.1016/S0166-218X(03)00255-5. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166218X03002555.

[99] X. Zhou, "On the fenchel duality between strong convexity and lipschitz continuous gradient," *arXiv preprint arXiv:1803.06573*, 2018.

[100] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[101] P. D. Tao *et al.*, "The DC (Difference of Convex Functions) Programming and DCA Revisited with DC Models of Real World Nonconvex Optimization Problems.," *Annals of Operations Research*, vol. 133, no. 1-4, pp. 23–46, 2005.

[102] X. T. Vo, "Learning with sparsity and uncertainty by difference of convex functions optimization," Ph.D. dissertation, Université de Lorraine, 2015.

[103] H. A. Le Thi and T. P. Dinh, "DC programming and DCA: thirty years of developments," *Mathematical Programming*, vol. 169, no. 1, pp. 5–68, 2018.

[104] S. Fujishige, *Submodular functions and optimization*. Elsevier, 2005.

[105] W. E. Hart, J.-P. Watson, and D. L. Woodruff, "Pyomo: Modeling and solving mathematical programs in python," *Mathematical Programming Computation*, vol. 3, no. 3, pp. 219–260, 2011.

[106] M. L. Bynum, G. A. Hackebeil, W. E. Hart, *et al.*, *Pyomo—Optimization Modeling in Python*. Springer Nature, 2021, vol. 67.

[107] W. Huyer and A. Neumaier, "Snobfit–stable noisy optimization by branch and fit," *ACM Transactions on Mathematical Software (TOMS)*, vol. 35, no. 2, pp. 1–25, 2008.

[108] N. I. Gould, D. Orban, and P. L. Toint, "Cutest: A constrained and unconstrained testing environment with safe threads for mathematical optimization," *Computational optimization and applications*, vol. 60, no. 3, pp. 545–557, 2015.

[109] L. Lukšan and J. Vlcek, "Test problems for nonsmooth unconstrained and linearly constrained optimization," *Technická zpráva*, vol. 798, 2000.