UCLouvain

École polytechnique de Louvain

A Derivative-Free Trust-Region Method Based on Finite-Difference Gradient Approximations

Author: **Dânâ DAVAR**
Supervisor: **Geovani NUNES GRAPIGLIA**
Readers: **Gianluca BIANCHIN, François GLINEUR**
Academic year 2022–2023
Master [120] in Mathematical Engineering

Abstract

This work presents a derivative-free trust-region method, based on finite-difference gradient approximations, for smooth convexly constrained optimization problems.

Under reasonable assumptions, it is shown that the method needs at most $\mathcal{O}(n\varepsilon^{-2})$ function evaluations to obtain an ε-approximate stationary point, where n is the problem dimension. Therefore, the worst-case complexity bound depends only linearly on n. This result represents a significant improvement with respect to some other state-of-the-art methods, and is confirmed by numerical experiments that show the relative efficiency of the new method.

Acknowledgements

First, I would like to express my sincere gratitude to Prof. Geovani Nunes Grapiglia. I am grateful for the time he spent on this work with me, and for his availability to answer all my questions during the year. I am thankful for his proactivity, for the quality of the references he shared with me and for his management of our work over the last months. It has been a real pleasure to work together, and thank him for his support and his kindness.

I also would like to say warmly thank you to the readers and jury members of this work, Prof. Gianluca Bianchin and Prof. François Glineur. I sincerely thank them for accepting to spend time and energy to evaluate our work.

Many thanks also to my school mathematics teacher, Mr. Pierre Bolly. I am grateful for his teaching and advice, which were helpful during my five years studies at the university.

Finally, I would like to say thank you to my mother and my father. Thank you for *everything* they have done for me since my birth. To both of them, I dedicate this work.

Contents

List of Notations

Δ	Trust-region radius
η	Trust-region minimum successful ratio
$\langle \cdot, \cdot \rangle$	Scalar product between two vectors
$\| \cdot \|$	Euclidean norm
$\mathbb{R}$	Set of real numbers
$\nabla^2 f(x)$	Hessian of f evaluated at x
$\nabla f(x)$	Gradient of f evaluated at x
Ω	Convex domain of the optimization problem
π	Stationarity measure at point x
ρ	Trust-region ratio
P_Ω	Orthogonal projection on the convex domain Ω
ε	Tolerance to a stationary point
B	Matrix approximation to $\nabla^2 f(x)$
e_j	j^{th} canonical vector
f	Objective function
$f(x)$	Objective function evaluated at x

$FE(\varepsilon)$ Number of function evaluations required to reach an ε-approximate stationary point

g Vector approximation to $\nabla f(x)$

h Step size of the gradient approximation

k Iteration index

L Lipschitz constant

m Quadratic model of the objective function

n Problem dimension

$T(\varepsilon)$ Number of iterations required to reach an ε-approximate stationary point

List of Abbreviations

FFD	Forward Finite-Difference
BFD	Backward Finite-Difference
CFD	Central Finite-Difference
ODE	Ordinary Differential Equation
TRFD	The new method: a Trust-Region method based on Finite-Difference gradient approximations
DFQRM	Derivative-Free Quadratic Regularization Method
BFGS	Broyden-Fletcher-Goldfarb-Shanno update for Hessian approximations
DFO	Derivative-Free Optimization

Chapter 1

Introduction

1.1 Motivation

Many practical applications require the solution of nonlinear optimization problems of the form

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && x \in \Omega, \end{aligned} \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a differentiable function and $\Omega \subset \mathbb{R}^n$ is a nonempty closed convex set. In order to address (1.1), iterative methods are the only choice to solve the problem efficiently, one example of which is illustrated in Figure 1.1. Those methods start with one initial guess of the minimizer and generate a sequence of better approximated solutions, each obtained on the basis of the previous one. To do so, those methods need to rely on the slope of the function at those points, i.e., the gradient of the objective function f.

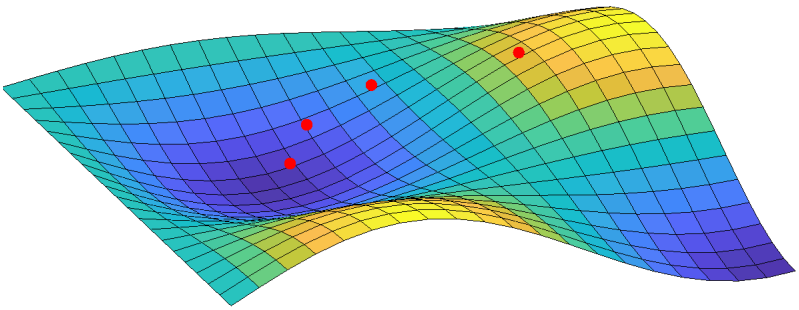


Figure 1.1: An iterative optimization method

However, in many applications, no information is accessible about the gradient. It is common, e.g., when function values come from computer simulations realized through a black-box software. In such case, although the objective function is differentiable, the only information we obtain are pairs $\{x, f(x)\}$. Typical examples are present in aerodynamic shape optimization [1], optimization of cardiovascular geometries [2, 3] or tuning of algorithmic parameters [4]. Therefore, in order to minimize f in such scenarios, we need methods that rely only on the function values $f(x)$. The field of optimization using such methods is called *Derivative-Free Optimization* (DFO).

1.2 Historical Overview

As mentioned in [5], the application of DFO methods to deterministic problems can be separated in several classes. For each of these, we can use local DFO methods which, in the best case, can guarantee the convergence to a local stationary point of f. Regarding global stationary points, deterministic DFO methods able to generate a global minimizer for general nonconvex functions do not exist.

Nowadays, the set of local DFO methods can be divided in three main classes: direct-search methods, model-based trust-region methods, and finite-difference based methods. The first two, i.e., direct-search and model-based trust-region, do not rely on approximated gradients of f, while the class of finite-difference based methods does. All three of them are derivative-free methods that can be used to solve (1.1). In Section 1.2.1, we review the class of direct-search methods, while in Section 1.2.2 we mention the class of model-based trust-region methods. Then, Section 1.2.3 deals with finite-difference based methods. Finally, in Section 1.2.4, we briefly cite some randomized methods used in DFO.

1.2.1 Direct-Search Methods

Direct-search methods are often used in practice. They try several points located along specific directions around the current iterate, looking for one where the function value is lower than that at the current point. As mentioned by Nocedal and J. Wright [6], if such point is obtained, we take it as the new iterate and repeat the process, possibly with a new set of directions to explore. If no satisfying point is discovered, then we reduce the step length and possibly adjust the directions.

In this class, one efficient strategy is proposed by pattern-search methods. At the

current iteration, those methods create a set of orthogonal directions of a given size. Therefore, the current iterate x_k is framed by candidate points, with potential lower function values, that will be evaluated. If one of them has a function value significantly lower than that of the current point, we set it as the new iterate and the frame will be centered at this new point. At each iteration, the set of directions and the size of those directions can be adapted. If no point produces a sufficient decrease, then we set $x_{k+1} = x_k$ and the size of the frame is reduced. Figure 1.2 illustrates this method. The current point is assigned to the red dot, while the next iterate is in blue.

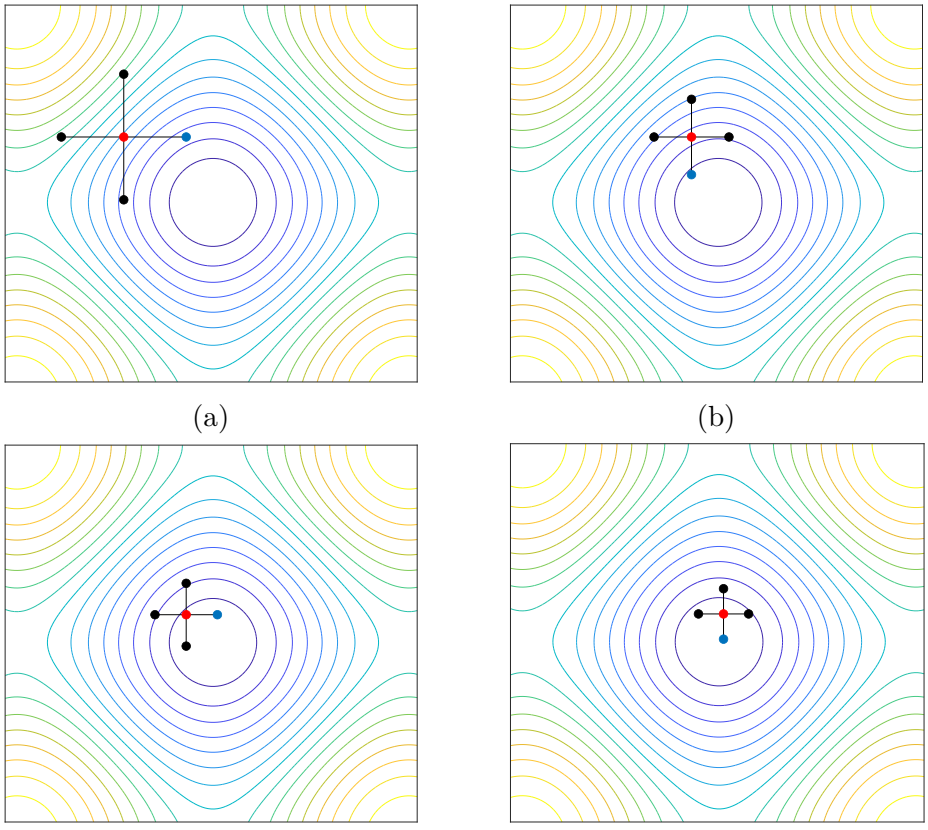


Figure 1.2: A direct-search method

Although direct-search methods can produce relatively good results by using function evaluations, it could be more relevant to make a different use of those values, with the aim of obtaining more ambitious results.

1.2.2 Model-Based Trust-Region Methods

Instead of using only function evaluations, another approach in DFO is to build a model m_k around the current iterate. This approach allows to have a good approximation of the shape of f around $f(x_k)$. Currently, to approximate the objective function properly at a current point and get a good new iterate, quadratic models are presented as one of the most efficient ways to choose. They have been used for several decades and are still of interest. However, those models are based on Taylor developments and therefore rely on the gradient of f. Thus, one could decide to build the derivative-free quadratic model m_k as an interpolation model, based on several well chosen points. Such methods are called *Model-Based* and have also been used for a long time. Very recently [7], Hough and Roberts implemented a model-based trust-region method for convexly constrained problems. They prove the global convergence and provide a worst-case complexity of $\mathcal{O}(n^3\varepsilon^{-2})$ and $\mathcal{O}(n^7\varepsilon^{-2})$ function evaluations, for linear and composite linear interpolation models, respectively, in order to reach some ε-approximate stationary point. In case of unconstrained problems, Garmanjani, Júdice and Vicente [8] derived a bound of $\mathcal{O}(n^2\varepsilon^{-2})$ function evaluations for model-based trust-region methods, in 2016. In practice, model-based methods are applied with a trust-region method, as the objective function can potentially be nonconvex. A nonconvex function could locally lead to a nonconvex quadratic model, which has no minimizer if there is no constraint on the minimization of m_k. Therefore, working through a trust-region method would allow to add restrictions to this minimization and give rise to a solution.

The first person to propose a derivative-free trust-region method was D. Winfield [9, 10]. In [10], he presents a quadratic model-based method obtained by interpolation of points, for which the model is minimized over a "constraining region of validity to locate the next trial point", as he says. This DFO method was the first in which the concept of a *trust*-region was mentioned. For the general construction of the model, let us consider the set of points

$$Y = \{y_1, y_2, ..., y_q\} \quad \text{with} \quad y_i \in \Omega, \quad i = 1, ...q,$$

for some $q \geq 1$. Then, we can build the quadratic model

$$m_k(d) = f(x_k) + \langle g_k, d\rangle + \frac{1}{2}\langle B_k d, d\rangle,$$

where g_k and B_k are approximations of $\nabla f(x_k)$ and $\nabla^2 f(x_k)$, respectively. Both have to be determined such that the interpolation conditions are satisfied, i.e.,

$$m_k(y_i) = f(y_i), \quad i = 1, ...q.$$

Moreover, the model m_k should also represent sufficiently well f in a fully-linear sense [5]. A function $m : \mathbb{R}^n \to \mathbb{R}$ is said to be a κ-fully linear model of f on $B(x_k; \Delta_k) = \{y : \|x_k - y\| \leq \Delta_k\}$ if

$$\begin{aligned} |f(x_k + d) - m_k(d)| &\leq \kappa_{ef}\Delta_k^2, \quad \forall d \in B(0; \Delta_k), \\ \|\nabla f(x_k + d) - \nabla m_k(d)\| &\leq \kappa_{eg}\Delta_k, \quad \forall d \in B(0; \Delta_k), \end{aligned}$$

for some $\kappa = (\kappa_{ef}, \kappa_{eg})$. Those conditions being satisfied, we then need to solve the sub-problem

$$\begin{aligned} \min_d \quad & m_k(d) = f(x_k) + \langle g_k, d\rangle + \frac{1}{2}\langle B_k d, d\rangle \\ \text{s.t.} \quad & \|d\| \leq \Delta_k. \end{aligned}$$

To judge the quality of the approximate solution obtained d_k, we use a condition expressed as (1.2), where $\eta \in (0, 1)$. The ratio ρ_k represents the decrease occurred in the objective function, with respect to the decrease produced in the model. If (1.2) holds, then we say that the iteration is successful and set $x_{k+1} = x_k + d_k$. In such case, there is a good match between m_k and f in the sense that, inside the trust-region, m_k is a good approximation of f. Therefore, it could be good to increase the trust-region radius for the next iterate. Otherwise, if $\rho_k < \eta$, then we should shrink the radius or update the set Y to find another approximate solution d_k, which should produce a higher ratio ρ_k. Figure 1.3 illustrates a trust-region step. It shows that the minimization of the model should be supported by a constraint, saying inside which region we should *trust* m_k as a good approximation of the objective function f.

$$\rho_k \coloneqq \frac{f(x_k) - f(x_k + d_k)}{m_k(0) - m_k(d_k)} \geq \eta. \tag{1.2}$$

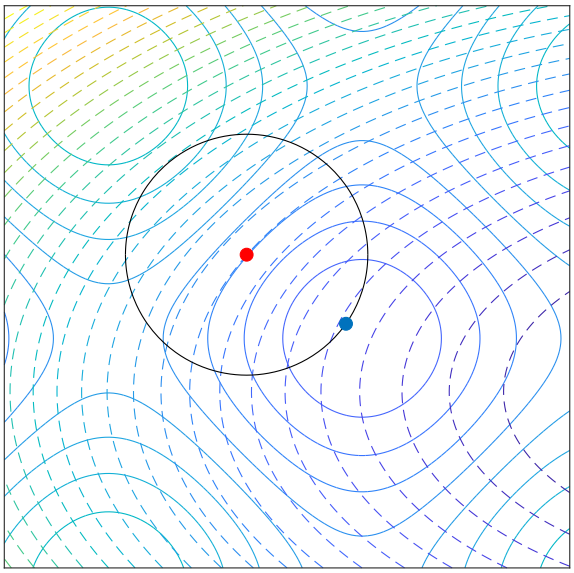


Figure 1.3: A trust-region step

This second class we have just presented has been widely used for the last decades in DFO, mostly because of his low usage of function evaluations required at each iteration. It is the main reason why finite-difference based methods have been almost neglected during this time, as it can be expensive to approximate the gradient at each iteration, especially for large-scale problems. However, this last main class of DFO methods, which is presented in the following section, has recently become very active and recent papers show remarkable results. This lets know that DFO methods based on finite-differences deserve further examinations.

1.2.3 Finite-Difference Based Methods

This last main class of DFO methods does rely on approximated gradients. The idea is to estimate sufficiently well the gradient of f around the current point, in order to replace it inside a derivative-based model m_k. Therefore, the method builds a model based on approximated gradients by means of finite-differences. Very recently [11], Shi et al. demonstrated encouraging empirical results, where they show that derivative-free methods based on finite-differences are able to be competitive with respect to other state-of-the-art DFO methods, and here we review the foundations of finite-difference based methods.

There exist three main ways to compute the j^{th} component of $\nabla f(x_k)$ by means of a finite-difference:

Forward Finite-Difference (FFD)

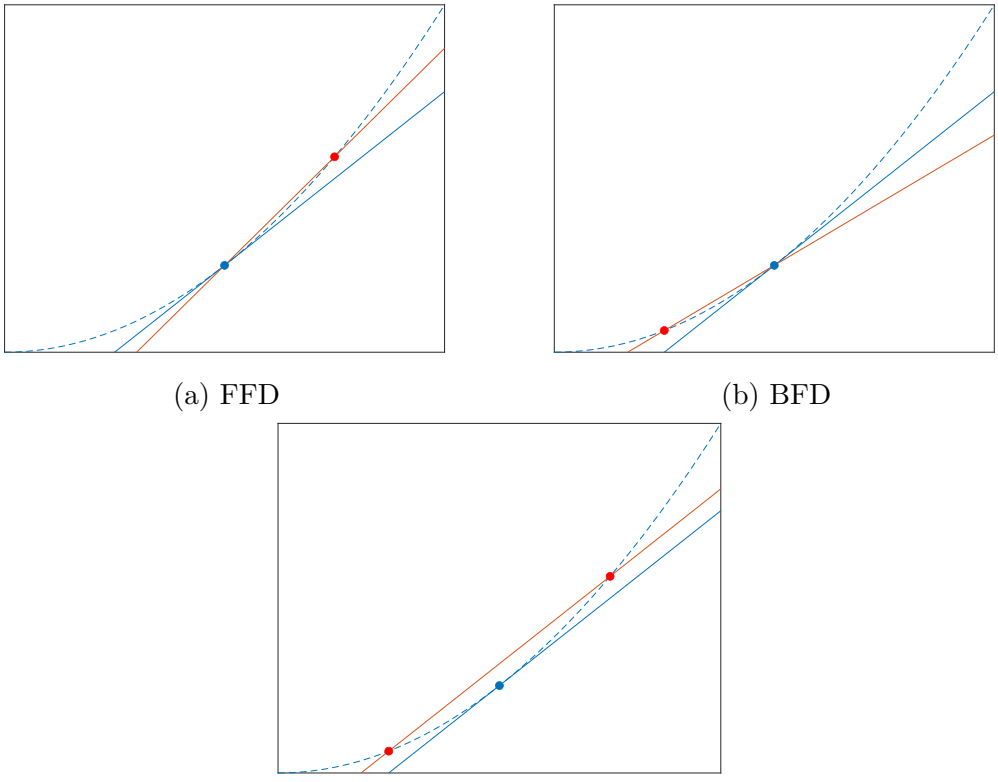
$$[\nabla f(x_k)]_j \simeq [g_k]_j = \frac{f(x_k + he_j) - f(x_k)}{h},$$

Backward Finite-Difference (BFD)

$$[\nabla f(x_k)]_j \simeq [g_k]_j = \frac{f(x_k) - f(x_k - he_j)}{h},$$

and Central Finite-Difference (CFD)

$$[\nabla f(x_k)]_j \simeq [g_k]_j = \frac{f(x_k + he_j) - f(x_k - he_j)}{2h}.$$



(a) FFD (b) BFD

(c) CFD

Figure 1.4: Different Finite-Differences

Although it is common to build the vector $g_k \in \mathbb{R}^n$ using a same type of finite-difference for all its components, there exist situations where we should use a combination of them. Indeed, if function evaluations are not available in forward direction for some components, it could be relevant to choose a BFD for these, while a FFD

could be used for the other components of g_k. However, although the three ways were available at the current iterate, one could find more accurate to use one of the three, rather than others. Indeed, Figure 1.4 gives the geometric interpretation of the three approximations, showing all approximations can not be of same accuracy around a same point of f.

Moreover, whatever the approximation chosen, the size of h will be of real matter. It should not be too large, otherwise the approximation will be poor, but it should also not be too small, as it could lead to poor approximations as well, due to important numerical errors. Those two problems are common in numerical methods when approximating a derivative and lead us to the two types of errors that can occur, namely, the truncation error, E_t, and the round-off error, E_r. Here, we illustrate both of them using a FFD approximation for the j^{th} component of $\nabla f(x)$.

Assuming exact function evaluations, the truncation error E_t is the difference between the derivative and its finite-difference approximation. By the mean value theorem, there exists some $\theta \in (0, 1)$ such that

$$[\nabla f(x)]_j = \frac{f(x + he_j) - f(x)}{h} - \frac{h}{2}\frac{\partial^2 f(x + \theta h e_j)}{\partial x_j^2}.$$

Defining C as a bound on the second-order derivative, we get

$$|E_t| \leq \frac{h}{2}C.$$

Therefore, the truncation error depends linearly to h and the approximation will be poor if h is too large. However, we should also consider inexact function evaluations, which can occur due to a limited number of digits available when evaluating f. Assuming the error generated is bounded by some ϵ, the absolute value of the round-off error is expressed as

$$|E_r| \leq \frac{2\epsilon}{h},$$

because of the term

$$\frac{f(x + he_j) - f(x)}{h},$$

which now contains inexact function values. This time, the resulting error depends inversely on h. Indeed, for too small values of h, subtraction of similar numbers can appear. This operation can be badly evaluated because of the limited number of digits available in each term, and imply a huge error. Finally, the total error can be

bounded by

$$|E| \leq \frac{2\epsilon}{h} + \frac{h}{2}C.$$

Those two different dependencies on h imply there exists some trade-off between both sources of errors, as shown in Figure 1.5. This optimal value can be obtained by equalizing the derivative of $|E|$ with respect to h to 0.

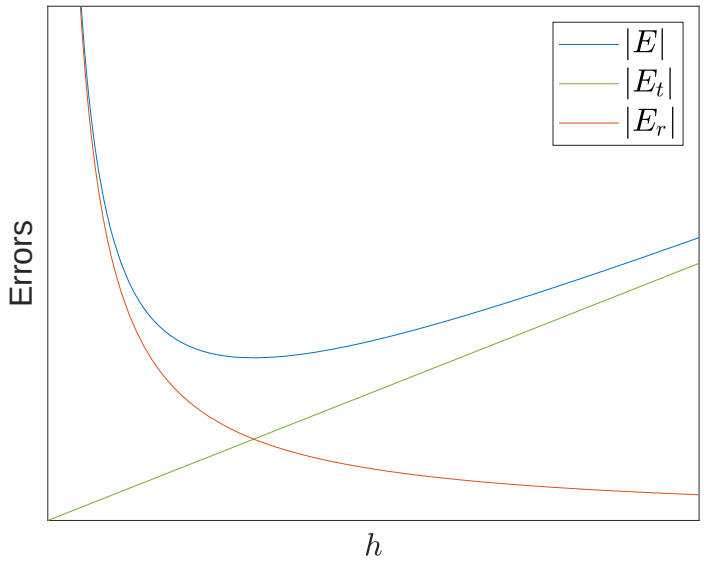


Figure 1.5: Truncation and round-off errors

Managing these errors, added to the cost of evaluations required to produce finite-differences, were the principal reasons why this last main class of DFO methods was neglected. However, the recent empirical experiments of Shi et al. [11] and the quadratic regularization method proposed by Grapiglia [12] prove there exist ways to tackle the common problems occurring in finite-difference based methods, especially concerning the number of function evaluations, which is one of the main subjects of this thesis.

1.2.4 Randomized Methods

Randomized methods usually have interesting theoretical properties and are used in practice for large-scale problems. They have the particularity to potentially reach the global minimum of an objective function containing several local minima. Two classical random search methods can be cited, as pure random search, whose global convergence is discussed by Zhigljavsky [13], and Nesterov random search [14]. However, randomized versions of the precedent classes of DFO methods also exist and

can be used for solving deterministic problems, as randomized direct-search methods [15] or randomized trust-region methods [16]. However, we focus on deterministic methods for solving deterministic problems, therefore we will not go further on randomized methods in this work. Nevertheless, the class of meta-heuristics methods is also of interest. Genetic algorithms and simulated annealing are popular heuristics methods in DFO, and are generally used in practice for stochastic versions of (1.1).

1.3 Contribution of this Work

In this thesis, we developed a derivative-free trust-region method, based on finite-difference gradient approximations, for smooth convexly constrained optimization problems. Our algorithm generates a sequence of sub-problems for which a quadratic model of the objective function is obtained around the current point. Here, this model relies on approximated gradients of the objective function, which is obtained using forward or backward finite-differences, while the Hessian is approximated and updated by the *Broyden-Fletcher-Goldfarb-Shanno* (BFGS) formula. Once the model is built, the solution of each sub-problem is approximately computed inside a region where we trust the model, in other words the *trust*-region, by means of an inner solver.

In Chapter 2, we present the classes of DFO algorithms that built our new method and mention the two main papers on which this work is based. By combining the hypotheses used in [17] by Karas et al. and the technique chosen in [12] by Grapiglia, we obtain the theoretical foundations required to prove the efficiency of the new method. Chapter 3 presents TRFD, the new method we have developed. First, it is presented for general nonempty closed convex sets, where function evaluations can be computed outside the domain, before presenting the particular bound constraints case, for which no function evaluation is allowed outside the feasible set. The latter case imposes to compute the gradient approximation using either a forward or backward finite-difference for each component of the vector. The choice depends on whether function evaluations are available in forward or backward directions. Moreover, for both variants of TRFD, a theoretical analysis is provided and demonstrates the efficiency of both versions to find an ε-approximate stationary point, using $\mathcal{O}(n\varepsilon^{-2})$ function evaluations. To our knowledge, TRFD is the first derivative-free trust-region method producing a linear dependence to n for smooth convexly constrained problems. Finally, Chapter 4 gives some numeri-

cal results for the application of the new method. Two implementations of TRFD are provided. One is applied to benchmark problems, while the other is used for a box-constrained problem. Results for the benchmark problems are based on 102 different unconstrained problems provided by Andrei [18]. The figures shown are *data profiles*, which are a specific way to show the efficiency of a method as a cumulative function of two characteristics: the percentage of problems solved, with respect to the number of function evaluations needed to solve them. Therefore, data profiles are provided and show the competitiveness of TRFD. The comparison is done with respect to DFQRM, which is the method proposed by Grapiglia in [12]. Regarding the results from the box-constrained case, they illustrate the application of the new method to a SIR model problem. Given a synthetic data set, the purpose of the application is to recover the optimal parameters β and γ producing the best approximation curves of the dynamical system. In this application, we show the relative competition between TRFD and the FMINCON Matlab function.

Chapter 2

Background

In this chapter, we describe in details the DFO methods related to our work. Section 2.1 deals with derivative-based trust-region methods. Section 2.2 presents a general derivative-free trust-region method for convexly constrained problems, while Section 2.3 describes a derivative-free quadratic regularization method based on finite-difference gradient approximations. Finally, 2.4 introduces the novelty presented in this work.

2.1 Derivative-Based Trust-Region Methods

2.1.1 Unconstrained Problems

If the gradient of f is available, then derivative-based models such as those built on Taylor developments should be used. In the k^{th} iteration, an efficient way to compute the step $d_k = x_{k+1} - x_k$ is to have a quadratic model m_k around the current iterate x_k. Moreover, as mentioned before, a good practice would be to work through a trust-region method. In case of derivative-based trust-region methods, the model m_k is defined as

$$m_k(d) = f(x_k) + \langle \nabla f(x_k), d \rangle + \frac{1}{2} \langle \nabla^2 f(x_k) d, d \rangle,$$

where $\nabla f(x_k)$ is the gradient of the objective function, and $\nabla^2 f(x_k)$ is its Hessian, both evaluated at the current point x_k. Once the method is built, we must solve

the derivative-based trust-region sub-problem expressed as

$$\begin{aligned} \min_{d} \quad & m_k(d) = f(x_k) + \langle \nabla f(x_k), d \rangle + \frac{1}{2} \langle \nabla^2 f(x_k) d, d \rangle \\ \text{s.t.} \quad & \|d\| \leq \Delta_k, \end{aligned} \tag{2.1}$$

in order to obtain the solution d_k. Such direction d_k may be any approximate solution of (2.1), as long as it produces a sufficient decrease in the model, in the sense of

$$m_k(0) - m_k(d_k) \geq c_1 \|\nabla m_k(0)\| \min \left\{ \frac{\|\nabla m_k(0)\|}{1 + \|\nabla^2 m_k(0)\|}, \Delta_k \right\},$$

for some constant c_1. Moreover, d_k should also satisfy condition (1.2), stated in Section 1.2.2 for model-based trust-region methods. If the decrease occurred by d_k implies $\rho_k \geq \eta$, it means there is a good match between the model and the objective function inside the trust-region. Therefore, it should be safe and a good idea to extend the size of the trust-region for the next iteration [6]. In case that the decrease is not sufficient, we reduce the size of the trust-region and reiterate the sub-problem.

In order to solve sub-problem (2.1), several approaches exist. One classical way to solve it approximately is by means of the Cauchy point, i.e., the minimizer of m_k lying on the steepest descent $-g_k$, constrained to the trust-region radius. However, we can be more ambitious and obtain a better approximation. Here, we mention the *Newton* method, which is an iterative process trying to approach the exact solution of (2.1). It is based on the idea that the solution can be obtained by adding a multiple of the identity to the Hessian of the model. Indeed, in Theorem 2.1.1, it is claimed that the solution d_k of (2.1) satisfies the equality

$$(\nabla^2 f(x_k) + \lambda I) d_k = -\nabla f(x_k),$$

for some $\lambda \geq 0$ and positive semi-definite matrix $(\nabla^2 f(x_k) + \lambda I)$. A formal proof is given by Nocedal and Wright [6].

Theorem 2.1.1. *The vector d_k is a global solution of the trust-region sub-problem*

$$\begin{aligned} \min_{d} \quad & m_k(d) = f(x_k) + \langle \nabla f(x_k), d \rangle + \frac{1}{2} \langle \nabla^2 f(x_k) d, d \rangle \\ \textit{s.t.} \quad & \|d\| \leq \Delta_k, \end{aligned}$$

if and only if d_k *is feasible and there is a scalar* $\lambda \geq 0$ *such that the following conditions are satisfied:*

$$(\nabla^2 f(x_k) + \lambda I)d_k = -\nabla f(x_k), \tag{2.2}$$

$$\lambda(\Delta_k - \|d_k\|) = 0, \tag{2.3}$$

$$(\nabla^2 f(x_k) + \lambda I) \quad \textit{is positive semi-definite.} \tag{2.4}$$

Equation (2.3) is an additional condition. It states that the solution d_k lies either inside, or strictly on the radius Δ_k. On one hand, if the solution lies inside the trust-region, then

$$d_k = -\nabla^2 f(x_k)^{-1} \nabla f(x_k), \tag{2.5}$$

and $\lambda = 0$, since $\|d_k\| < \Delta_k$. On the other hand, if the solution is strictly localized on the radius, then the point $d = -\nabla^2 f(x_k)^{-1} \nabla f(x_k)$ lies outside the trust-region. In such case, $\|d_k\| = \Delta_k$ and λ must take some positive value to satisfy (2.2). Therefore, the idea is to use an iterative method to find such $\lambda \geq 0$ for a solution d_k whose norm is equal to the trust-region radius. Actually, the method is only used for $\lambda > 0$, as the case $\lambda = 0$ corresponds to the first scenario (2.5), for which the solution d_k is already known. Therefore, for $\lambda > 0$, as explained in [6], we define the function

$$d(\lambda) = -(\nabla^2 f(x_k) + \lambda I)^{-1} \nabla f(x_k),$$

for λ sufficiently large so that $\nabla^2 f(x_k) + \lambda I \succ 0$, and seek a value $\lambda_* > 0$ such that

$$\|d(\lambda_*)\| = \Delta_k. \tag{2.6}$$

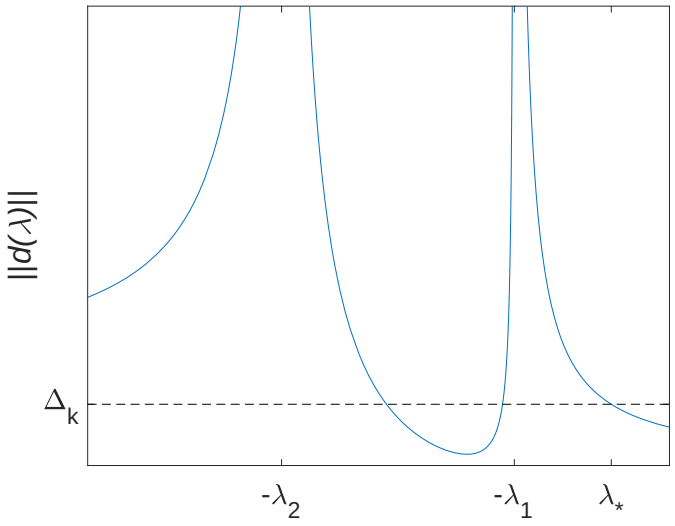


Figure 2.1: $\|d(\lambda)\|$ (inspired from [6])

For this purpose, let $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ be the eigenvalues of the matrix $\nabla^2 f(x_k)$ and Q be the orthogonal matrix composed of the eigenvectors of $\nabla^2 f(x_k)$. Thanks to the eigendecomposition of the matrix, the following expression can be obtained,

$$\|d(\lambda)\|^2 = \sum_{j=1}^{n} \frac{(q_j^T g)^2}{(\lambda_j + \lambda)^2}, \tag{2.7}$$

where q_j is the j^{th} eigenvector of $\nabla^2 f(x_k)$. Equation (2.7) tells us that, for $\lambda > -\lambda_1$, $\|d(\lambda)\|$ is a positive non-increasing function. Therefore, as shown on Figure 2.1, we can reach some value λ_* such that $\|d(\lambda_*)\| = \Delta_k$ on the interval $\lambda \in (-\lambda_1, \infty)$. Solving equation (2.6) is equivalent to finding the root of the function

$$\phi_1(\lambda) = \|d(\lambda)\| - \Delta_k = 0. \tag{2.8}$$

However, we see that some troubles can appear when λ is greater but close to $-\lambda_1$. Therefore, another expression for solving (2.8) is preferred, e.g., as dividing by $\|d(\lambda)\| \Delta_k$, which gives the following expression

$$\phi_2(\lambda) = \frac{1}{\Delta_k} - \frac{1}{\|d(\lambda)\|} = 0,$$

which makes $\phi_2(\lambda)$ almost linear near of $-\lambda_1$. Then, in order to find the root of the latter function, Newton-Raphson's method can be applied to $\phi_2(\lambda)$, which gives

$$\lambda^{(l+1)} = \lambda^{(l)} - \frac{\phi_2(\lambda^{(l)})}{\phi_2'(\lambda^{(l)})}. \tag{2.9}$$

By some manipulations based on (2.9), an iterative algorithm can be implemented as proposed in Algorithm 2.1. However, we should be careful with this method. During the iterations, we can reach values of λ for which the matrix $\nabla^2 f(x_k) + \lambda I$ is not positive semi-definite. This implies that it can not be factorized as the product RR^T, which is required in the following algorithm. That is why safeties are taken in Step 0 and Step 1, as proposed by Conn et al. [19], with lower and higher bounds on λ, denoted as λ^L and λ^U respectively, in which we denote $\nabla f(x_k)$ by v and $\nabla^2 f(x_k)$ by H for clarity in the expressions.

$$\lambda^L = \max\left\{0, -\min_i\{H_{i,i}\}, \frac{\|v\|}{\Delta_k} - \min\left\{\max_i\left\{H_{i,i} + \sum_{j\neq i}|H_{i,j}|\right\}, \|H\|_F, \|H\|_\infty\right\}\right\},$$

$$\lambda^U = \max\left\{0, \frac{\|v\|}{\Delta_k} + \min\left\{\max_i\left\{-H_{i,i} + \sum_{j\neq i}|H_{i,j}|\right\}, \|H\|_F, \|H\|_\infty\right\}\right\}.$$

Algorithm 2.1 Inner-Solver (Newton Method)

Step 0 Set $\Delta_0 > 0$, $\lambda^{(0)} \in (\lambda^L, \lambda^U)$ and $i \coloneqq 0$.

Step 1 If $\lambda^{(i)} > \max\{0, -\lambda_1\}$, factorize

$$\nabla^2 f(x_k) + \lambda^{(i)} I = R^T R,$$

and solve

$$R^T R p_i = -\nabla f(x_k), \quad R^T q_i = p_i.$$

Step 2 Set

$$\lambda^{(i+1)} = \lambda^{(i)} + \left(\frac{\|p_i\|}{\|q_i\|}\right)^2 \left(\frac{\|p_i\| - \Delta_k}{\Delta_k}\right).$$

Step 3 Set $i \coloneqq i + 1$ and go to Step 1.

Typically, only four or five iterations are needed in practice to get $\lambda^{(i+1)} = \lambda^{(i)}$ and to stop the solver. This algorithm is an inner-solver that enables to solve the sub-problem (2.1), and is the one we chose in our new method, when applied to unconstrained problems.

2.1.2 Convexly Constrained Problems

In the specific case of derivative-based trust-region methods constrained to convex domains, a new constraint is added to the derivative-based trust-region sub-problem (2.1), which now takes the following form

$$\begin{aligned} \min_d \quad & m_k(d) = f(x_k) + \langle \nabla f(x_k), d\rangle + \frac{1}{2}\langle \nabla^2 f(x_k) d, d\rangle \\ \text{s.t.} \quad & \|d\| \leq \Delta_k, \\ & x_k + d \in \Omega. \end{aligned}$$

Consequently, a new inner-solver is required, but above all a new criticality measure is needed to minimize f inside the convex domain Ω. As shown in [19], we can define the measure

$$\chi(x) = |\min_{\substack{x+d\in\Omega \\ \|d\|\leq 1}} \langle \nabla f(x), d\rangle|, \tag{2.10}$$

which represents the norm of the negative gradient when d can be in the opposite direction of the gradient. Of course, in the unconstrained case, definition (2.10) corresponds to the intuitive criticality measure

$$\chi(x) = \|\nabla f(x)\|\,. \tag{2.11}$$

However, (2.10) is not the only way to define the stationarity of x. Indeed, still in the case of convexly constrained problems, it can be good to reformulate (2.10) as

$$\chi(x) = \|x - P_\Omega(x - \nabla f(x))\|\,. \tag{2.12}$$

This new definition has the advantage to conserve the continuity of χ and to still preserve a geometric interpretation. Indeed, if the point $x - \nabla f(x)$ had to belong to Ω, then we would find (2.11). If not, then (2.12) would represent the distance between x and $P_\Omega(x - \nabla f(x))$, the latter lying on the boundary of Ω.

This last section about derivative-based optimization introduces now to derivative-*free* optimization, for which we highlight the two main papers that inspired this work.

2.2 Derivative-Free Trust-Region Methods

In [17], Karas et al. propose a general derivative-free trust-region algorithm for minimizing a smooth function within a closed convex set Ω

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & x \in \Omega. \end{aligned} \tag{2.13}$$

The method generates a sequence of approximated solutions of quadratic constrained

sub-problems

$$\begin{aligned} \min_{d} \quad & m_k(d) = f(x_k) + \langle g_k, d\rangle + \frac{1}{2}\langle B_k d, d\rangle \\ \text{s.t.} \quad & \|d\| \leq \Delta_k \\ & x_k + d \in \Omega, \end{aligned} \tag{2.14}$$

where g_k is an approximation to $\nabla f(x_k)$ and B_k is an approximation to $\nabla^2 f(x_k)$. In order to prove the global convergence of the general method, they make the following assumptions.

A1 *The gradient of f is L-Lipschitz continuous, i.e.,*

$$\|\nabla f(y) - \nabla f(x)\| \leq L\,\|y - x\| \quad \forall x, y \in \Omega.$$

A2 *There exists $f_{low} \in \mathbb{R}$ such that $f(x) \geq f_{low}$ $\forall x \in \Omega$.*
A3 *There exists a constant $c_2 > 0$ such that*

$$\|\nabla f(x_k) - g_k\| \leq c_2 \Delta_k \quad \forall k \geq 1. \tag{2.15}$$

A4 *The matrices B_k are uniformly bounded, that is, there exists a constant M such that $\|B_k\| \leq M$ $\forall k \geq 1$.*

Moreover, they assume that the approximate solution d_k of (2.14) satisfies the following sufficient decrease condition

$$m_k(0) - m_k(d_k) \geq c_1 \pi_k \min\left\{\frac{\pi_k}{1 + \|B_k\|}, \Delta_k\right\}, \tag{2.16}$$

where c_1 is a constant independent of k, and π_k is a stationarity measure defined as

$$\pi_k := \|x_k - P_\Omega(x_k - g_k)\|.$$

The approximate solution d_k can be computed by means of any inner-solver, as Algorithm 2.2 which is a revisited version of the FISTA algorithm, see Section 5 in [7]. It allows to minimize approximately the model m_k, constrained to the intersection of convex domains, using Dijkstra's algorithm for the projections. Therefore, this inner-solver allows to solve a sequence of sub-problems such as (2.14), and is the one we use in the application of our new method to a box-constrained problem in Section 4.2, in which the stopping conditions are also mentioned.

Algorithm 2.2 Inner-Solver (Revisited FISTA)

Step 0 Given convex set $\Omega \subset \mathbb{R}^n$, starting point $x_0 \in \Omega$, model gradient g_0, approximated Hessian B_k, and trust-region radius Δ_k. Set $y_0 = x_0$, $L = \|B_k\|$ if $B_k \neq 0$ (or $L = 1$ otherwise), and $t_0 = 1$.

Step 1.1 Set $j \coloneqq 0$.

Step 1.2 Project the gradient step onto $D \coloneqq C \cap B(x_0, \Delta_k)$, set $x_{j+1} = P_D(y_j - \frac{1}{L} g_j)$ and go to Step 1.3.

Step 1.3 If the stopping conditions are satisfied, then return the step $(x_{j+1} - x_j)$. Otherwise, set $t_{j+1} = (1 + \sqrt{1 + 4t_j^2})/2$, $y_{j+1} = x_{j+1} + (\frac{t_j - 1}{t_{j+1}})(x_{j+1} - x_j)$, $g_{j+1} = g_j + B_k(y_{j+1} - y_j)$ and go to Step 2.

Step 2 Set $j \coloneqq j + 1$ and go to Step 1.2.

Then, the general algorithm proposed by Karas et al. to solve problem (2.13), by means of any inner-solver satisfying (2.16), is the following.

Algorithm 2.3 General Derivative-Free Trust-Region

Step 0 Given $x_1 \in \Omega$, $\alpha > 0$, $\Delta_1 > 0$, $\eta \in (0, 1)$, set $k \coloneqq 1$.

Step 1.1 Construct the model $m_k(d)$.

Step 1.2 If $\Delta_k < \alpha \pi_k$, go to Step 2. Otherwise, set $\Delta_{k+1} = \frac{\Delta_k}{2}$, $d_k = 0$ and go to Step 3.

Step 2 Find an approximate solution d_k of (2.14). If $\rho_k \coloneqq \frac{f(x_k) - f(x_k + d_k)}{m_k(0) - m_k(d_k)} \geq \eta$, set $\Delta_{k+1} = \Delta_k$ and go to Step 3. Otherwise, set $\Delta_{k+1} = \frac{\Delta_k}{2}$, $d_k = 0$ and go to Step 3.

Step 3 Set $x_{k+1} = x_k + d_k$, $k \coloneqq k + 1$ and go to Step 1.1.

Remarks: The parameter α is arbitrary and allows to balance the magnitude of π_k, depending on the problem. Moreover, if Δ_k is large, then the model may possibly

not be a good approximation of the objective function, that is why they shrink the trust-region radius in Step 1.2 if the condition is not satisfied. Based on hypotheses **A1**, **A2**, **A3** and **A4**, the global convergence of this general algorithm is stated in Theorem 2.2.1 and proved in detail by Karas et al. in [17].

Theorem 2.2.1. *Suppose that Hypotheses A1 to A4 hold. Then,*

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

In their paper, the equation we are most interested is (2.15) in hypothesis **A3**. Satisfying such condition means that the approximated gradient g_k must represent sufficiently well the gradient of the objective function $\nabla f(x_k)$, which will be a key property of the new method.

2.3 A Derivative-Free Quadratic Regularization Method

In [12], Grapiglia proposes a derivative-free quadratic regularization method with finite-difference gradient approximations for smooth unconstrained problems

$$\min_{x\in\mathbb{R}^n} \quad f(x).$$

The method generates a sequence of approximated solutions for quadratic unconstrained sub-problems

$$\min_{y\in\mathbb{R}^n} \quad m_{x,\sigma}(y) := f(x) + \langle g, y - x\rangle + \frac{1}{2}\langle B(y-x), y-x\rangle + \frac{\sigma}{2}\|y-x\|^2,$$

subject to some acceptance conditions. $g \in \mathbb{R}^n$ is an approximation to the gradient, $B \in \mathbb{R}^{n\times n}$ is a symmetric positive semidefinite and $\sigma > 0$ is the regularization parameter. To demonstrate the efficiency of the method, assumptions **A1**, **A2** and **A4** were stated, with $\Omega = \mathbb{R}^n$. The following algorithm is proposed in the paper.

Algorithm 2.3 DFQRM

Step 0 Given $x_1 \in \mathbb{R}^n$, a symmetric positive semi-definite matrix $B_1 \in \mathbb{R}^{n\times n}$, $\sigma_1 \geq \sigma_{min} > 0, \varepsilon > 0$, and $\theta \in [0, 1)$, set $k \coloneqq 1$.

Step 1. Set $i \coloneqq 0$.

Step 1.1 For

$$h_{k,i} = \frac{2\varepsilon}{5(2^i\sigma_k)\sqrt{n}}, \tag{2.17}$$

compute $g_{k,i} \in \mathbb{R}^n$ by

$$[g_{k,i}]_j = \frac{f(x_k + h_{k,i}e_j) - f(x_k)}{h_{k,i}}, \quad j = 1, ..., n.$$

Step 1.2 If

$$\|g_{k,i}\| \geq \frac{4\varepsilon}{5},$$

go to Step 1.3. Otherwise, set $i \coloneqq i + 1$ and go to Step 1.1.

Step 1.3 Consider the quadratic model

$$m_{x_k,2^i\sigma_k}(y) \coloneqq f(x_k) + \langle g_{k,i}, y - x_k\rangle + \frac{1}{2}\langle B_k(y - x_k), y - x_k\rangle + \frac{2^i\sigma_k}{2}\|y - x_k\|^2,$$

and compute an approximate solution $x^+_{k,i}$ of the sub-problem

$$\min_{y\in\mathbb{R}^n} \quad m_{x_k,2^i\sigma_k}(y),$$

such that

$$m_{x_k,2^i\sigma_k}(x^+_{k,i}) \leq f(x_k) \quad \text{and} \quad \|\nabla m_{x_k,2^i\sigma_k}(x^+_{k,i})\| \leq \theta(2^i\sigma_k)\|x^+_{k,i} - x_k\|.$$

Step 1.4 If

$$f(x_k) - f(x^+_{k,i}) \geq \frac{(1-\theta)(2^i\sigma_k)}{8}\|x^+_{k,i} - x_k\|^2$$

holds, set $i_k = i, g_k = g_{k,i_k}$ and go to Step 2. Otherwise, set $i \coloneqq i + 1$ and go to Step 1.1.

Step 2 Set $x_{k+1} = x^+_{k,i_k}, \sigma_{k+1} = \max\{2^{i_k-1}\sigma_k, \sigma_{min}\}$, choose a symmetric positive semi-definite matrix $B_{k+1} \in \mathbb{R}^{n\times n}$, set $k \coloneqq k + 1$, and go to Step 1.

Under the hypotheses mentioned, it is shown by Theorem 2.3.1 that the proposed method needs at most $\mathcal{O}(n\varepsilon^{-2})$ function evaluations to reach an ε-approximate solution for nonconvex problems. The formal proof is provided in [12].

Theorem 2.3.1. *Suppose that A1, A2 and A4 hold. Let $T(\varepsilon)$ be the first iteration index such that $\|\nabla f(x_{T(\varepsilon)+1})\| \leq \varepsilon$ and let $FE(\varepsilon)$ be the total number of function evaluations performed by Algorithm 2.3 up to the $T(\varepsilon)^{th}$ iteration. Then,*

$$FE(\varepsilon) \leq \mathcal{O}(n\varepsilon^{-2}). \tag{2.18}$$

The main technique comes from the use of $\sqrt{n}$ in (2.17), which at the end avoids the number of function evaluations $FE(\varepsilon)$ to depend quadratically on the problem dimension n. The worst-case complexity (2.18) is the best bound obtained up to now in the literature for deterministic DFO methods applied to smooth nonconvex unconstrained optimization problems with exact function values.

2.4 A New Derivative-Free Trust-Region Method

In this work, we bring a contribution to ideas from both articles [17] and [12]. In [17], the construction of the gradient approximation was not discussed by Karas et al., while here we will explicitly impose all components to follow a finite-difference approximation. Furthermore, in [17], the asymptotic global convergence to a stationary point is obtained, but nothing has been said on the maximum number of iterations and function evaluations needed to reach an ε-approximate stationary point. The theoretical analyses needed to obtain those bounds are presented in this master thesis, based on the theoretical developments proposed in both papers [17] and [12].

For the new method, we use finite-differences based on the idea (2.17) proposed by Grapiglia, combined with **A1** in order to recover **A3** used in [17]. As the other hypotheses of the latter paper will be also satisfied, the new method is globally convergent too. Therefore, by imposing a wise condition on h, we recover condition (2.15) that was needed for Karas et al. to prove the global convergence of trust-region methods for convexly constrained problems. As a result, this combination generates a new derivative-free trust-region method, globally convergent, based on finite-difference gradient approximations, subject to a closed nonempty convex set. The use of inequality (2.15) is the foundation allowing to exploit the theoretical

analysis of the new method. For the first time, a derivative-free trust-region method is built based on finite-difference gradient approximations.

Chapter 3

The New Method

This chapter presents the new method. Section 3.1 starts with the assumptions of our work, while Section 3.2 gives some definitions and includes the auxiliary results. Then, we present a theoretical analysis of the new method, for which two cases are considered. The first one is mentioned in Section 3.3, in which we deal with general convex domains where function values can be computed outside the feasible set. Its worst-case complexity is presented in Section 3.4. Then, the second case is shown in Section 3.5. It is a particular problem for which variables are bounded, as in a box, and where no function evaluation is allowed out of it.

3.1 Assumptions

Now, we consider assumptions **A1**, **A2** and (2.16).

3.2 Notations and Auxiliary Results

The aim is to solve smooth convexly constrained problems of the form

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & x \in \Omega, \end{aligned} \tag{3.1}$$

by means of a derivative-free trust-region method, based on finite-difference gradient approximations. Given $x \in \Omega$, we define respectively the quadratic model around x, the stationarity measure at x, the ratio of decrease in f with respect to the model

decrease, the trust-region radius around x and some number η as:

$$\begin{aligned} m(d) &\coloneqq f(x) + \langle g, d\rangle + \frac{1}{2}\langle Bd, d\rangle, \\ \pi &\coloneqq \|x - P_\Omega(x - g)\|, \\ \rho &\coloneqq \frac{f(x) - f(x+d)}{m(0) - m(d)}, \\ \Delta &> 0, \\ \eta &\in (0, 1), \end{aligned}$$

where $g \in \mathbb{R}^n$ is the finite-difference gradient approximation to $\nabla f(x)$ and $B \in \mathbb{R}^{n\times n}$ is a symmetric matrix. In the first place, general convex sets are considered with function evaluations allowed outside of the feasible set. Therefore, there is no specific restriction on g and one could decide to use a FFD to build it. However, it should still satisfy (2.15), which is not an assumption here, but an implication of the construction of the new method. The first Lemma recovers this latter inequality.

Lemma 3.2.1. *Suppose that A1 holds. Given $x \in \Omega$ and $h > 0$, let $g \in \mathbb{R}^n$ be defined by*

$$g_j = \frac{f(x + he_j) - f(x)}{h} \quad j = 1, ..., n \tag{3.2}$$

with

$$0 < h = \min\left\{\frac{\varepsilon}{\sqrt{n}}, \frac{\Delta}{\sqrt{n}}\right\}. \tag{3.3}$$

Then,

$$\|\nabla f(x) - g\| \leq \frac{L}{2}\Delta. \tag{3.4}$$

Proof. By **A1**, (3.2) and (3.3),

$$\|\nabla f(x) - g\| \leq \frac{\sqrt{n}L}{2}h \leq \frac{L}{2}\Delta.$$

□

The next Lemma proves that if the trust-region radius is sufficiently small, then the iteration is successful and the stationarity measure can be bounded from below.

Lemma 3.2.2. *Assume that*

$$\|x - P_\Omega(x - \nabla f(x))\| > \varepsilon,$$

for some $\varepsilon \in (0, 1)$*, and*

$$\Delta \leq \min\left\{\frac{2}{L}, \frac{1}{1+\|B\|}, \frac{c_1(1-\eta)}{L+\|B\|/2}\right\}\frac{\varepsilon}{2}. \tag{3.5}$$

Then,

$$\rho \geq \eta \quad \textit{and} \quad \pi \geq \frac{\varepsilon}{2}.$$

Proof. By triangle inequality, the contraction property of projections and Lemma 3.2.1,

$$\begin{aligned}\|x - P_\Omega(x - \nabla f(x))\| &\leq \|x - P_\Omega(x - g)\| + \|P_\Omega(x - g) - P_\Omega(x - \nabla f(x))\| \\ &\leq \pi + \|\nabla f(x) - g\| \\ &\leq \pi + \frac{L}{2}\Delta.\end{aligned}$$

Since $\varepsilon < \|x - P_\Omega(x - \nabla f(x))\|$ and $\Delta \leq \frac{\varepsilon}{L}$, we have

$$\pi \geq \frac{\varepsilon}{2}.$$

Using (2.16), **A1** and Lemma 3.2.1,

$$\begin{aligned}1 - \rho &= \frac{m(0) - m(d) - (f(x) - f(x+d))}{m(0) - m(d)} \\ &= \frac{-(\langle g, d\rangle + \frac{1}{2}\langle Bd, d\rangle + f(x) - f(x+d))}{m(0) - m(d)} \\ &\leq \frac{f(x+d) - f(x) - \langle \nabla f(x), d\rangle + \langle \nabla f(x), d\rangle - \langle g, d\rangle - \frac{1}{2}\langle Bd, d\rangle}{c_1\pi \min\left\{\frac{\pi}{1+\|B\|}, \Delta\right\}} \\ &\leq \frac{|f(x+d) - f(x) - \langle \nabla f(x), d\rangle| + |\langle \nabla f(x), d\rangle - \langle g, d\rangle| + \left|\frac{1}{2}\langle Bd, d\rangle\right|}{c_1\pi \min\left\{\frac{\pi}{1+\|B\|}, \Delta\right\}} \\ &\leq \frac{\frac{L}{2}\|d\|^2 + \|\nabla f(x) - g\|\|d\| + \frac{1}{2}\|B\|\|d\|^2}{c_1\pi \min\left\{\frac{\pi}{1+\|B\|}, \Delta\right\}} \\ &\leq \frac{\frac{L}{2}\Delta^2 + \frac{L}{2}\Delta^2 + \frac{\|B\|}{2}\Delta^2}{c_1\pi \min\left\{\frac{\pi}{1+\|B\|}, \Delta\right\}}\end{aligned}$$

$$\leq \frac{(L + \frac{\|B\|}{2})\Delta^2}{c_1 \frac{\varepsilon}{2} \min\left\{\frac{\varepsilon}{2(1+\|B\|)}, \Delta\right\}}.$$

By (3.5), we get

$$\Delta \leq \frac{\varepsilon}{2(1 + \|B\|)}.$$

Therefore,

$$1 - \rho \leq \frac{(L + \frac{\|B\|}{2})\Delta}{c_1 \frac{\varepsilon}{2}}.$$

Moreover by (3.5), we also have

$$\Delta \leq \frac{c_1(1-\eta)}{(L + \|B\|/2)} \frac{\varepsilon}{2}.$$

Then,

$$1 - \rho \leq 1 - \eta$$
$$\rho \geq \eta.$$

□

Let us consider now the following algorithm, which is the new method, TRFD, applied for general convex sets.

3.3 Algorithm

Algorithm 3.1 TRFD for general convex sets

Step 0 Given $x_1 \in \Omega$, $\varepsilon \in (0,1)$, $\Delta_1 > 0$, $\Delta_{max} > \Delta_1$, $\eta \in (0,1)$, $B_1 \in \mathbb{R}^{n\times n}$ symmetric, set $k \coloneqq 1$.

Step 1.1 Set $i \coloneqq 0$.

Step 1.2 For $h_i \coloneqq \min\left\{\frac{\varepsilon}{\sqrt{n}}, \frac{(0.5)^i\Delta_k}{\sqrt{n}}\right\}$, compute $g_k^{(i)}$ such that

$$\|\nabla f(x_k) - g_k^{(i)}\| \leq \frac{L}{2}\sqrt{n}h_i. \tag{3.6}$$

Step 1.3 If $\pi_k^{(i)} \coloneqq \|x_k - P_\Omega(x_k - g_k^{(i)})\| \geq \varepsilon/2$, then go to Step 2.1. Otherwise, set $i = i + 1$ and go to Step 1.2.

Step 2.1 Compute $d_k^{(i)}$ solving approximately the trust-region sub-problem

$$\begin{aligned} \min_d \quad & m_k(d) \coloneqq f(x_k) + \langle g_k^{(i)}, d\rangle + \frac{1}{2}\langle B_k d, d\rangle \\ \text{s.t.} \quad & \|d\| \leq (0.5)^i\Delta_k \\ & x_k + d \in \Omega. \end{aligned}$$

Step 2.2 If $\rho_k^{(i)} \coloneqq \frac{f(x_k)-f(x_k+d_k^{(i)})}{m_k(0)-m_k(d_k^{(i)})} \geq \eta$, then set $i_k = i$, $\rho_k = \rho_k^{(i_k)}$, $\pi_k = \pi_k^{(i_k)}$, $g_k = g_k^{(i_k)}, d_k = d_k^{(i_k)}$ and go to Step 3. Otherwise, set $i \coloneqq i + 1$ and go to Step 1.2.

Step 3 Set $x_{k+1} = x_k + d_k$, $\Delta_{k+1} = \min\left\{(0.5)^{i_k-1}\Delta_k, \Delta_{max}\right\}$, $B_{k+1} \in \mathbb{R}^{n\times n}$ symmetric, $k \coloneqq k + 1$ and go to Step 1.1.

Remark: one way to compute the approximated gradient g_k is to produce a forward finite-difference at each component, i.e.,

$$[g_k]_j = \frac{f(x_k + he_j) - f(x_k)}{h}, \quad j = 1, ..., n,$$

but other computations are possible, as long as g_k satisfies inequality (3.6) in Step

1.2. Now, let us consider the additional assumption **A4**. With this assumption, the next Lemma gives upper and lower bounds on Δ_k.

Lemma 3.3.1. *Suppose that A1 and A4 hold. Given $\varepsilon \in (0, 1)$, let $\{\Delta_k\}$ be generated by TRFD. Then,*

$$\tau\varepsilon \leq \Delta_k \leq \Delta_{max} \quad \forall\, k \geq 1, \tag{3.7}$$

where τ is a constant defined as

$$\tau := \frac{\min\left\{4\Delta_1, \frac{2}{L}, \frac{1}{1+M}, \frac{c_1(1-\eta)}{L+M/2}\right\}\frac{1}{2}}{2}.$$

Proof. Let us work through an induction argument. Clearly, (3.7) is true for $k = 1$. Now let us suppose it is true for some $k \geq 1$. By Step 3 of the new algorithm, we have

$$\Delta_{k+1} = \min\left\{(0.5)^{i_k-1}\,\Delta_k, \Delta_{max}\right\} \leq \Delta_{max}.$$

Now it remains to show that

$$\Delta_{k+1} = \min\left\{(0.5)^{i_k-1}\,\Delta_k, \Delta_{max}\right\} \geq \tau\varepsilon.$$

In the case where

$$\Delta_{k+1} = \Delta_{max},$$

then

$$\Delta_{k+1} > \Delta_1 \geq \tau\varepsilon.$$

For the other case, we need to consider two cases when

$$\Delta_{k+1} = (0.5)^{i_k-1}\,\Delta_k.$$

If $i_k = 0$, then

$$\Delta_{k+1} = 2\Delta_k > \Delta_k \geq \tau\varepsilon.$$

For $i_k \geq 1$, let us suppose by contradiction that

$$\Delta_{k+1} = (0.5)^{i_k-1}\Delta_k < \tau\varepsilon,$$

is true. By Lemma 3.2.2, this would mean that at iteration k, the first index allowing $\rho_k^{(i)} \geq \eta$ was $i = i_k - 1$, which contradicts the optimality of i_k, as it would lead to

$i_k = i_k - 1$. Therefore,

$$\Delta_{k+1} \geq \tau\varepsilon.$$

So (3.7) holds for $k+1$, which completes the induction argument. □

3.4 Worst-Case Complexity Analysis

The next Theorem gives a worst-case complexity on the number of iterations needed by *TRFD* before reaching an ε-approximate stationary point.

Theorem 3.4.1. *Suppose that A1, A2, (2.16) and A4 hold. Given $\varepsilon \in (0,1)$, let $\{x_k\}_{k=1}^{T}$ be generated by TRFD such that*

$$\|x_k - P_\Omega(x_k - \nabla f(x_k))\| > \varepsilon \quad k = 1, ..., T.$$

Then,

$$T \leq \mathcal{O}(\varepsilon^{-2}).$$

Proof. Using (2.16), Step 1.3 of Algorithm 3.1, Lemma 3.3.1 and **A2**, we get

$$\begin{aligned}
f(x_1) - f_{low} &\geq f(x_1) - f(x_{T+1}) \\
&= \sum_{k=1}^{T} f(x_k) - f(x_{k+1}) \\
&\geq \sum_{k=1}^{T} \eta(m_k(0) - m_k(d_k)) \\
&\geq \sum_{k=1}^{T} \eta c_1 \pi_k \min\left\{\frac{\pi_k}{1+M}, (0.5)^{i_k}\Delta_k\right\} \\
&\geq \sum_{k=1}^{T} \eta c_1 \frac{\varepsilon}{2} \min\left\{\frac{\varepsilon}{2(1+M)}, (0.5)\Delta_{k+1}\right\} \\
&\geq \sum_{k=1}^{T} \eta c_1 \frac{\varepsilon}{2} \min\left\{\frac{\varepsilon}{2(1+M)}, (0.5)\tau\varepsilon\right\} \\
&= T\eta c_1 \frac{\varepsilon^2}{4} \min\left\{\frac{1}{1+M}, \tau\right\}.
\end{aligned}$$

We conclude that

$$T \leq \frac{4(f(x_1) - f_{low})}{\eta c_1 \min\left\{\frac{1}{1+M}, \tau\right\}} \varepsilon^{-2} = \mathcal{O}(\varepsilon^{-2}).$$

□

The next Corollary establishes a bound on the number of function evaluations needed by *TRFD* to get an ε-approximate stationary point.

Corollary 3.4.2. *Let $\{x_k\}_{k\geq 1}$ be generated by TRFD. Given $\varepsilon \in (0,1)$, assume that $T(\varepsilon)$ is the first iteration index such that*

$$\|x_{T(\varepsilon)+1} - P_\Omega(x_{T(\varepsilon)+1} - \nabla f(x_{T(\varepsilon)+1}))\| \leq \varepsilon,$$

and let $FE(\varepsilon)$ be the total number of function evaluations up to the $T(\varepsilon)^{th}$ iteration of TRFD. Then,

$$FE(\varepsilon) \leq \mathcal{O}(n\varepsilon^{-2}).$$

Proof. At iteration k, the maximum number of function evaluations is bounded by $(n+1)(i_k+1)$. On the other hand, $\Delta_{k+1} = \min\{(0.5)^{i_k-1}\Delta_k, \Delta_{max}\}$, so

$$\begin{aligned} &\Delta_{k+1} \leq (0.5)^{i_k-1}\Delta_k \\ \implies\ &i_k + 1 \leq \log_{0.5}(\frac{\Delta_{k+1}}{\Delta_k}) + 2. \end{aligned}$$

Thus, considering the evaluation of $f(x_1)$ and (3.5) in Lemma 3.2.2, we get

$$\begin{aligned} FE(\varepsilon) &\leq 1 + \sum_{k=1}^{T}(n+1)(i_k+1) \\ &\leq 1 + (n+1)(2T + \log_{0.5}(\Delta_{T+1}) - \log_{0.5}(\Delta_1)) \\ &\leq 1 + (n+1)(2T + \log_{0.5}(\tau\varepsilon) - \log_{0.5}(\Delta_1)) \\ &= \mathcal{O}(n(\varepsilon^{-2} + \log_{0.5}(\varepsilon))) \\ &= \mathcal{O}(n(\varepsilon^{-2} + \log_2(\varepsilon^{-1}))) \\ &= \mathcal{O}(n\varepsilon^{-2}). \end{aligned}$$

□

3.5 Bound Constraints

This section is a specific case of the previous method, for which general convex sets were of interest. Here, we assume the convex domain Ω to be expressed through bound constraints. Moreover, we suppose that no function evaluation is available outside the set, so we forbid the computation of function values outside the domain. Bound constraints problems are relevant in many applications, as in physical or dynamical problems, where parameters must belong to some range of values. As a result, those constraints imply to adjust the way we build the gradient approximation. Indeed, if no forward function evaluation is allowed for the j^{th} component of x, then we should evaluate the function backwards, leading to a backward finite-difference for the j^{th} component of g.

We state one Lemma to demonstrate the fundamental inequality (2.15) needed on the accuracy of the approximated gradient, recovered as (3.4) in Lemma 3.2.1 for general convex sets. Figure 3.1 illustrates the definitions involved for the next statement, in the specific case where $n = 2$.

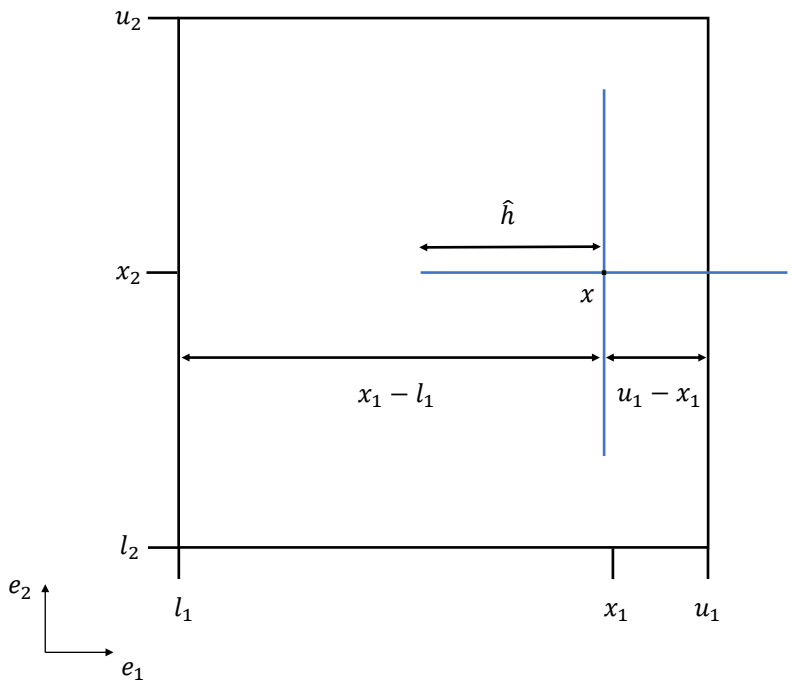


Figure 3.1: Illustration of Lemma 3.5.1

Lemma 3.5.1. *Suppose that A1 holds. Given $x \in \Omega$ and $\Delta > 0$, let $\hat{h} \leq \frac{\Delta}{\sqrt{n}}$. Moreover, let us define $h_j^F = \min\left\{u_j - x_j, \hat{h}\right\}$, $h_j^B = \min\left\{x_j - l_j, \hat{h}\right\}$ with $l_j < u_j$ and $g \in \mathbb{R}^n$ by*

$$[g]_j = \begin{cases} \frac{f(x+h_j^F e_j)-f(x)}{h_j^F} & \text{if } h_j^F \geq h_j^B, \\ \frac{f(x)-f(x-h_j^B e_j)}{h_j^B} & \text{otherwise,} \end{cases}$$

for $j = 1, ..., n$. Then,

$$\|\nabla f(x) - g\| \leq \frac{L}{2}\Delta.$$

Proof. If $h_j^F \geq h_j^B$,

$$\begin{aligned} &\left|f(x + h_j^F e_j) - f(x) - \langle \nabla f(x), h_j^F e_j\rangle\right| \leq \frac{L}{2}\|h_j^F e_j\|^2 \\ \implies &\left|\frac{f(x + h_j^F e_j) - f(x)}{h_j^F} - \langle \nabla f(x), e_j\rangle\right| \leq \frac{L}{2}h_j^F \\ \implies &\left|[g]_j - [\nabla f(x)]_j\right| \leq \frac{L}{2}h_j^F \\ \implies &\left|[g]_j - [\nabla f(x)]_j\right| \leq \frac{L}{2}\frac{\Delta}{\sqrt{n}}. \end{aligned}$$

Otherwise, if $h_j^F < h_j^B$,

$$\begin{aligned} &\left|f(x) - f(x - h_j^B e_j) - \langle \nabla f(x), h_j^B e_j\rangle\right| \leq \frac{L}{2}\|h_j^B e_j\|^2 \\ \implies &\left|\frac{f(x) - f(x - h_j^B e_j)}{h_j^B} - \langle \nabla f(x), e_j\rangle\right| \leq \frac{L}{2}h_j^B \\ \implies &\left|[g]_j - [\nabla f(x)]_j\right| \leq \frac{L}{2}h_j^B \\ \implies &\left|[g]_j - [\nabla f(x)]_j\right| \leq \frac{L}{2}\frac{\Delta}{\sqrt{n}}. \end{aligned}$$

Consequently,

$$\|g - \nabla f(x)\| = \sqrt{\sum_{j=1}^{n}\left|[g]_j - [\nabla f(x)]_j\right|^2} \leq \sqrt{\sum_{j=1}^{n}\left(\frac{L}{2}\frac{\Delta}{\sqrt{n}}\right)^2} = \sqrt{n\left(\frac{L}{2}\frac{\Delta}{\sqrt{n}}\right)^2} = \frac{L}{2}\Delta.$$

□

This result recovers the same inequality (3.4) obtained in Lemma 3.2.1. Since other statements, i.e., Lemma 3.2.2, Lemma 3.3.1, Theorem 3.4.1 and Corollary 3.4.2 were

proved for general convex sets, and as by Lemma 3.5.1 the inequality (2.15) is still valid for bound constraints with $c_2 = \frac{L}{2}$, the complexity analysis done in Section 3.4 for general convex domains remains true in this specific case.

The following algorithm is the new method, TRFD, applied in the bound constraints case, where l and u are vectors composed of the lower and upped bounds in each direction j, respectively. Step 0 and Step 1 are presented on page 35, while Step 2 and Step 3 are on the following page. The presentation of the algorithm is followed by practical remarks about the initial parameters η and Δ_1. Those remarks are available for both Algorithms 3.1 and 3.2.

Algorithm 3.2 TRFD for bound constraints sets

Step 0 Given $x_1 \in \Omega := [l, u]$, $\varepsilon \in (0, 1)$, $\Delta_1 > 0$, $\Delta_{max} > \Delta_1$, $\eta \in (0, 1)$, $B_1 \in \mathbb{R}^{n\times n}$ symmetric, set $k := 1$.

Step 1.1 Set $i := 0$.

Step 1.2 For $h_i := \min\left\{\frac{\varepsilon}{\sqrt{n}}, \frac{(0.5)^i\Delta_k}{\sqrt{n}}\right\}$, let

$$h_j^F := \min\{u_j - x_j, h_i\} \quad \text{and} \quad h_j^B := \min\{x_j - l_j, h_i\} \quad j = 1, ..., n.$$

Step 1.3 If $h_j^F \geq h_j^B$, compute the j^{th} component of $g_k^{(i)} \in \mathbb{R}^n$ such that

$$[g_k^{(i)}]_j = \frac{f(x_k + h_j^F e_j) - f(x_k)}{h_j^F}.$$

Otherwise,

$$[g_k^{(i)}]_j = \frac{f(x_k) - f(x_k - h_j^B e_j)}{h_j^B}.$$

Step 1.4 If $\pi_k^{(i)} := \|x_k - P_\Omega(x_k - g_k^{(i)})\| \geq \varepsilon/2$, then go to Step 2.1. Otherwise, set $i := i + 1$ and go to Step 1.2.

Step 2.1 Compute $d_k^{(i)}$ solving approximately the trust-region sub-problem

$$\begin{aligned} \min_d \quad & m_k(d) \coloneqq f(x_k) + \langle g_k^{(i)}, d\rangle + \frac{1}{2}\langle B_k d, d\rangle \\ \text{s.t.} \quad & \|d\| \leq (0.5)^i \Delta_k \\ & x_k + d \in \Omega. \end{aligned}$$

Step 2.2 If $\rho_k^{(i)} \coloneqq \frac{f(x_k)-f(x_k+d_k^{(i)})}{m_k(0)-m_k(d_k^{(i)})} \geq \eta$, then set $i_k = i$, $\rho_k = \rho_k^{(i_k)}$, $\pi_k = \pi_k^{(i_k)}$, $g_k = g_k^{(i_k)}, d_k = d_k^{(i_k)}$ and go to Step 3. Otherwise, set $i \coloneqq i + 1$ and go to Step 1.2.

Step 3 Set $x_{k+1} = x_k + d_k$, $\Delta_{k+1} = \min\left\{(0.5)^{i_k-1}\Delta_k, \Delta_{max}\right\}$, $B_{k+1} \in \mathbb{R}^{n\times n}$ symmetric, $k \coloneqq k + 1$ and go to Step 1.1.

Remarks: the choice of η to class an iteration as successful or not is critical. On one hand, we should not take high values for η, as it could lead to loop several times in Step 2.1 to get $\rho_k \geq \eta$, which implies to decrease a lot the trust-region radius, especially at the first iteration $k = 1$, and would lead to an excessive number of function evaluations. Moreover, in such scenarios, iteration k leaves a small radius to the next iterates, which means a less efficient decrease for these. On the other hand, η should not be too small, as it would imply to allow less good ratios, which are often linked to poor approximations of the model. As a consequence, a trade-off should be chosen regarding η.

Moreover, added to a wise choice for η, the value of Δ_1 also matters. It should be sufficiently large in order to allow an efficient big first step $x_2 = x_1 + d_1$, but this could potentially lead to a bad solution x_2 if Δ_1 is too large. Therefore, the radius should not be too big, but it should also not be too small, which could possibly lead to miss an efficient first decrease and thus impact the next iterations. Therefore, a trade-off should be also taken for Δ_1.

Chapter 4

Illustrative Numerical Results

In Section 4.1, numerical results are presented considering a set of unconstrained benchmark problems. In Section 4.2, we illustrate the applicability of the proposed method in the calibration of a SIR model.

The numerical experiments were performed with MATLAB (R2023a) on a PC with microprocessor Intel(R) Core(TM) i5-7200U CPU, and 8 GB of RAM memory.

4.1 Comparison on Benchmark Problems

In our first experiment, we compared the following MATLAB implementations on the set of 102 test problems from the Andrei's collection [18]:

- **TRFD**: Algorithm 3.1, with the following parameters: $\Delta_1 = 20$, $\Delta_{max} = 1500$, $\varepsilon = 10^{-5}$ and B_k updated by the *BFGS* formula, i.e.,

$$B_{k+1} = \begin{cases} B_k + \frac{y_k y_k^T}{s_k^T y_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} & \text{if} \quad s_k^T y_k > 0, \\ B_k \quad \text{otherwise,} \end{cases}$$

with $B_1 = I$, $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$.

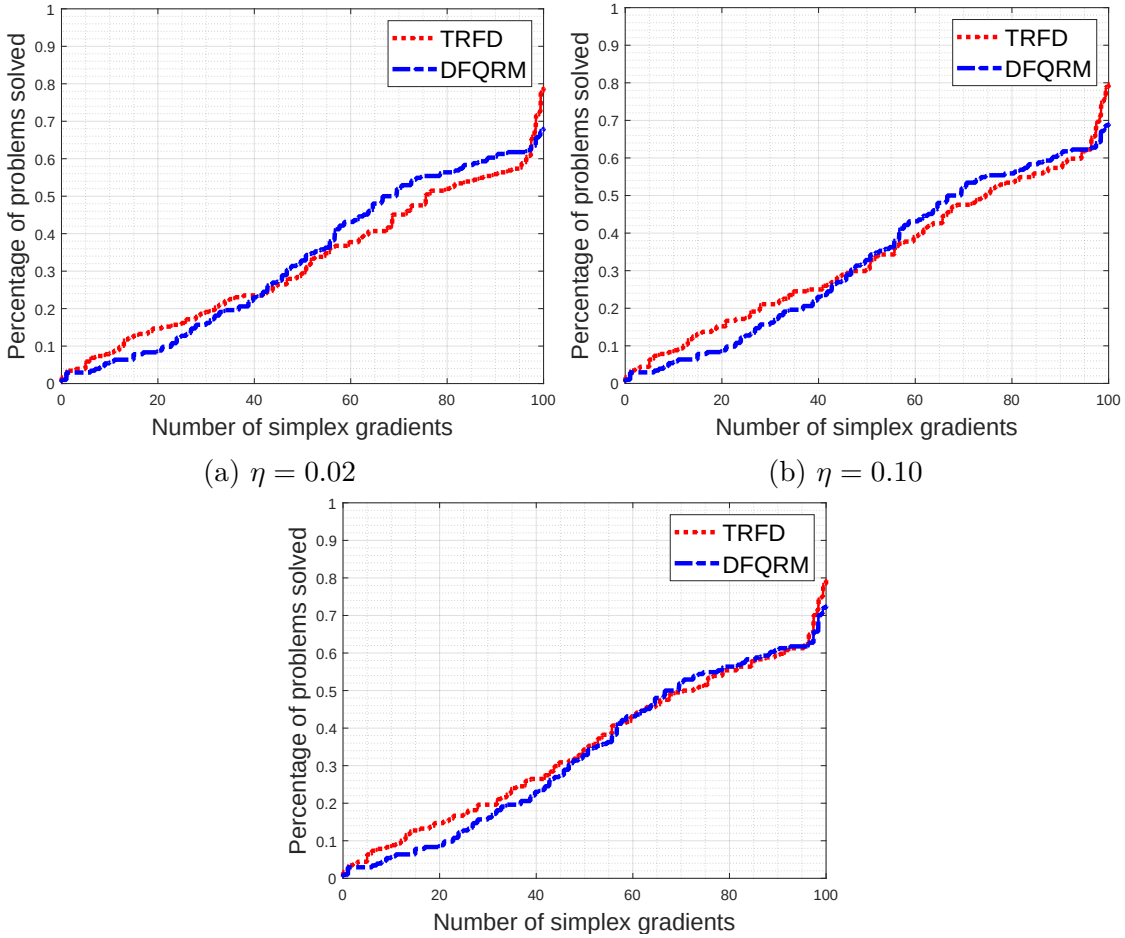
- **DFQRM**: Algorithm 1 proposed by Grapiglia in [12], with $\sigma_{min} = 10^{-2}, \varepsilon = 10^{-5}, \theta = 0$ and B_k updated by the *BFGS* formula, with $B_1 = I$, $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$.

In TRFD, the trust-region sub-problems are solved using the Newton's method with the safe-guards described in Section 2.1.1. Moreover, the stationarity measure

is simplified to $\pi_k = \|g_k\|$, as

$$P_\Omega(x_k - g_k) = x_k - g_k.$$

For all functions in [18], the dimension was fixed as $n = 48$. Moreover, two choices of starting points were used, namely, $x_0 = 10^s\, x_0$, where $s \in \{0, 1\}$ and x_0 is the starting point provided in [18]. For each problem, a budget of $100(n + 1) = 4900$ function evaluations was allowed to each method, which corresponds to 100 simplex gradients. Figure 4.1 shows the corresponding *data profiles*[1] for different values of η. Parameter η is the minimum value that ρ_k should satisfy at each iteration, as stated in condition (1.2).



(a) $\eta = 0.02$ (b) $\eta = 0.10$

(c) $\eta = 0.15$

Figure 4.1: Evolution of data profiles as a function of η

[1] Those were performed using the code *data_profile.m* from Moré and Wild, freely available at https://www.mcs.anl.gov/~more/dfo/.

Two cumulative information are represented on data profiles. On one hand, the horizontal axis is the number of *simplex gradients* needed by a method to solve a problem, one simplex gradient corresponding to n function evaluations. While on the other hand, the vertical axis gives the cumulative percentage of problems solved by a method. A method is said to *solve* a problem with tolerance $\tau \in (0, 1)$ if it generates a point x_* satisfying the convergence test

$$\frac{f(x_0) - f(x_*)}{f(x_0) - f_L} \geq 1 - \tau,$$

where x_0 is the starting point of the problem, and f_L is the lowest function value obtained by any of the compared methods within the 100 simplex gradients budget. Therefore, by means of data profiles, the graphs presented on Figure 4.1 can show the competitiveness between TRFD and DFQRM, with η successively set to $0.02, 0.10$ and 0.15, for $\tau = 10^{-7}$. When η grows, TRFD's curve climbs, especially between 50 and 90 simplex gradients. In addition, if the maximum budget was halved, we observe TRFD would be the method that solves the most problems, especially for small numbers of simplex gradients. For higher budgets, both methods compete until reaching the highest number of function evaluations allowed.

As a remark, other methods such as DS or FDBFGS would also have been of interest to be tested. However, as seen in the numerical experiments provided by Grapiglia [12], those two methods were dominated by DFQRM. Therefore, we made the choice to test TRFD only with respect to DFQRM.

4.2 A Box-Constrained Problem

In all fields of science, data is observed to understand the behavior of specific systems. One aim is to fit the given data by a model to estimate the data we were not able to measure, and thus obtain quantitative approximations about them. Usually, the model is determined so that it minimizes the error it produces on the given data. Those models, in fact, are often solutions of dynamical systems which are governed by Ordinary Differential Equations (ODE). A pandemic is an example of a dynamical system, which can be approached by a SIR model system (4.1). It is expressed through 3 differential equations, depending on two parameters β and γ. β denotes the transmission rate of the disease, while γ is the rate of recovery, both assumed to be constant during the pandemic. According to their definitions, both

parameters β and γ have a value lying between 0 and 1.

$$\begin{aligned} \frac{dS}{dt} &= -\beta SI, \\ \frac{dI}{dt} &= \beta SI - rI, \\ \frac{dR}{dt} &= rI. \end{aligned} \tag{4.1}$$

The first ODE gives the dynamical behavior of the *Susceptible* population, the second leads the *Infected* population behavior, while the last one governs the *Recovered* population. Consequently, given (β, γ), the solutions of this system are $S(t;\beta,\gamma), I(t;\beta,\gamma)$ and $R(t;\beta,\gamma)$, which denote respectively S, I and R in (4.1) and are three time-dependent curves. Those are the models giving the behavior of the three populations over time, for such rates β and γ. Therefore, considering some $x_{true} = (\beta_{true}, \gamma_{true})$, the aim of this application is to recover x_{true} given a noisy data set $b = F(x_{true}) + \varepsilon$ where $F : [0,1] \times [0,1] \to \mathbb{R}^{3\times 101}$ and $\varepsilon \sim \mathcal{N}(0, 02)$, as done similarly in [20]. We chose normalized initial conditions

$$\begin{aligned} S(0; x_{true}) &= 1 - 10^{-6}, \\ I(0; x_{true}) &= 10^{-6}, \\ R(0; x_{true}) &= 0, \end{aligned}$$

and discretized the time interval $[0, 100]$ by increments of size 1. Given some x, the continuous solutions generated by (4.1) are denoted as $S(t;x), I(t;x)$ and $R(t;x)$. Then, we define $s_j(x) = S(t_j;x)$, $i_j(x) = I(t_j;x)$ and $r_j(x) = R(t_j;x)$ for $j = 0, ..., 100$. Moreover, we set $F(x) = [s(x), i(x), r(x)]$, with $s(x) = [s_0(x), ..., s_{100}(x)]$, $i(x) = [i_0(x), ..., i_{100}(x)]$ and $r(x) = [r_0(x), ..., r_{100}(x)]$. Finally, the noisy data is synthetically generated as $b = F(x_{true}) + \varepsilon$, using $x_{true} = (0.5, 0.3)$. In order to recover x_{true}, we solve

$$\begin{aligned} \min_{x} \quad & \|F(x) - b\|^2 \\ \text{s.t.} \quad & x \in [0,1] \times [0,1]. \end{aligned} \tag{4.2}$$

The objective function $f(x) = \|F(x) - b\|^2$ can be written as

$$f(\beta,\gamma) = \|F(\beta,\gamma) - b\|^2 = \sum_{j=0}^{100} (s_j(\beta,\gamma) - b_j)^2 + \sum_{j=0}^{100} (i_j(\beta,\gamma) - b_j)^2 + \sum_{j=0}^{100} (r_j(\beta,\gamma) - b_j)^2.$$

Therefore, given a noisy data set governed by (4.1), as in Figure 4.2, the goal is to recover the solution $x_{true} = (\beta_{true}, \gamma_{true}) = (0.5, 0.3)$ producing the three curves $S(t; x_{true}), I(t; x_{true})$ and $R(t; x_{true})$ that minimize the error on the noisy data, in the sense of the *Least Square Error* (LSE) approximation, as done in Figure 4.3.

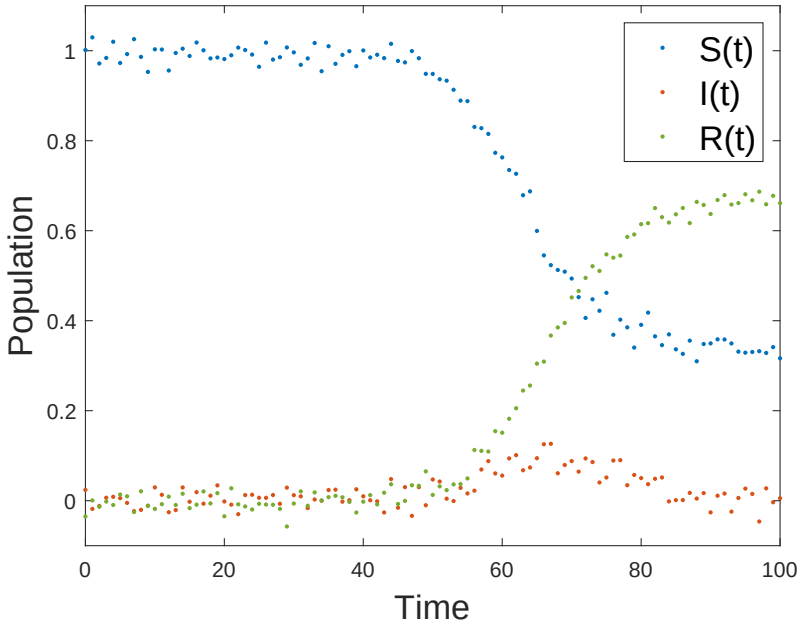


Figure 4.2: SIR data

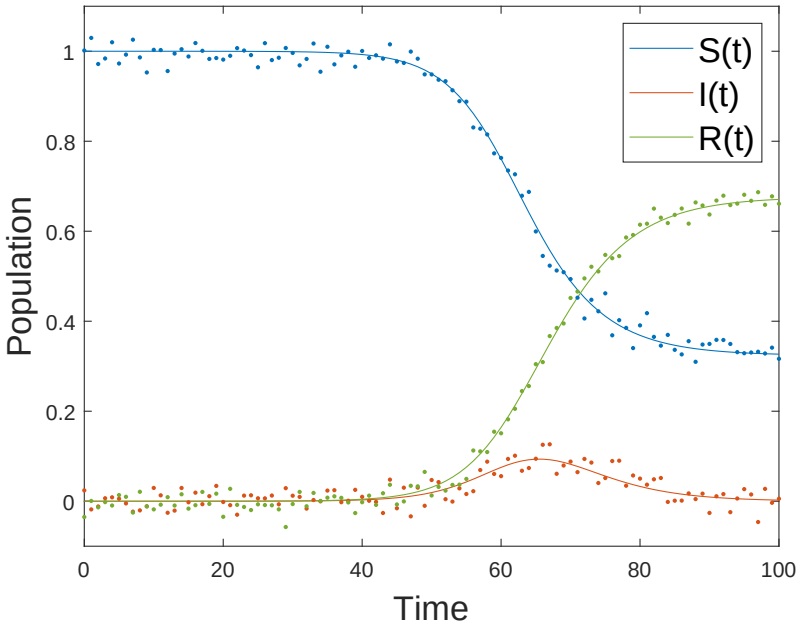


Figure 4.3: SIR curves

Remarks: In reality, a data set is never *exactly* governed by a system of ODE, since equations always have limitations to describe dynamical behaviors of real complex systems. That is why we took perturbed values for the data set, and not exact values.

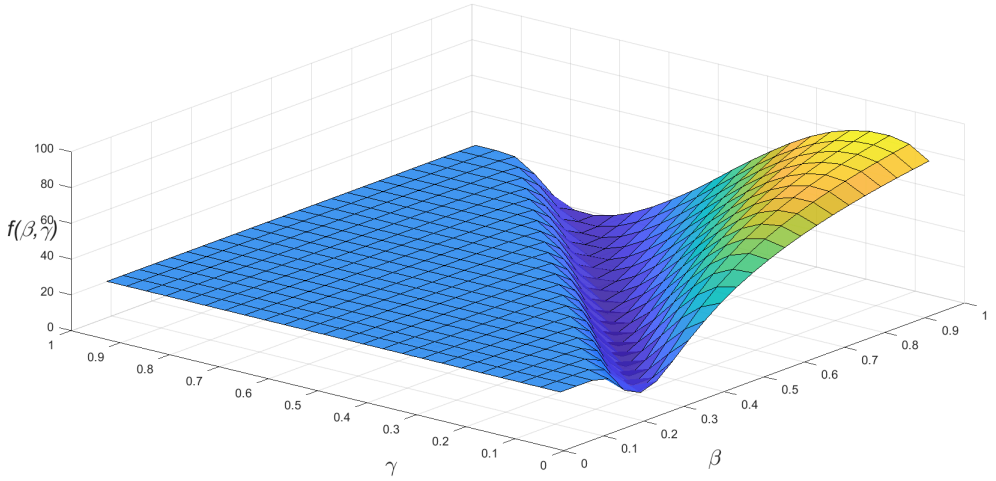


Figure 4.4: Objective function $f(\beta, \gamma)$

Figure 4.4 shows $f(\beta, \gamma)$ for $(\beta, \gamma) \in [0, 1] \times [0, 1]$, where a global minimum is present around the point $(0.5, 0.3)$. We observe that choosing a starting point x_0 such that $\gamma > \beta$ could lead to bad results, as the function values are almost constant around x_0 and thus induce local minima. Indeed, for such points (β, γ), the recovery rate γ is bigger than the transmission rate β, which gives no chance to produce a pandemic. Therefore, system (4.1) produces an abnormal pandemic behavior, leading to trivial, almost horizontal, solutions for $S(t; x_0), I(t; x_0)$ and $R(t; x_0)$. All points which belong to this region, i.e., approximately all points such that $\gamma > \beta$, produce almost the same behavior in system (4.1). Therefore, these points generate more or less the same error on the data. It is the reason why $f(\beta, \gamma)$ is nearly constant in this whole region.

In this application, Algorithm 3.2 was applied and the sub-problems were solved using FISTA's inner solver [7], with a stopping criteria $\|x_{j+1} - x_j\| \leq 10^{-12}$ and tolerance 10^{-10} for the Dijkstra's projection. In order to judge the performance of TRFD in this specific box-constrained problem, we created a set of 168 equally spaced starting points x_0 in the lower right triangle of the domain $[0, 1] \times [0, 1]$. Then, for each x_0, we compared which of TRFD or the FMINCON Matlab function obtains the lowest value $f(\beta, \gamma)$. During the experimentation, the following parameters were

chosen for TRFD: $\Delta_1 = 0.008$, $\Delta_{max} = 1500$, $\varepsilon = 10^{-8}$ and B_k is updated by the *BFGS* formula, with $B_1 = I$, $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$. The results are presented by a bar graph in Figure 4.5.

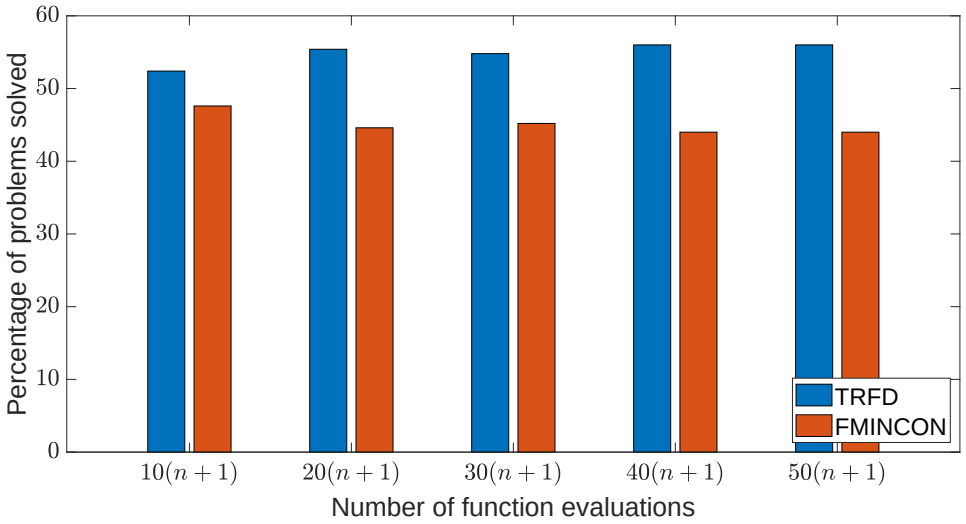


Figure 4.5: TRFD and FMINCON

Chapter 5

Conclusion

5.1 Summary

In this work, we presented a derivative-free trust region method, based on finite-difference gradient approximations, for smooth convexly constrained optimization problems. Under common assumptions, we obtained a bound on the number of function evaluations for the method to reach an ε-approximate stationary point. The use of the problem dimension inside the gradient approximations allowed this bound to depend linearly on n. Numerical experiments were also provided and showed the relative efficiency of TRFD.

5.2 Directions for Future Research

In a future research, it would be relevant to explore derivative-free trust-region methods, based on finite-differences, for composite non-smooth functions with possibly inexact function values. Although we used the l_2-norm in our application, it would be sometimes suitable to make use of the l_1-norm. This choice would give rise to situations where $f(x)$ can be written as $h(F(x) - b)$, with $h = \|\cdot\|_1$ being a non-smooth function. The application of new and more efficient DFO methods, such as derivative-free trust-region methods based on finite-differences, would be appropriate to those problems.

Bibliography

[1] Karbasian H.R. and Vermeire B.C. "Gradient-free aerodynamic shape optimization using Large Eddy Simulation". In: *Computers and Fluids* 232.105185 (2022). DOI: 10.1016/j.compfluid.2021.105185.

[2] Marsden A.L. et al. "A computational framework for derivative-free optimization of cardiovascular geometries". In: *Computer Methods in Applied Mechanics and Engineering* 197.21-24 (2008), pp. 1890–1905. DOI: 10.1016/j.cma.2007.12.009.

[3] Russ J.B. et al. "Design optimization of a cardiovascular stent with application to a balloon expandable prosthetic heart valve". In: *Materials and Design* 209.109977 (2021). DOI: 10.1016/j.matdes.2021.109977.

[4] Audet C. and Orban D. "Finding Optimal Algorithmic Parameters Using Derivative-Free Optimization". In: *SIAM Journal on Optimization* 17.3 (2006), pp. 642–664. DOI: 10.1137/040620886.

[5] Larson J. et al. "Derivative-free optimization methods". In: *Acta Numerica* 28 (2019), pp. 287–404. DOI: 10.1017/S0962492919000060.

[6] Nocedal J. and Wright S.J. *Numerical Optimization*. 2nd ed. Springer, 2006. ISBN: 978-0-387-40065-5.

[7] M. Hough and L. Roberts. "Model-Based Derivative-Free Methods for Convex-Constrained Optimization". In: *SIAM Journal on Optimization* 32 (2022), pp. 2461–2996. DOI: 10.1137/21M1460971.

[8] Garmanjani R. et al. "Trust-region methods without using derivatives: worst case complexity and the nonsmooth case". In: *SIAM Journal on Optimization* 26.4 (2016), pp. 1987–2011. DOI: 10.1137/151005683.

[9] Winfield D. H. "Function and Functional Optimization by Interpolation in Data Tables". In: *PhD thesis at Harvard University* (1969).

[10] Winfield D. H. “Function minimization by interpolation in a data table”. In: *Journal of the Institute of Mathematics and its Applications* 12 (1973), pp. 339–347. DOI: 10.1093/imamat/12.3.339.

[11] Shi H-J M. et al. “On the Numerical Performance of Derivative-Free Optimization Methods Based on Finite-Difference Approximations”. In: *Optimization Methods and Software* 38.2 (2022). DOI: 10.48550/arXiv.2102.09762.

[12] Grapiglia G.N. “Worst-case evaluation complexity of a derivative-free quadratic regularization method”. In: *Optimization Letters* (2023), pp. 1–19. DOI: 10.1007/s11590-023-01984-z.

[13] Zhigljavsky A. *Theory of global random search*. Vol. 65. Springer Science & Business Media, 2012. ISBN: 978-94-010-5519-2.

[14] Nesterov Y. and Spokoiny V. “Random Gradient-Free Minimization of Convex Functions”. In: *Foundations of Computational Mathematics* 17 (2017), pp. 527–566. DOI: 10.1007/s10208-015-9296-2.

[15] Van Dyke B. and Asaki T. J. “Using QR decomposition to obtain a new instance of mesh adaptive direct search with uniformly distributed polling directions”. In: *Journal of Optimization Theory and Applications* 159.3 (2013), pp. 805–821. DOI: 10.1007/s10957-013-0356-y.

[16] Bandeira A. S. et al. “Convergence of trust-region methods based on probabilistic models”. In: *SIAM Journal on Optimization* 24.3 (2014), pp. 1238–1264. DOI: 10.1137/130915984.

[17] E. W. Karas et al. “Global convergence of trust-region algorithms for constrained minimization without derivatives”. In: *Applied Mathematics and Computation* 220 (2013), pp. 324–330. DOI: 10.1016/j.amc.2013.06.041.

[18] Neculai A. “An Unconstrained Optimization Test Functions Collection”. In: *Advanced Modeling and Optimization* 10.1 (2008), pp. 147–161.

[19] Conn A.R. et al. *Trust region methods*. MPS/SIAM Series on Optimization. SIAM, 2000. ISBN: 978-0-89871-460-9.

[20] Aravkin A. Y. et al. “A proximal quasi-Newton trust-region method for nonsmooth regularized optimization”. In: *SIAM Journal on Optimization* 32.2 (2022), pp. 900–929. DOI: 10.48550/arXiv.2103.15993.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**