DERIVATIVE FREE MULTILEVEL OPTIMIZATION METHODS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BENGİSEN PEKMEN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
SCIENTIFIC COMPUTING

AUGUST 2009

Approval of the thesis:

DERIVATIVE FREE MULTILEVEL OPTIMIZATION METHODS

submitted by **BENGİSEN PEKMEN** in partial fulfillment of the requirements for the degree of
Master of Science in Scientific Computing Department, Middle East Technical University by,

Prof. Dr. Ersan Akyıldız _____
Dean, Graduate School of **Applied Mathematics**

Prof. Dr. Bülent Karasözen _____
Head of Department, **Scientific Computing**

Prof. Dr. Bülent Karasözen _____
Supervisor, **Department of Mathematics, METU**

Assist. Prof. Dr. Ömür Uğur _____
Co-supervisor, **Institute of Applied Mathematics, METU**

Examining Committee Members:

Prof. Dr. Gerhard Wilhelm Weber _____
Institute of Applied Mathematics, METU

Prof. Dr. Bülent Karasözen _____
Department of Mathematics, METU

Assist. Prof. Dr. Songül Kaya Merdan _____
Department of Mathematics, METU

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: BENGİSEN PEKMEN

Signature :

ABSTRACT

DERIVATIVE FREE MULTILEVEL OPTIMIZATION METHODS

Pekmen, Bengisen
M.S., Department of Scientific Computing
Supervisor : Prof. Dr. Bülent Karasözen
Co-Supervisor : Assist. Prof. Dr. Ömür Uğur

August 2009, 85 pages

Derivative free optimization algorithms are implementations of trust region based derivative-free methods using multivariate polynomial interpolation. These are designed to minimize smooth functions whose derivative are not available or costly to compute. The trust region based multilevel optimization algorithms for solving large scale unconstrained optimization problems resulting by discretization of partial differential equations (PDEs), make use of different discretization levels to reduce the computational cost. In this thesis, a derivative free multilevel optimization algorithm is derived and its convergence behavior is analyzed. The effectiveness of the algorithms is demonstrated on a shape optimization problem.

Keywords: Trust region method, multivariate interpolation, derivative free optimization, multilevel optimization

ÖZ

TÜREVSİZ ÇOK KATMANLI ENİYİLEME YÖNTEMLERİ

Pekmen, Bengisen
Yüksek Lisans, Bilimsel Hesaplama Programı
Tez Yöneticisi : Prof. Dr. Bülent Karasözen
Ortak Tez Yöneticisi : Yrd. Doç. Dr. Ömür Uğur

Ağustos 2009, 85 sayfa

Çok değişkenli interpolasyona dayalı, güvenilir bölge tabanlı türevsiz eniyileme yöntemleri, türevi olmayan ya da yaklaşık olarak türevi hesaplanamayan, düzgün fonksiyonların eniyilemesi için tasarlanmıştır. Kısmi türevli diferansiyel denklemlerin ayrıklaştırılmaları sonucu elde edilen büyük boyuttaki kısıtlamasız eniyileme problemleri için çok katmanlı eniyileme yöntemleri kullanılmaktadır. Bu çalışmada geliştirilen türevsiz çok katmanlı eniyileme yönteminin yakınsama analizi yapılmış, bir geometrik eniyileme problemine uygulanarak elde edilen sayısal sonuçlar verilmiştir.

Anahtar Kelimeler: güvenilir bölge yöntemi, çok değişkenli interpolasyon, türevsiz eniyileme, çok katmanlı eniyileme

To my mother and father

ACKNOWLEDGMENTS

I thank my supervisor Prof. Dr. Bülent Karasözen for giving me the possibility to do the necessary research work and for guiding me through my graduate study in our institute.

I also want to thank Prof. Dr. Stefan Ulbrich in Mathematics Department of Darmstadt Technical University in Germany for helping me to develop some new ideas to progress my master study. And I thank my collaborator Hamdullah Yücel during this scope of DAAD project studies in Darmstadt Technical University.

It is a great pleasure that I have now the opportunity to express my gratitude to my mother Habibe Pekmen and my father Selçuk Pekmen for their limitless love and for letting me free in my choices.

Thanks also Prof. Dr. Gerhard Wilhelm Weber, Assist. Prof. Dr. Ömür Uğur and Assist. Prof. Dr. Hakan Öktem for giving me worthy informations about some other areas of applied mathematics and for their kindness during my master study. I also would like to thank the other members of Institute of Applied Mathematics.

Finally, I owe special thanks to my friends who gave me huge support to complete this thesis.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

For continuously differentiable functions, the standard mathematical characterization of a local minimum, given by first-order necessary conditions, requires that the first-order derivatives are zero. However, for a variety of reasons there have always been many cases when derivatives are unavailable or unreliable. Therefore, nonlinear optimization techniques called derivative-free optimization methods has been needed. Optimization without derivatives is one of the most important, open and challenging areas in computational science and engineering due to increasing complexity in mathematical modelling [13], [24]. The another reason for derivative free methods is inappropriateness of some derivative approximations. For instance, automatic differentiation techniques can not be applied in all cases. In particular, if the objective function is computed using black-box simulation package, automatic differentiation is impossible like in computational fluid dynamics problems. Moreover, applying finite difference derivative approximation may not be accurate when the function evaluations are costly and when there are noisy.

Derivative free optimization (DFO) methods build models of functions based on sample function value or they directly exploit a sample set of function values without building an explicit model. When the model construction is considered, models is typically built by polynomial interpolation or regression. Most algorithms proposed in search on DFO methods are based on trust-region techniques. Trust-region methods minimize trust region subproblems defined by the constructed models. For every minimization, a step size is found. By this step size, a possible reduction is investigated. Meanwhile, control of the geometry of the sample sets, where the function is evaluated, is guaranteed by any condition for geometry (poisedness).

Large-scale finite dimensional optimization problems often arise from the discretization of

infinite-dimensional problems, a primary example being optimal control problems in terms of either ordinary or partial differential equations. While the direct solution of such problems for a discretization level is often possible using the existing packages for large-scale numerical optimization, this technique typically makes very little use of the fact that the underlying infinite-dimensional problem may be described at several discretization levels; the approach thus rapidly becomes inconvenient. To make explicit use of this fact, a class of trust-region methods is used. The algorithms in this class make use of the discretization level as a means of speeding up the computation of the step. This use is recursive, leading to true multilevel optimization methods [21], [22], [57]. A simple first approach of using the different levels of discretization for an infinite-dimensional problem is to use coarser grids in order to compute approximate solutions which can then be used as starting points for the optimization problem on a finer grid.

In our study, our aim is to combine multilevel structure with the derivative free optimization. Therefore, we developed an algorithm called multilevel derivative free optimization (MDFO). Global convergence to first-order critical points is proved under assumptions. Numerical results of different probable versions of MDFO on a shape optimization problem are also presented. When we compare these results with each other, we obtained accurate results considering function evaluations.

The thesis is organized as follows:

- Chapter 2 introduces derivative-free optimization methods.
- Chapter 3 describes multilevel optimization methods.
- Chapter 4 is devoted to MDFO.
- Chapter 5 consists of numerical results for the shape optimization problem.
- Chapter 6 reaches a conclusion and gives the future perspectives.

CHAPTER 2

DERIVATIVE FREE OPTIMIZATION METHODS

2.1 Introduction

A class of nonlinear optimization methods called derivative free methods has been extensively developed in the past decade [13], [24]. There is a high demand especially in the industrial applications because of the expensiveness and difficulty or the unavailability of derivatives of the objective function $f(x)$. Unavailability of derivatives occurs when the computation of $f(x)$ at a given point x results from some physical, chemical or econometrical experiment or measurement, or is a result of large and expensive computer simulation for which the source code is either unavailable or unmodifiable which can be considered as a *black box*.

The idea of derivative free methods for minimization was firstly remarked by Wright such that direct search methods first appear to have been suggested in the 1950's [65]. The first simplex based search was that of Spendley *et al.* [50], which like the work of Torczon [59], preserved the essential geometry. This was improved on the grounds of efficiency by no longer insisting on maintaining the regularity of the simplex, first by Campey and Nickols [7], and then by Nelder and Mead [32]. In 1964, Powell [37] described a method for solving the nonlinear minimization problem based on the use of conjugate gradients. The properties of the method were analyzed by Brent [5].

Independently, Winfield used the available objective function values $f(x_i)$ for building a quadratic model by interpolation [63], [64]. This model is assumed to be valid in a neighbourhood of the current iterate, which is described as a trust region (a hypersphere centered at x_i), whose radius is iteratively adjusted. The model is then minimized within the trust region.

A strong impact on derivative free optimization has been Powell's views. The first major con-

tribution of Powell was a method for solving the nonlinear unconstrained minimum problem based on the use of conjugate gradients [37]. But, recognizing the difficulty of the conjugate-directions method which suffers from determining near conjugate directions when the Hessian of the function is ill-conditioned, Powell suggested using the orthogonal transformations of sets of conjugate directions [40]. Following this idea, he proposed to approximate the matrix of second derivatives by modifying an initial estimate [41].

The most remarkable contribution of Powell was the proposal of linear multivariate interpolation for the approximation of the objective function and constraints in a constrained optimization problem [42]. Improving on this idea, Powell then used a multivariate quadratic interpolation model of the objective function in a trust-region framework in an unconstrained case [43], similar to Winfield's proposals.

There are four mainly classes of derivative-free optimization (DFO) methods [24]. The first class DFO algorithms are *direct search* or *pattern search* methods which are based on the exploration of the variable space by using sample points from a predefined class geometric pattern and use either the Nelder-Mead simplex algorithm or parallel direct search algorithm [43], [65]. The inherit of smoothness of the objective function is not used in these methods. Therefore, a very large number of function evaluations are required. They can be useful for non-smooth problems. The second class of DFO's are line search methods which consists of a sequence of $n+1$ one-dimensional searches introduced by Powell [37]. The third class of the algorithms is constituted by the combination of finite difference techniques (introduced by [16]) coupled with quasi-Newton method [38]. The last class of the methods are based on modeling the objective function by multivariate interpolation in combination with the trust-region techniques. And we will be interested in this class step by step [2], [46], [48].

2.2 Description of the DFO Algorithms

As a motivation to DFO methods, there is an objective function $f : \mathbb{R}^n \to \mathbb{R}$ which is (smooth) nonlinear, bounded below and derivatives of it are not available. The problem is an unconstrained minimization of the form

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

where $\nabla f(x)$ and $\nabla^2 f(x)$ can not be computed for any x.

DFO algorithms belong to the class of trust region methods which are iterative and built around the current iterate as a cheaper model of the objective function which is easier to minimize than the objective function [24]. Therefore, the first main ingradient of a DFO algorithm is to choose an objective function model.

At the kth step the quadratic model within the trust-region $\mathcal{B}_k$,

$$\mathcal{B}_k \stackrel{def}{=} \{x_k + s \ : \ s \in \mathbb{R}^n \text{ and } \|s\| \leq \Delta_k\},$$

with the trust-region radius Δ_k and $\|.\|$ is Euclidean norm, may be given as

$$m_k(x_k + s) = f(x_k) + \langle g(x_k), s\rangle + \frac{1}{2}\langle s, H(x_k)s\rangle \tag{2.1}$$

for some $g \in \mathbb{R}^n$ and some symmetric $n \times n$ matrix H, where $\langle ., .\rangle$ denotes the inner product. The vector g and H do not necessarily correspond to the first and second derivatives of the objective function f. They are determined by requiring that the model (2.1) interpolates the function f at a set $Y = \{y_i\}_{i=1}^{|Y|}$ of points containing the current iterate x_k

$$f(y_i) = m_k(y_i) \quad \text{for all } y_i \in Y \tag{2.2}$$

where Y denotes the set of interpolation points, which is a subset of the set of points at which the values of f is known, including the current iterate. To build the full quadratic model (2.1), the determination of $f(x_k)$, the components of the vector g_k and the entries of the matrix H_k are required. Besides, the cardinality of Y must be equal to

$$p = \frac{1}{2}(n + 1)(n + 2).$$

However, if $n > 1$, the condition (2.2) is not sufficient for the existence and uniqueness of the interpolant and to guarantee the good quality of the model. To ensure that the existence and uniqueness of the interpolant (namely, that the points of Y do not collapse into a lower dimensional subset or lie on a quadratic curve), geometric conditions known as poisedness on the set Y are added to the conditions (2.2).

The general form of the DFO algorithm can be given as follows with the given constants [24],

$$0 < \eta_0 \leq \eta_1 < 1 \quad \text{and} \quad 0 < \gamma_0 \leq \gamma_1 < 1 \leq \gamma_2,$$

Step 1: Initializations.

Let x_s and $f(x_s)$ be given. Choose an interpolation set Y containing x_s.

Determine $x_0 \in Y$ such that $f(x_0) = \min_{y_i \in Y} f(y_i)$.

Choose an initial trust region radius $\Delta_0 > 0$. Set $k = 0$.

Step 2: Build the model.

Using the poised interpolation set Y, build the model $m_k(x_k + s)$.

Step 3: Minimize the model within the trust region.

Compute the point x_k^+ such that

$$m_k(x_k^+) = \min_{x \in \mathcal{B}_k} m_k(x).$$

Compute $f(x_k^+)$ and the ratio

$$\rho_k \stackrel{def}{=} \frac{f(x_k) - f(x_k^+)}{m_k(x_k) - m_k(x_k^+)}.$$

Step 4: Update the interpolation set.

- *Successful step :*

  If $\rho_k \geq \eta_1$, include x_k^+ in Y by dropping one of the existing interpolation points.

- *Unsuccessful step :*

  If $\rho_k < \eta_1$ and Y is inadequate in $x \in \mathcal{B}_k$, improve the geometry.

Step 5: Update the trust-region radius.

- *Successful step:* If $\rho_k \geq \eta_1$, then set

$$\Delta_{k+1} \in [\Delta_k, \gamma_2 \Delta_k].$$

- *Unsuccessful step:* If $\rho_k < \eta_1$ and Y is adequate in $\mathcal{B}_k$, then set

$$\Delta_{k+1} \in [\gamma_0 \Delta_k, \gamma_1 \Delta_k].$$

- Otherwise, set $\Delta_{k+1} = \Delta_k$.

Step 6: Update the current iterate.

Determine $\hat{x}_k$ such that

$$f(\hat{x}_k) = \min_{y_i \in Y, y_i \neq x_k} f(y_i).$$

Then, if

$$\hat{\rho}_k \stackrel{def}{=} \frac{f(x_k) - f(\hat{x}_k)}{m_k(x_k) - m_k(x_k^+)} \geq \eta_0,$$

set $x_{k+1} = \hat{x}_k$. Otherwise, $x_{k+1} = x_k$. Increment k by one ($k := k + 1$) and go to Step 2.

2.3 Interpolating Models

To achieve better local convergence rates in general and to capture the curvature information of the function, it is essential to consider the nonlinear models instead of linear models. As the simplest nonlinear model which is often used, the quadratic polynomial model can be considered. Non-polynomial models such as radial basis functions can also be used in DFO [13].

If a model is a truncated Taylor series expansion of first or second order, then the quality of the model is derived from Taylor expansion error bounds. If the model is an polynomial interpolation based, there exist similar bounds, but, unlike Taylor expansion bounds, they do depend not only on the center of the expansion and on the function that is being approximated, but also on the set of interpolation points. Thus, in order to maintain the quality of the interpolation model, it is essential to maintain the quality of the interpolation set [13].

2.3.1 Polynomial Interpolation

Consider $\mathcal{P}_n^d$ the space of polynomials of degree less than or equal to d in $\mathbb{R}^n$ [13].

Given a set $Y = \left\{y^0, \ldots, y^p\right\} \subset \mathbb{R}^n$ of interpolation points and a basis $\phi = \left\{\phi_0(x), \ldots, \phi_p(x)\right\}$ of $\mathcal{P}_n^d$ which is a set of $p_1 = p + 1$ polynomials of degree less than or equal to d that span $\mathcal{P}_n^d$. Then any polynomial $m(x)$ of degree at most d in $\mathbb{R}^n$ that interpolates a given function $f(x)$ at the points in Y can be written as

$$m(x) = \sum_{j=0}^{p} \alpha_j \phi_j(x).$$

The coefficients $\alpha_0, \ldots, \alpha_p$ can be determined from the interpolation conditions

$$m(y^i) = \sum_{j=0}^{p} \alpha_j \phi_j(y^i) = f(y^i), \quad i = 0, \ldots, p.$$

In matrix form, the following linear system can be written by these conditions

$$M(\phi, Y)\alpha_\phi = f(Y),$$

where

$$M(\phi, Y) = \begin{pmatrix} \phi_0(y^0) & \phi_1(y^0) & \cdots & \phi_p(y^0) \\ \phi_0(y^1) & \phi_1(y^1) & \cdots & \phi_p(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_0(y^p) & \phi_1(y^p) & \cdots & \phi_p(y^p) \end{pmatrix}, \quad \alpha_\phi = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_p \end{pmatrix}, \quad f(Y) = \begin{pmatrix} f(y^0) \\ f(y^1) \\ \vdots \\ f(y^p) \end{pmatrix}.$$

If the matrix $M(\phi, Y)$ is nonsingular, then the system will have a unique solution.

Definition 2.3.1 *The set Y is **poised** for polynomial interpolation in $\mathbb{R}^n$ if the corresponding matrix $M(\phi, Y)$ is nonsingular for some basis ϕ in $\mathcal{P}_n^d$, in other words, if $det(M(\phi, Y)) \neq 0$.*

Example 2.3.2 *Suppose $n = 2$ and Y is a set of six points on a unit circle. Then Y cannot be interpolated by a polynomial of the form*

$$a_0 + a_1x_1 + a_2x_2 + a_{1,1}x_1^2 + a_{1,2}x_1x_2 + a_{2,2}x_2^2,$$

hence, Y is not poised with respect to the space of quadratic polynomials.
On the other hand, Y can be interpolated by a polynomial of the form

$$a_0 + a_1x_1 + a_2x_2 + a_{1,1}x_1^2 + a_{1,2}x_1x_2 + a_{1,1,1}x_1^3,$$

thus, Y is poised in an appropriate subspace of the space of cubic polynomials [49].

Definition 2.3.3 *A set of points Y is called **well-poised**, if it remains poised under small perturbations, i.e., if $det(M(\phi, Y))$ is sufficiently large and the matrix $M(\phi, Y)$ is well-conditioned.*

Theoretically, if the set Y is poised, then we can solve the linear system and find the interpolation polynomial. Nevertheless, numerically the matrix $M(\phi, Y)$ is often ill-conditioned even when it is nonsingular. Obviously, conditioning of $M(\phi, Y)$ depends on the choice of the basis $\{\phi_i(x)\}$ [48]. Because of this, the condition number of $M(\phi, Y)$ generally is not an indicator of well-poisedness.

2.3.2 Lagrange Polynomials

Definition 2.3.4 *In a set of interpolation points* $Y = \left\{y^0, y^1, \ldots, y^p\right\}$*, a basis of* $p_1 = p + 1$ *polynomials* $l_j(x)$, $j = 0, \ldots, p$, *in* $\mathcal{P}_n^d$ *is called a basis of Lagrange polynomials if*

$$l_j(y^i) = \delta_{i,j} = \begin{cases} 0, & i \neq j \\ 1, & i = j. \end{cases}$$

If Y is poised, then the basis of Lagrange polynomials exist and is uniquely defined [13].

For any function $f : \mathbb{R}^n \to \mathbb{R}$ and any poised set $Y = \left\{y^0, y^1, \ldots, y^p\right\} \subset \mathbb{R}^n$, the unique polynomial $m(x)$ that interpolates $f(x)$ on Y can be expressed as

$$m(x) = \sum_{i=0}^{p} f(y^i) l_i(x),$$

where $\{l_i(x), i = 0, \ldots, p\}$ is the basis of Lagrange polynomials for Y [13].

Example 2.3.5 *Consider interpolating the cubic function*

$$f(x_1, x_2) = x_1 + x_2 + 2x_1^2 + 3x_2^3$$

at the six interpolating points $y^0 = (0, 0), y^1 = (1, 0), y^2 = (0, 1), y^3 = (2, 0), y^4 = (1, 1), y^5 = (0, 2)$. *So,* $f(y^0) = 0, f(y^1) = 3, f(y^2) = 4, f(y^3) = 10, f(y^4) = 7, f(y^5) = 26$. *The corresponding Lagrange polynomials can be found by thinking* $l_j(x_1, x_2) = a_0 + a_1x_1 + a_2x_2 + a_3x_1^2 + a_4x_1x_2 + a_5x_2^2$ *where* $a_0, \ldots, a_5$ *are coefficients which can be found by using the definition 2.3.4. After calculations,*

$$\begin{aligned} l_0(x_1, x_2) &= 1 - \tfrac{3}{2}x_1 - \tfrac{3}{2}x_2 + \tfrac{1}{2}x_1^2 + \tfrac{1}{2}x_2^2 + x_1x_2 \\ l_1(x_1, x_2) &= 2x_1 - x_1^2 - x_1x_2 \\ l_2(x_1, x_2) &= 2x_2 - x_2^2 - x_1x_2 \\ l_3(x_1, x_2) &= -\tfrac{1}{2}x_1 + \tfrac{1}{2}x_1^2 \\ l_4(x_1, x_2) &= x_1x_2 \\ l_5(x_1, x_2) &= -\tfrac{1}{2}x_2 + \tfrac{1}{2}x_2^2. \end{aligned}$$

Thus, the model function can be written as

$$\begin{aligned} m(x_1, x_2) &= 0l_0(x_1, x_2) + 3l_1(x_1, x_2) + 4l_2(x_1, x_2) + 10l_3(x_1, x_2) \\ &+ 7l_4(x_1, x_2) + 26l_5(x_1, x_2). \end{aligned}$$

One of the most useful features of Lagrange polynomials is an upper bound which is a measure of poisedness of interpolation set Y in a region $\mathcal{B}$. It's shown for any x in the convex hull [1]

$$\left\|\mathcal{D}^r f(x) - \mathcal{D}^r m(x)\right\| \leq \frac{1}{(d+1)!}\nu_d \sum_{i=0}^{p} \left\|y^i - x\right\|^{d+1} \left\|\mathcal{D}^r l_i(x)\right\|$$

where $\mathcal{D}^r$ denotes r-th derivative of a function, ν_d is an upper bound on $\mathcal{D}^{d+1} f(x)$ and $\|.\|$ is Euclidean norm. Note that $f(x)$ is required to have a bounded $(d+1)$st derivative by this bound. When $r = 0$, the bound will be

$$|f(x) - m(x)| \leq \frac{1}{(d+1)!} p_1 \nu_d \Lambda_l \Delta^{d+1},$$

where

$$\Lambda_l = \max_{0 \leq i \leq p} \max_{x \in \mathcal{B}(Y)} |l_i(x)|,$$

and Δ is the diameter of the smallest ball $\mathcal{B}(Y)$ containing Y. Here, Λ_l is related to the Lebesgue constant of Y [13].

Alternative ways to define Lagrange polynomials can be found and defined as in [13].

Λ - *poisedness*

A measure of poisedness should reflect how well interpolation set *spans* the region where interpolation is of interest [13].

Well poisedness and poisedness of Y also has to depend on the polynomial space from which an interpolant is chosen (Example 2.3.2).

Λ-poisedness can be defined as in [13] :

Definition 2.3.6 *Let $\Lambda > 0$ and a set $B \in \mathbb{R}^n$ be given. Let $\phi = \left\{\phi_0(x), \phi_1(x), \ldots, \phi_p(x)\right\}$ be a basis in $\mathcal{P}_n^d$. A poised set $Y = \left\{y^0, \ldots, y^p\right\}$ is said to be Λ-poised in B (in the interpolation sense) if and only if*

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

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

or, equivalently,

[1] The convex hull of a given set $S \subset \mathbb{R}^n$ is the smallest convex set containing S (meaning that it is the intersection of all convex sets containing S). The convex hull of a set is always uniquely defined and consists of all convex combinations of elements of S, i.e., of all linear combinations of elements of S whose scalars are nonnegative and sum up to one [47].

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

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

or, equivalently,

3. *replacing any point in* Y *by any* $x \in B$ *can increase the volume of the set* $\left\{\phi(y^0), .., \phi(y^p)\right\}$ *at most by a factor* Λ.

2.3.3 Newton Fundamental Polynomials

The basis of quadratic functions and the point set Y is given such a way that $M(\phi, Y)$ has the following structure [48] :

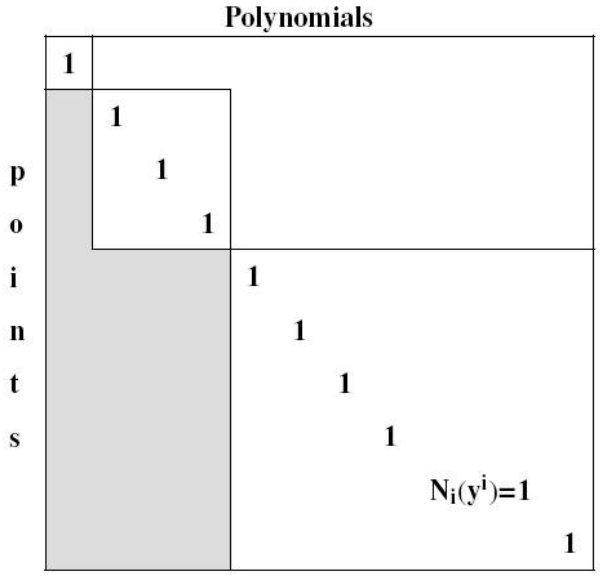

Figure 2.1: Newton Fundamental Polynomial

Nonzero coefficients of basis $\phi_i(x^j) \neq 0$ occur iff i, j belong to the same subset :

$$\{1\}, \quad \{x_1, \ldots, x_n\}, \quad \left\{x_1^2, \ldots, x_n^2, x_1x_2, \ldots, x_{n-1}x_n\right\}$$

The set of interpolation points is partitioned into three *blocks* Y^0, Y^1, Y^2 which correspond to constant terms, linear terms and quadratic terms. Hence, Y^0 has a single element, Y^1 has n elements and Y^2 has $n(n+1)/2$ elements [48].

The basis NFP $\{N_i(.)\}$ is also parititoned into three blocks $\left\{N_i^0(.)\right\}, \left\{N_i^1(.)\right\}, \left\{N_i^2(.)\right\}$ with the appropriate number of elements in each block like Ys. So, the basis elements and the interpolation points are set in one-to-one correspondence.

A Newton polynomial $N_i(.)$ and a point y^i correspond $\Longleftrightarrow N_i(y^i) = 1$ and $N_i(y^j) = 0$ for all index j within first l blocks.

Gram-Schmidt orthogonalization procedure to obtain NFP basis with, for example, the basis of monomials [48] :

- ⋆ *Initialize* $\left\{\bar{N}_i(.)\right\}_{i=1}^{p}$
- ⋆ *For i=1,. . . , p* :
  - *Normalize* : $\bar{N}_i(x) \leftarrow \frac{\bar{N}_i(x)}{\bar{N}_i(y^i)}$
  - *Orthogonalize* : $\bar{N}_j(x) \leftarrow \bar{N}_j(x) - \bar{N}_j(y^j)\bar{N}_i(x)$
    for j in the same or "later" block as i
- ⋆ *Set* $N_i(.) = \bar{N}_i(.), i = 1, \ldots, p.$

Here, for every step i the value $\bar{N}_i(y^i)$ is called the *pivot value*. 'bar' is used to show intermediate polynomials and to distinguish them from NFP's $N_i(.)$, obtained at the end of the procedure.

Example 2.3.7 *Consider* $Y^0 = \{(0,0)\}, \quad Y^1 = \{(1,0),(0,1)\}, \quad Y^2 = \{(2,0),(1,1),(0,2)\}$ *corresponding to the initial basis functions* $1, x_1, x_2, x_1^2, x_1x_2, x_2^2$ *respectively.*
Applying the above procedure, we obtain

$$N_1 = 1, \quad N_2 = x_1, \quad N_3 = x_2$$

$$N_4 = \frac{1}{2}(x_1^2 - x_1), \quad N_5 = x_1x_2, \quad N_6 = \frac{1}{2}(x_2^2 - x_2)$$

The set Y is poised if and only if the pivot value is nonzero. However, for numerical purposes it is important that $\left|N_i(y^i)\right|$ is sufficiently large (not very close to zero). Small pivot values result in large coefficients of NFPs (due to the normalization step) and lead to numerical instability.

When a neighbourhood B and a constant K (independent of B) is given, an interpolation set Y is *well-poised* if there exists a corresponding basis of Newton fundamental polynomials such that $|N_i(x)| \leq K$ for all $i = 1, \ldots, p$ and all x in B [48].

In any other way as in [8], in practice ,

$$\left|\bar{N}_i(y^i)\right| \geq \theta,$$

θ is called a *pivoting threshold*. Also, if this condition is satisfied, then the NFPs are said to be well poised. Moreover, this bound shows that interpolation is 'locally good' and provides a first order approximation of $f(x)$.

One of the advantages of the threshold pivoting is that it prevents points that are very close to each other to be included in the interpolation set, but rather it has a 'spreading' effect on the interpolation points. This helps to maintain the noise and it is one of the major factors responsible for the robustness of DFO on noisy problems [48].

In case 'bad' pivot is encountered during getting NFP procedure, typically all possible remaining pivots should be considered and a pivot that passes the threshold is chosen. If no such pivot exists, the procedure is terminated and the interpolation set Y is restricted only to the points which correspond to completed pivots.

Thus, one may exit with an incomplete interpolation set and with a set of Newton polynomials which span only some subspace of the space of quadratic polynomials. In this case, an interpolation can still be constructed in the subspace.

Above procedure can also be applied when the initial cardinality of the interpolation set $p < (n + 1)(n + 2)/2$. The procedure terminates once p NFPs have been constructed. Again, the outcome of this is an incomplete basis of Newton polynomials.

The subspace spanned by an incomplete basis depends on the choice of the initial basis. For example [48], if $n = 2$ and $p = 5 < (n + 1)(n + 2)/2 = 6$, and the initial basis is $\{1, x_1, x_2, x_1^2, x_1x_2, x_2^2\}$, then the final incomplete Newton polynomials will span the subset of quadratic function of the form $a_0 + a_1x_1 + a_2x_2 + a_3x_1^2 + a_4x_1x_2$.

As an alternative way to complete an interpolation set to a full well-poised set, one can continue procedure by picking an arbitrary interpolation point from the appropriate neighbourhood, so that the value of the pivot can pass the threshold. For example, a point $y_- = y_i \neq x_k$

is replaced by another point y_+ such that

$$y_+ = \underset{y \in \mathcal{B}_k}{\operatorname{argmax}} \; |N(y)|,$$

provided that $|N_i(y_+)| + 1 \geq \theta$ [9].

2.3.4 Regression Nonlinear Models

When there is noisy data, the interpolation of objective function is generally replaced by least-squares regression. In this case, the interpolation conditions $M(\phi, Y)\alpha = f(Y)$ are solved in the least-squares sense, meaning that a solution is found such that $\|M(\phi, Y)\alpha - f(Y)\|_2^2$ is minimized [13].

2.3.5 Underdetermined Interpolating Models

When the case that the number of interpolation points in Y is smaller than the number of elements in the polynomial basis ϕ is considered, the matrix $M(\phi, Y)$ defining the interpolation conditions has more columns than rows and the interpolation polynomials defined by

$$m(y^i) = \sum_{k=0}^{q} \alpha_k \phi_k(y^i) = f(y^i), \quad i = 0, \ldots, p,$$

are no longer unique [13].

2.4 Trust-Region Methods for Derivative Free Models

Trust region methods are a most common area of research for the solutions of nonlinear programming problems. Attractive features of trust-region methods are :

- They are based on quadratic models. This provides to deal with curvature information.
- The convergence properties of trust-region methods is both comprehensive and elegant to use in many problems.

This section is based on [2], [13]. The main idea is to use a model for the objective function which is in a neighborhood of the current point. The neighborhood is called *trust-region* which can be defined as

$$\mathcal{B}(x_k; \Delta_k) \stackrel{def}{=} \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$$

for every kth iteration Δ is the trust-region radius and $\|.\|$ can be taken as iteration dependent norm (here, Euclidean norm). Typically, the model is written in the form

$$m_k(x_k + s) = m_k(x_k) + s^T g_k + \frac{1}{2} s^T H_k s, \tag{2.3}$$

where $g_k = \nabla Q_k$ and $H_k = \nabla^2 Q_k$. If Q_k is the first-order Taylor model, then $Q_k(x_k) = f(x_k)$ and $g_k = \nabla f(x_k)$. For second order derivatives, we write $H_k = \nabla^2 f(x_k)$. In the derivative-free case, models are used where $H_k \neq \nabla^2 f(x_k), g_k \neq \nabla f(x_k)$. If interpolation is absent, $Q_k(x_k) \neq f(x_k)$.

In the unconstrained case, the local model problem which is called the trust region subproblem will be

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

where $m_k(x_k + s)$ is the model in (2.3) and $\mathcal{B}(0; \Delta_k)$ is the trust region of radius Δ_k centered at 0 and $s = x - x_k$.

The basic trust region algorithm [2]:

1. **Initialization**

   An initial point x_0 and an initial trust region Δ_0 are given.

   The constants $\eta_1, \eta_2, \gamma_1, \gamma_2$ are also given and satisfy

   $$0 < \eta_1 \leq \eta_2 < 1 \quad \text{and} \quad 0 < \gamma_1 \leq \gamma_2 < 1,$$

Compute $f(x_0)$ and set $k = 0$.

2. **Model definition**

   Choose the norm $\|.\|_k$ and define a model m_k in B_k.

3. **Step Computation**

   Compute a step s_k that "sufficiently reduces the model" m_k and such that $x_k + s_k \in B_k$.

4. **Acceptance of the trial point**

   Compute $f(x_k + s_k)$ and define

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

   If $\rho_k \geq \eta_1$ then define $x_{k+1} = x_k + s_k$; otherwise define $x_{k+1} = x_k$.

5. **Trust region radius update**

   Set

$$\Delta_{i+1}^{+} \in \begin{cases} [\Delta_k, \infty) & \text{if } \rho_k \geq \eta_2 \\ [\gamma_2 \Delta_k, \Delta_k) & \text{if } \rho_k \in [\eta_1, \eta_2) \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k) & \text{if } \rho_k < \eta_1. \end{cases}$$

   Increment k by one ($k := k + 1$) and go to step 2.

In the following discussions during this section, all norms will be taken as Euclidean norm.

The Cauchy Step

This is the step, s_k^C, to the minimum of the model along the steepest descent direction within the trust region. If, for $t \geq 0$,

$$t_k^C = \underset{x_k - t g_k \in \mathcal{B}(x_k; \Delta_k)}{\operatorname{argmin}} m_k(x_k - t g_k),$$

is defined, then the Cauchy step is given by

$$s_k^C = -t_k^C g_k. \tag{2.4}$$

Theorem 2.4.1 *If the model (2.3) and the Cauchy step (2.4) is considered, then*

$$m_k(x_k) - m_k(x_k + s_k^C) \geq \frac{1}{2} \|g_k\| \min \left[\frac{\|g_k\|}{\|H_k\|}, \Delta_k \right],$$

where $\|g_k\| / \|H_k\| = +\infty$ *is assumed when* $H_k = 0$.

If the function $m_k(x_k + s)$ is not a linear or a quadratic function, Theorem 2.4.1 is not directly applicable.

In fact, it is not necessary to actually find the Cauchy step to achieve global convergence to first-order stationarity. It is sufficient to relate the step computed to the Cauchy step. Thus, it is *assumed* that for all iterations k,

$$m_k(x_k) - m_k(x_k + s_k) \geq \kappa_{fcd} \left[m_k(x_k) - m_k(x_k + s_k^C) \right]$$

for some constant $\kappa_{fcd} \in (0, 1]$. By theorem (2.4.1), equivalently,

$$m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa_{fcd}}{2} \|g_k\| \min \left[\frac{\|g_k\|}{\|H_k\|}, \Delta_k \right].$$

This assumption is the minimum requirement for how well one has to do at solving the trust-region subproblem to achieve global convergence to first-order critical points.

The eigenstep

When a quadratic model and global convergence to second-order critical points are considered, the model reduction that is required can be achieved along a direction related to the greatest negative curvature. Assuming H_k has at least one negative eigenvalue and the most negative eigenvalue of H_k is defined as $\tau_k \stackrel{def}{=} \lambda_{\min}(H_k)$. In this case, a step of negative curvature s_k^E referred to as the eigenstep can be determined by

$$(s_k^E)^T(g_k) \leq 0, \quad \left\| s_k^E \right\| = \Delta_k, \quad \text{and} \quad (s_k^E)^T H_k(s_k^E) = \tau_k \Delta_k^2.$$

The eigenstep s_k^E is the eigenvector H_k corresponding to the most negative eigenvalue τ_k. Note that due to negative curvature, s_k^E is the minimizer of the quadratic function along that direction inside the trust region. The model decreases by the eigenstep in the following way.

Lemma 2.4.2 *Suppose that the model Hessian H_k has negative eigenvalues. Then*

$$m_k(x_k) - m_k(x_k + s_k^E) \geq -\frac{1}{2} \tau_k \Delta_k^2.$$

If $m_k(x_k + s)$ is not a quadratic function, then Lemma 2.4.2 and Theorem 2.4.1 are not directly applicable.

As in Cauchy step,

$$m_k(x_k) - m_k(x_k + s_k) \geq \kappa_{fed} \left[m_k(x_k) - m_k(x_k + s_k^E) \right]$$

for some constant $\kappa_{fed} \in (0, 1]$. To yield a fraction of Cauchy step decrease in here, it is *assumed* for all iterations k,

$$m_k(x_k) - m_k(x_k + s_k) \geq \kappa_{fod} \left[m_k(x_k) - \min \left\{ m_k(x_k + s_k^C), m_k(x_k + s_k^E) \right\} \right]$$

for some constant $\kappa_{fod} \in (0, 1]$.

By using the previous assumption, Lemma 2.4.2 and the Cauchy decrease Theorem 2.4.1,

$$m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa_{fod}}{2} \max \left[\|g_k\| \min \left\{ \frac{\|g_k\|}{\|H_k\|}, \Delta_k \right\}, -\tau_k \Delta_k^2 \right] .$$

Trust-region subproblem

Now, it will be given a detailed analysis how the trust-region subproblem $\min_{s \in \mathcal{B}(0;\Delta_k)} m_k(x_k + s)$ can be solved. For the analysis, the following theorem is necessary [13].

Theorem 2.4.3 *Any global minimizer s_* of $m(x+s) = m(x) + s^T g + \frac{1}{2} s^T H s$, subject to $\|s\| \leq \Delta$, satisfies the equation*

$$[H + \lambda_* I] \, s_* = -g, \qquad (*)$$

where $H + \lambda_* I$ is positive semidefinite, $\lambda_* \geq 0$, and

$$\lambda_* \left(\|s_*\| - \Delta \right) = 0. \qquad (**)$$

If $H + \lambda_* I$ is positive definite, then s_* is unique.

If Δ is large enough and H is positive definite, the complementarity conditions $(**)$ are satisfied with $\lambda_* = 0$ and the unconstrained minimum lies within the trust region. A solution lies on the boundary of the trust region in all other circumstances and $\|s_*\| = \Delta$.

Assume that H has an eigendecomposition

$$H = QEQ^T,$$

where E is a diagonal matrix of eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$, and Q is an orthogonal matrix of associated eigenvectors. Then,

$$H + \lambda I = Q(E + \lambda I)Q^T.$$

Theorem 2.4.3 indicates that the value of investigated λ must satisfy $\lambda \geq -\lambda_1$ (as only then is $H + \lambda I$ positive semidefinite), and, if $\lambda > -\lambda_1$, the model minimizer is unique (as this ensures that $H + \lambda I$ is positive definite).

Suppose that $\lambda > -\lambda_1$. Then $H + \lambda I$ is positive definite, and thus $(*)$ has a unique solution,

$$s(\lambda) = -\left[H + \lambda I\right]^{-1} g = -Q\left(E + \lambda I\right)^{-1} Q^T g.$$

However, the solution which is looked for depends upon the nonlinear inequality

$$\|s(\lambda)\| \leq \Delta.$$

So,

$$\|s(\lambda)\|^2 = \left\|Q\left(E + \lambda I\right)^{-1} Q^T g\right\|^2 = \left\|\left(E + \lambda I\right)^{-1} Q^T g\right\|^2 = \sum_{i=1}^{n} \frac{\gamma_i^2}{(\lambda_i + \lambda)^2},$$

where γ_i is the ith component of $Q^T g$. By this last equality, if $\lambda > -\lambda_1$, then $\|s(\lambda)\|$ is a continuous, nonincreasing function of λ on $(-\lambda_1, +\infty)$ that tends to zero as λ tends to $+\infty$. Moreover, provided $\gamma_j \neq 0$, then $\lim_{\lambda \to -\lambda_j} \|s(\lambda)\| = +\infty$. Thus, provided $\gamma_1 \neq 0$, $\|s(\lambda)\| = \Delta$ for a unique value of $\lambda \in (-\lambda_1, +\infty)$.

When H is positive definite and $\left\|H^{-1} g\right\| \leq \Delta$ the solution corresponds to $\lambda = 0$. Otherwise, when H is positive definite and $\left\|H^{-1} g\right\| > \Delta$ there is a unique solution to $(*)$ in $(0, +\infty)$.

When H is not positive definite and $\gamma_1 \neq 0$ it is needed to find a solution to $(*)$ with $\lambda > -\lambda_1$. Because of high nonlinearities in the neighborhood of $-\lambda_1$, solving the following secular equation becomes preferable;

$$\frac{1}{\|s(\lambda)\|} = \frac{1}{\Delta}.$$

This equation is close to linear in the neighborhood of the optimal λ. Because of the near linearity in the region of interest, the fast convergence of Newton's method is expected. But, since an unsafeguarded Newton method may fail to converge, the method should be safeguarded carefully.

When H is not positive definite and $\gamma_1 = 0$, this is called the hard case, since there is no solution to $(*)$ in $(-\lambda_1, +\infty)$, when $\Delta > \|s(-\lambda_1)\|$. However, there is a solution at $\lambda = -\lambda_1$, but it includes an eigenvector of H corresponding to the eigenvalue λ_1, which thus has to be estimated. Details can be found in [54].

***Fully Linear and Fully Quadratic Models*:**

Let a function f that is differentiable on an open domain (denote this open set, $\acute{L}(x_0) = \underset{x \in L(x_0)}{\cup} \mathcal{B}(x; \Delta_{\max})$ for any fiven x_0 and $\Delta_{\max}$ and where $L(x_0) = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$ is level set) and ∇f is Lipschitz continuous be given. Let κ_f and κ_g be fixed positive constants. For any given $\Delta \in (0, \Delta_{\max})$ and for any given $x \in L(x_0)$, consider a class of model functions $\mathcal{M} = C^1(\mathbb{R}^n, \mathbb{R})$. The class $\mathcal{M}$ is called *fully linear class* on $\mathcal{B}(x; \Delta)$ if for any model function $m \in \mathcal{M}$

$$\|\nabla f(x+s) - \nabla m(x+s)\| \leq \kappa_g \Delta, \qquad \forall s \in \mathcal{B}(0; \Delta)$$

$$|f(x+s) - m(x+s)| \leq \kappa_f \Delta^2, \qquad \forall s \in \mathcal{B}(0; \Delta).$$

A model that belongs to a fully linear class $\mathcal{M}$ and satisfy these two conditions is called *fully linear* on $\mathcal{B}(x; \Delta)$.

Let a function f which is twice continuously differentiable and $\nabla^2 f$ is Lipschitz continuous on an open domain be given. Let $\kappa_f, \kappa_g, \kappa_h$ be positive, fixed constants. For any given $\Delta \in (0, \Delta_{\max})$ and for any fiven $x \in L(x_0)$, consider a class of model functions $\mathcal{M} = C^2(\mathbb{R}^n, \mathbb{R})$. The class $\mathcal{M}$ is called a *fully quadratic* class on $\mathcal{B}(x; \Delta)$ if for any model function $m \in \mathcal{M}$

$$\left\|\nabla^2 f(x+s) - \nabla^2 m(x+s)\right\| \leq \kappa_h \Delta, \qquad \forall s \in \mathcal{B}(0; \Delta)$$

$$\|\nabla f(x+s) - \nabla m(x+s)\| \leq \kappa_g \Delta^2, \qquad \forall s \in \mathcal{B}(0; \Delta)$$

$$|f(x+s) - m(x+s)| \leq \kappa_f \Delta^3, \qquad \forall s \in \mathcal{B}(0; \Delta).$$

Any model m belongs to a fully quadratic class $\mathcal{M}$ and satisfying these three conditions is called *fully quadratic* on $\mathcal{B}(x; \Delta)$.

Note that the model (2.3) must be fully linear in order to ensure global convergence to a first-order critical point.

2.5 Interpolation Based Derivative Free Optimization Methods

2.5.1 DFO Algorithm based on NFP

The DFO method which is used here [48] is described as in [9]. The main distinction between this method and the earlier methods is in the approach to handling the geometry of the interpolation set. NFPs (Newton Fundamental Polynomials) is used to construct the interpolation model.

Step 0 : Initialization

Let a starting point x^m and the value $f(x^m)$ be given.

Choose an initial trust region radius $\Delta_0 > 0$.

Choose at least one additional point[2] not further than Δ_0 away from x^m to create an initial well-poised interpolation set Y and initial basis of NFPs.

Determine $x^0 \in Y$ which has the best objective function value; i.e.,

$f(x^0) = \min_{y^i \in Y} f(y^i)$.

Set $k = 0$. Set parameters η_0, η_1 to measure progress : $0 < \eta_0 < \eta_1 < 1$.

Step 1 : Build the model

Using the interpolation set Y and the basis of NFP, build an interpolation model $Q_k(x)$.

Step 2 : Minimize the model within the trust region

Set $B_k = \left\{x : \left\|x - x^k\right\| \leq \Delta_k\right\}$. Compute the point $\hat{x}^k$ such that

$$Q_k(\hat{x}^k) = \min_{x \in B_k} Q_k(x).$$

Compute $f(\hat{x}^k)$ and the ratio

$$\rho_k \equiv \frac{f(x^k) - f(\hat{x}^k)}{Q_k(x^k) - Q_k(\hat{x}^k)}.$$

Step 3 : Update the interpolation set

- ⋆ If $\rho_k \geq \eta_0$, include $\hat{x}^k$ in Y, dropping one the existing interpolation points if necessary.
- ⋆ If $\rho_k < \eta_0$, include $\hat{x}^k$ in Y, if it improves the quality of the model.

[2] With respect to this algorithm the model is built up from a few points but in Tamas Terlaky's algorithm [52], n additional points are used

- ⋆ If $\rho_k < \eta_0$ and there are less than $n + 1$ points in the intersection of Y and B_k, generate a new interpolation point in B_k, while preserving/improving well-poisedness.
- ⋆ Update the basis of the Newton Fundamental Polynomials.

Step 4 : Update the trust region radius

- ⋆ If $\rho_k \geq \eta_1$, increase the trust region radius

$$\Delta_{k+1} \in [\Delta_k, \gamma_2 \Delta_k].$$

- ⋆ If $\rho_k < \eta_0$ and the cardinality of $Y \cap B_k$ was less than $n + 1$ when $\hat{x}^k$ was computed, reduce the trust region

$$\Delta_{k+1} \in [\gamma_0 \Delta_k, \gamma_1 \Delta_k].$$

- ⋆ Otherwise, set

$$\Delta_{k+1} = \Delta_k.$$

Step 5 : Update the current iterate

Determine $\bar{x}^k$ with the best objective function value

$$f(\bar{x}^k) = \min_{\substack{y^i \in Y \\ y^i \neq x^k}} f(y^i).$$

If improvement is sufficient (w.r.t. predicted improvement)

$$\bar{\rho}^k \equiv \frac{f(x^k) - f(\bar{x}^k)}{Q_k(x^k) - Q_k(\hat{x}^k)} \geq \eta_0,$$

set $x^{k+1} = \bar{x}^k$.

Otherwise, set $x^{k+1} = x^k$.

Increment k by one ($k := k + 1$) and go to Step 1.

In Step 3 and Step 5, the improvement procedure works in the following way. Let $\delta(Y) = \det(M(\phi, Y))$ [17]. And let m_1, m_2, m_3 be the factors of the improvement in $\delta(Y)$.

For $i = 1, \ldots,$ the number of points in a set,

1. Find the furthest point x^f from the current point x^c.

2. Replace x^f by a candidate point x.

3. Calculate

$$\text{ratio} = \frac{\text{determinant after replacement}}{\text{determinant before replacement}}.$$

   If $\left\|x^f - x^c\right\| \leq \Delta$ and ratio $\geq m_2$, then accept x, return.

   Else, if $\left\|x^f - x^c\right\| \leq m_3\Delta$ and ratio $> m_1$, then accept x, return.

   Else, if $\left\|x^f - x^c\right\| \geq m_3\Delta$ and $\delta(Y) > \epsilon$, then accept x, return.

   Else, find the new furhest point $x^{\hat{f}}$ and go to Step 2.

where the norms are Euclidean norm ($\|.\|_2$).

In this implementation, the factor $m_1 = 1, m_2 = 3, m_3 = 4$ are set. These can be modified as needed.

The other implementation parts of the algorithm such as dropping a point, termination criteria will be mentioned in Chapter 4.

It has been proved that DFO is globally convergent to a local minimum, provided that the approximation model is at least fully linear to ensure a reasonable approximate result of the objective function [10].

2.5.2 UOBYQA and CONDOR

UOBYQA is **U**nconstrained **O**ptimization **BY** **Q**uadratical **A**pproximation and CONDOR is **CO**nstrained, **N**onlinear, **D**irect, parallel **O**ptimization using the trust **R**egion method for high-computing load function [2], [3].

The algorithms of UOBYQA and CONDOR are similar. Both of them are based on the same ideas and have nearly the same behavior. CONDOR is an extension of UOBYQA. Small differences is due to the algorithms used inside the main algorithms. In other words, these two

- sample the search space, making evaluations in a way that reduces the influence of the noise, and they
- construct a *fully* quadratic model based on Lagrange Interpolation technique. The curvature information is obtained from the quadratic model. This technique is less sensitive to the noise and leads to high quality local quadratic models which directly guide the search to the nearest local optimum. Using a full quadratic model enables to compute Newton's steps. Newton's steps have a proven quadratic convergence speed. Since some evaluations of the objective function are lost to build the quadratical model, 'nearly' quadratic convergence speed is obtained which can be called Q-superlinear convergence by Powell [44]. These quadratical models are built using the least number of evaluations.

Inside UOBYQA and CONDOR algorithms, the Moré and Sorenson Algorithm [2] is used for the computation of the trust region step.

Moré and Sorensen trust region model is to seek a solution s^* of the minimization problem :

$$\min_{s\in\mathbb{R}^n} \quad Q(x_k + s) \equiv f(x_k) + \langle g_k, s\rangle + \frac{1}{2}\langle s, H_k s\rangle \quad \text{subject to} \quad \|s\|_2 < \Delta.$$

where g_k is the approximation of the gradient of $f(x)$ evaluated at x_k and H_k is the approximation of the Hessian matrix of $f(x)$ evaluated at x_k. This minimization problem can be written by a translation of the polynomial Q_k.

$$\min_{s\in\mathbb{R}^n} \quad Q(s) \equiv \langle g, s\rangle + \frac{1}{2}\langle s, Hs\rangle \quad \text{subject to} \quad \|s\|_2 < \Delta.$$

When CPU time is concerned to reach the local optimum, computer *paralellization* of the function evaluations is a way of CONDOR. CONDOR takes a similar road as in PDS (Parallel direct search) by proposing an extension of the original UOBYQA that several CPUs used in parallel. Numerical results show that this addition makes CONDOR the fastest available algorithm for noisy, high computing load objective functions (fastest, in terms of function evaluations).

UOBYQA and CONDOR are globally convergent to a local (or global) optimum. The detailed information about CONDOR algorithm can be found in [2] and UOBYQA algorithm is in [2] or [3].

2.5.3 Similarities and Differences between CONDOR and DFO

At the same way as [2], [24], [36] :

- DFO uses Newton polynomials while CONDOR uses Lagrange polynomials as the basis for the space of quadratic polynomials.
- When DFO algorithm starts, it builds a fully linear model (using only $n + 1$ evaluations of the objective function) and then directly uses this simple model to guide the research into the space.
- In DFO, when a point is 'too far' from the current position , the model could be *invalid* and could not represent correctly the local shape of the objective function. This 'far point' is rejected and replaced by a closer point. But this operation requires an evaluation of the objective function. Thus, in some situation, it is preferable to lower degree of the polynomial which is used as local model (and drop 'far point'), to avoid this evaluation. Therefore, DFO is using a polynomial of degree oscillating between 1 and a "full" 2. CONDOR uses full quadratic polynomials.
- In CONDOR and UOBYQA the Moré and Sorensen algorithm is used for the computation of the trust region step. Numerically, it is very stable and give very high precision results. In DFO, a general purpose tool NPSOL [3] is used giving high precision results

[3] NPSOL is a set of subroutines for minimizing a smooth function subject to constraints, which may include simple bounds on the variables, linear constraints, and smooth nonlinear constraints. It uses a sequential quadratic programming (SQP) algorithm, in which each search direction is the solution of a QP subproblem [36].

but that can not be compared to the Moré and Sorenson algorithm when the precision is critical.

- In CONDOR (also in UOBYQA) the *validity* of the model is checked using the two following equations :

  – All the interpolation points must be close to the current point x_k:

  $$\left\|x_j - x_k\right\| \leq 2\rho, \quad j = 1, \ldots, N$$

  where ρ is a measure of initial and final value of global trust region radius. This condition prevents the algorithm from sampling the model at $(n+1)(n+1)/2$ new points.

  – Powell's Heuristic:

  $$\frac{M}{6}\left\|x_j - x_k\right\|^3 \max_d \left\{\left|l_j(x_k + d)\right| : \|d\| \leq \rho\right\} \leq \epsilon, \quad j = 1, \ldots, N$$

  where $l_j(x)$ is a basis of Lagrange polynomials and $\left|\frac{d^3}{d\alpha^3} f(x + \alpha v)\right| \leq M$ such that $\|v\| = 1$, $\alpha \in \mathbb{R}$.

  This last equation provides to "keep far points" inside the model, still being assured that it is valid. It allows to have a 'full' polynomial of second degree for a 'cheap price'.

  The checking of the validity of the model in DFO is mainly based on this second equation, which is not very often satisfied by the trust-region radius. Therefore, more function evaluations are needed in order to rebuild the interpolation polynomial in some cases.

2.5.4 NEWUOA (NEW Unconstrained Optimization Approximation)

This approach is to seek the minimum of a function $F(x), x \in \mathbb{R}^n$, which is specified by a subroutine (provided by user) that calculates the value of $F(x)$ for any given vector $x \in \mathbb{R}^n$. There are no constraints and no derivatives are required. NEWUOA is generally suitable for noisy objective functions [46].

Let $Y = \left\{y^1, \ldots, y^m\right\}$ be the interpolation set. Powell focused in particular on models based on $m = 2n + 1$ points. These models are obtained from one iteration to the next by minimizing the Frobenius norm of the change in the Hessian, subject to the interpolation conditions. The complexity of an iteration is very attractive and is of order $O(n^3)$.

Another interesting property is that for any kth iteration, one of the two types of iteration 'trust region' or 'alternative' is chosen. Trust-region iteration works in the same way like basic trust-region. An *alternative iteration* usually tries to improve the quadratic model by moving the interpolation point that is furthest from x^k, where k is still the iteration number.

2.5.5 BOOSTERS (Bierlaire and Oeuvray Optimization STrategy Exploiting Radial Surrogates)

The necessity to develope this algorithm can be described as [33] :

- *Non-polynomial models.* These polynomials have a big potential and a lot of applications could benefit from their use.
- *The multivariate scattered data interpolation problem.* RBFs is a tool to solve this problem. Under some assumptions the existence and uniqueness is guaranteed.
- *Flexibility.* While a full quadratic model is based on exactly $m_1 = n(n + 1)/2 + n + 1$ points, in BOOSTERS' models are based on $m_2 \geq n + 1$. Flexibility comes from the choice the number of the interpolation points by the user.
- *Smoothness and robustness to noise.* Models based on RBFs are insensitive to noise. The value of the objective function can be changed without affecting very much the entire model. Furthermore, RFBs are often known to be smoothing functions and the solutions to variational problems.

In multivariate interpolation based on RBFs, in order to interpolate a function f whose values on a set $Y = \left\{y_1, \ldots, y_p\right\} \subset \mathbb{R}^n$ are known, a function is considered of the form

$$s(x) = \sum_{i=1}^{p} \lambda_i \phi\left(\|x - y_i\|\right) + q(x),$$

where q is a low degree polynomial, $\lambda_i, i = 1, \ldots, p$, are the interpolant parameters to be determined, ϕ is an application from $\mathbb{R}_+$ to $\mathbb{R}$ and $\|.\|$ is the standard 2-norm.

For the choice of the trust-region model m_k which is twice continuously differentiable, it is required that a function $\phi\left(\|x - y_i\|\right)$ as smooth as possible and the degree of the polynomial added to the radial terms does not exceed one. So, $q(x) = c_0 + c^t x$ is taken in the model.

The radial function of type r^β is the only without parameters, except exponent, which belongs to twice continuously differantiable functions space in $\mathbb{R}^n$. Note that cubic splines, corresponding to the choice of $\beta = 3$ in dimension 1, are the smoothest interpolation functions. Consequently, in [33], the following trust region models is chosen

$$m_k(x) = \sum_{i=1}^{p} \lambda_i \|x - x_i\|^3 + c_0 + c^t x.$$

More detailed parts and application on BOOSTERS can be found [33], [35].

2.5.6 ORBIT (Optimization by Radial Basis function Interpolation in Trust regions)

This method is a trust-region framework based on Radial Basis Functions (RBFs) [62]. The problem is an unconstrained local minimization $\min_{x \in \mathbb{R}^n} \quad f(x)$.

ORBIT algorithms is based on forming a model which is computationally simple to evaluate and possesses well-behaved derivatives. The model is optimized over compact regions to generate new points which can be evaluated by the computationally expensive function. Using this new function value to update the model, an iterative process works.

As a different definition of trust-region $\mathcal{B}$, the trust-region norm (at iteration k), $\|.\|_k$, is distinguished from the standard 2-norm $\|.\|$ and other norms in the sequel (e.g. ∞-norm). It is assumed that for some constant c_k such that $\|.\|_2 \leq c_k \|.\|_k$.

A nonlinear interpolation model is formed using fewer than a quadratic (in the dimension) number of points. A so-called 'fully linear tail' is employed to guarantee that the model

approximates both the function and its gradient reasonably well. The interpolation model is taken in the form

$$m_k(x_k + s) = \sum_{j=1}^{|\mathcal{Y}|} \lambda_j \phi(\left\| s - y_j \right\|) + p(s),$$

where $\phi : \mathbb{R}_+ \to \mathbb{R}$ is a univariate function, $p \in \mathcal{P}^n_{d-1}$, $|\mathcal{Y}|$ is the cardinality of the interpolation set $\mathcal{Y}$ and $\|.\|$ is Euclidean norm.

The algorithm of ORBIT, which can be found in [62] detailed, works with a set of displacements, $\mathcal{D}_k$, from the current center x_k. About this set, $d_i \in \mathcal{D}_k \leftrightarrow f(x_k + d_i)$ is known. Since evaluation of f is computationally expensive, the importance of having complete memory of all points previously evaluated by the algorithm is emphasized. This is an important difference between ORBIT and previous algorithms BOOSTERS, UOBYQA and NEWUOA, where, in order to reduce linear algebraic costs, the interpolation set is changed by at most one point and hence very limited memory is required.

2.5.7 Minimal Norm Hessians (MNH)

This method has two new features [61]:

- Unlike previous algorithms, which were driven by a desire to keep linear algebraic overhead to $O(n^3)$ operations per iteration, MNH views overhead as negligible relative to the expense of function evaluation. This allows greater flexibility in using points from the *bank*. It is viewed tha data gained from each function evaluation contributes to a bank of insight into the function.

- MNH models are formed from interpolation sets in a computationally stable manner which guarantees that the models are well-behaved. In other words, the model is convergent to first order critical point without any difficulty.

Quadratic model is taken as

$$m_k(x_k + s) = f(x_k) + g_k^T + \frac{1}{2} s^T H_k s \,,$$

where $g_k = \nabla f(x_k)$ and $H_k = \nabla^2 f(x_k)$.

Given an initial point x_0 and a maximum radius $\Delta_{\max}$, f is sampled within the relaxed set which is defined as

$$\mathcal{L} = \{y \in \mathbb{R}^n \ : \ \|x - y\|_2 \leq \Delta_{\max} \quad \text{for some } x \text{ with } f(x) \leq f(x_0)\}$$

With the interpolation set $\mathcal{Y} = \left\{y_1 = 0, y_2, \ldots, y_{|\mathcal{Y}|}\right\} \subset \mathbb{R}^n$ interpolation condition

$$m_k(x_k + y_j) = f(x_k + y_j) \qquad \text{for all } y_j \in \mathcal{Y} \ .$$

Now, it is defined

$$\mu(x) \overset{def}{=} [1, \chi_1, \ldots, \chi_n],$$

$$\nu(x) \overset{def}{=} \left[\frac{\chi_1^2}{2}, \ldots, \frac{\chi_n^2}{2}, \frac{\chi_1\chi_2}{\sqrt{2}}, \ldots, \frac{\chi_{n-1}\chi_n}{\sqrt{2}}\right],$$

where χ_i denotes the ith component of $x \in \mathbb{R}^n$.

Then $[\mu(x), \nu(x)]$ is taken as form of basis for the linear space of quadratics in n variables, $\mathcal{Q}^n$. So, any quadratic $m_k \in \mathcal{Q}^n$ can be written as

$$m_k(x - x_k) = \alpha^T \mu(x - x_k) + \beta^T \nu(x - x_k)$$

for coefficients $\alpha \in \mathbb{R}^{n+1}$ and $\beta \in \mathbb{R}^{\frac{n(n+1)}{2}}$. By the way, f is denoted as

$$\begin{bmatrix} M_{\mathcal{Y}} \\ N_{\mathcal{Y}} \end{bmatrix}^T \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = f, \tag{2.5}$$

where $M_{\mathcal{Y}} \in \mathbb{R}^{n+1\times|\mathcal{Y}|}$ and $N_{\mathcal{Y}} \in \mathbb{R}^{n(n+1)/2\times|\mathcal{Y}|}$ are defined by $M_{i,j} = \mu_i(y_j)$ and $N_{i,j} = \nu_i(y_j)$, respectively. Note that these matrices depend on the interpolation set $\mathcal{Y}$.

In MNH, it is focused on solutions to the interpolation problem (2.5) that are of *minimum norm* with respect to the vector β. Hence the required solution (α, β) satisfies

$$\min\left\{\frac{1}{2}\|\beta\|^2 \ : \ M_{\mathcal{Y}}^T\alpha + N_{\mathcal{Y}}^T\beta = f\right\}. \tag{2.6}$$

The solution to this system is interested in because it represents the quadratic whose Hessian matrix is of *minimum* Frobenius norm [13] since $\|\beta\| = \left\|\nabla^2_{xx} m(x)\right\|_F$. If other *minimal norm* quadratics is found, they may be drawn to those with Hessian of minimal norm because the resulting solution procedure will be relative to fully linear models.

The solution way of (2.6) and the detailed parts is in [61].

2.5.8 Retrospective Trust-Region Method

A new trust-region method for unconstrained optimization where the radius update is computed using the model information at the current iterate rather than at the preceding one is introduced. The update is then performed according to how well the current model retrospectively predicts the value of the objective function at last iterate [55].

The mail goal is to propose a trust-region algorithm for unconstrained optimization problem that determines Δ_{k+1} according to how well m_{k+1} predicts the value of the objective function at x_k and in that connection the radius is updated with the change in models [55].

The difference this new algorithm from the basic trust-region algorithm comes in the ratio of reductions step. Namely, the trust-region radius is updated after each sucsessful iteration k (i.e, at the beginning of iteration $k+1$) on the *retrospective ratio*

$$\tilde{\rho}_{k+1} \overset{def}{=} \frac{f(x_{k+1}) - f(x_{k+1} - s_k)}{m_{k+1}(x_{k+1}) - m_{k+1}(x_{k+1} - s_k)} = \frac{f(x_k) - f(x_k + s_k)}{m_{k+1}(x_k) - m_{k+1}(x_k + s_k)}$$

ρ_k is continued to be used to decide whether the trial iterate may be accepted. In other words, deciding acceptance of the trial iterate and determining the radius update are two remarkable parts of the new algorithm.

Algorithm : Retrospective trust-region algorithm (RTR) [55]

Step 0: Initialization. An initial point x_0 and initial trust-region radius $\Delta_0 > 0$ are given. The constants $\eta_1, \tilde{\eta}_1, \tilde{\eta}_2, \gamma_1$ and γ_2 are given and satisfy $0 < \eta_1 < 1$, $0 < \tilde{\eta}_1 \leq \tilde{\eta}_2 < 1$, and $0 < \gamma_1 \leq \gamma_2 < 1$. Compute $f(x_0)$ and set $k = 0$.

Step 1: Model definition. Select a twice-continuously differentiable m_k defined in $\mathcal{B}_k$.

Step 2: Retrospective trust-region radius update. If $k = 0$, go to Step 3.
If $x_k = x_{k-1}$, choose Δ_k in $[\gamma_1\Delta_{k-1}, \gamma_2\Delta_{k-1})$. Else, define

$$\tilde{\rho}_k = \frac{f(x_{k-1}) - f(x_k)}{m_k(x_{k-1}) - m_k(x_k)}$$

and choose

$$\Delta_k \in \begin{cases} [\Delta_{k-1}, \infty) & \text{if } \tilde{\rho}_k \geq \tilde{\eta}_2 \\ [\gamma_2\Delta_{k-1}, \Delta_{k-1}) & \text{if } \tilde{\rho}_k \in [\tilde{\eta}_1, \tilde{\eta}_2) \\ [\gamma_1\Delta_{k-1}, \gamma_2\Delta_{k-1}) & \text{if } \tilde{\rho}_k < \tilde{\eta}_1 \ . \end{cases}$$

Step 3: Step Calculation. Compute a step s_k that "sufficiently reduces the model" m_k and such that $x_k + s_k \in \mathcal{B}_k$.

Step 4: Acceptance of the trial point. Compute $f(x_k + s_k)$ and define

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

If $\rho_k \geq \eta_1$, define $x_{k+1} = x_k + s_k$ otherwise, define $x_{k+1} = x_k$.
Increment k by one ($k := k + 1$) and go to Step 1.

2.5.9 Wedge Methods

Owing to Wedge methods, it is possible to find a point which improves the value of the model while maintaining an acceptable level of poisedness. This algorithm follows the approach of attempting to generate points which simultaneously provide sufficient decrease for the model and satisfy the Λ-poisedness condition. At every iteration, the trust-region subproblem minimization is augmented by an additional constraint which does not allow to lie near a certain manifold which has the shape of 'wedge' [13].

At the current iterate x_c, the model is defined as

$$m_c(x_c + s) = f(x_c) + g_c^T s + \frac{1}{2} s^T G_c s$$

where the vector $g_c \in \mathbb{R}^n$ and the $n \times n$ symmetric matrix G_c must be determined so that the model interpolates f at a set of sample points ($G_c = 0$ for linear models) [26].

To define the model m_c uniquely, in addition to x_c, a set of m points $\sum_c = \left\{y^1, \ldots, y^m\right\}$, which are called *satellites of* x_c, are maintained. For a linear model $m = n$, for a quadratic model $m = (n + 1)(n + 2)/2 - 1$ should be chosen. Then the interpolation conditions

$$m_c(x_c) = f(x_c), \qquad m(y^l) = f(y^l), \quad l = 1, \ldots, m,$$

are imposed. When the model m_c is uniquely determined by these conditions, the interpolation set $\{x_c\} \cup \sum_c$ is called the *non-degenerate*.

Suppose that the current iteration starts with a non-degenerate set of sample points $\{x_c\} \cup \sum_c$. Identify the farthest satellite from the current iterate x_c, say $y^{l_{out}}$, as the point that will be removed from $\sum_c$. This choice promotes the conservation of points that provide local

information of f around x_c. Then define a 'region' in $\mathbb{R}^n$ that contains all the points $x_c + s$ that, if included in the interpolation set in place of $y^{l_{out}}$, would result in a degenerate set of sample points. A set $\mathcal{W}_c$ that contains $\mathcal{T}_c$ is also defined. $\mathcal{W}_c$ is designed to avoid points that are very near $\mathcal{T}_c$ [26].

After the "wedge" $\mathcal{W}_c$ is determined, a trial step s_c is computed by solving the subproblem

$$\begin{aligned} \min_s \quad m_c(x_c + s) &= f(x_c) + g_c^T s + \tfrac{1}{2} s^T G_c s \\ \text{subject to } \|s\| &\leq \Delta_c \,, \\ s &\notin \mathcal{W}_c \,, \end{aligned}$$

and define $x_+ := x_c + s_c$. If x_+ does not reduce f, the current iterate is not updated, and x_+ may or may not be discarded, depending on how far it is from the current iterate x_c compared with $y^{l_{out}}$. The second constraint is called a 'wedge constraint'. For a linear model $m_k(x_k + s) = f(x_k) + g_k^T s$ the wedge constraint can be illustrated as $\left\|b_k^T s\right\| \geq \gamma \|b_k\| \|s\|$ where γ is the parameter determining the 'width' of the wedge and $b_k \in \mathbb{R}^n$ is a vector [26].

2.6 Benchmarking of Derivative Free Optimization Methods

The benchmarking procedures for derivative-free optimization algorithms are explored when there is a limited computational budget [30]. These on selected applications with trajectory plots provide useful information to users. Particularly, users can find the solver that delivers the largest reduction within a given computational budget.

To evaluate the performance of derivative-free solvers, *performance profiles* with the convergence test is used. Help of these performance profiles to users is to choose a solver that provides a given reduction in function value within a limit of μ_f function evaluations.

Performance Profiles are defined in terms of *performance measure* $t_{p,s} > 0$ obtained for each $p \in \mathcal{P}$ (a set of benchmark problems) and $s \in \mathcal{S}$ (a set of optimization solvers). The *performance ratio* is defined by

$$r_{p,s} \overset{def}{=} \frac{t_{p,s}}{\min\left\{t_{p,s} : 1 \leq s \leq |\mathcal{S}|\right\}} \,,$$

where $|\mathcal{S}|$ denotes the cardinality of $\mathcal{S}$.

The best solver for a particular problem reaches the lower bound $r_{p,s} = 1$. If $r_{p,s} = \infty$, solver s fails to converge on problem p.

The performance profile of a solver $s \in \mathcal{S}$ can be defined as the fraction of problems where the performance ratio is at most α, namely,

$$\rho_s(\alpha) = \frac{1}{|\mathcal{P}|}\text{size}\left\{p \in \mathcal{P} : r_{p,s} \leq \alpha\right\}; \tag{2.7}$$

(2.7) is a probability distribution for the ratio $r_{p,s}$.
Note that $\rho_s(1)$ is the fraction of problems for which solver $s \in \mathcal{S}$ is the fastest and that for α sufficiently large, $\rho_s(\alpha)$ is the fraction of problems solved by $s \in \mathcal{S}$. Solvers with high values for $\rho_s(\alpha)$ are preferable.

Convergence test for benchmarking derivative-free solvers that does not depend on the gradient can be written

$$f(x_0) - f(x) \geq (1 - \tau)(f(x_0) - f_L), \tag{2.8}$$

where $\tau > 0$: tolerance,
x_0 : the starting point for the problem,
$f : \mathbb{R}^n \to \mathbb{R}$: the unconstrained problem's objective function which may be noisy or non-differentiable and evaluation of f is computationally expensive.
f_L : computed for each problem $p \in \mathcal{P}$ as the smallest value of f obtained by any solver within a given number μ_f of function evaluations.

Users with expensive optimization problems are interested in the percentage of problems that can be solved (for a given tolerance τ) with α function evaluations. This information can be obtained by *data profile* in which $t_{p,s}$ is the number of function evaluations required to satisfy (2.8) for a given tolerance τ and n_p is the number of variables in $p \in \mathcal{P}$ such that it is defined as

$$d_s(\alpha) = \frac{1}{|\mathcal{P}|}\text{size}\left\{p \in \mathcal{P} : \frac{t_{p,s}}{n_p + 1} \leq \alpha\right\} .$$

There is a limit μ_f on the total number of function evaluations, and if the convergence test (2.8) is failed after μ_f evaluations, $t_{p,s} = \infty$. The unit of cost which is $n_p + 1$ function evaluations can be easily translated into function evaluations. This unit of cost has another advantage that $d_s(\alpha)$ can then be interpreted as the percentage of problems that can be solved with the equivalent of α (simplex) gradient estimates.

Performance profiles and data profiles are probability density functions, and so monotone increasing, step functions with a range in $[0, 1]$. However, performance profiles compare different solvers, while data profiles display the raw data. Particularly, performance profiles

do not provide the number of function evaluations required to solve any of the problems. Also note that the data profile for a given solver $s \in \mathcal{S}$ is independent of other solvers; this is not the case for performance profiles.

As another connection between performance and data profiles, the limiting value of $\rho_s(\tau)$ as $\tau \to \infty$ is the percentage of problems that can be solved with μ_f function evaluations. Thus [30],

$$d_s(\mu_f) = \lim_{\tau \to \infty} \rho_s(\tau) \, .$$

This shows that the data profile d_s measures the reliability of the solver (for a given tolerance τ) as a function of the budget μ_f since the limiting value of ρ_s can be interpreted as the reliability of the solver.

Application to the chosen derivative-free solvers and obtained results can be found in [30].

2.7 Self-Correcting Geometry in the Model-Based Algorithms

In [58], it is shown that to ignore geometry considerations altogether if one wishes to maintain global convergence is impossible. Besides, an algorithm that resorts to the geometry-improving steps as little as possible (while still maintaining a mechanism for taking geometry into account) can be provided . As it was shown in the study [58], the design and convergence proof of this new algorithm crucially depends on a self-correction mechanism resulting from the combination of the trust-region mechanism with the polynomial interpolation setting. This mechanism also throws some light on the good numerical results reported by the study [27] for a method where no care is ever taken to guarantee poisedness of the interpolation set.

The main idea of the new method is to rely on new points generated by the algorithm to maintain poisedness of the interpolation set. Some special care is taken for the final criticality test and also the geometry is observed using Lagrange polynomials.

A remarkable point of the algorithm in [58] is that the trust-region radius is not reduced when an interpolation point is exchanged with unsuccessful trial point.

CHAPTER 3

MULTILEVEL OPTIMIZATION

3.1 Multigrid Optimization

In 2005, Lewis and Nash studied some model problems for the multigrid optimization of systems governed by differential equations [31]. The goal is to demonstrate not a particular optimization-based multigrid algorithm but an optimization-based multigrid algorithm which is distinct from and superior to an equation-based multigrid approach. Moreover, how multigrid can be used to accelerate nonlinear programming algorithms is shown.

Consider the nonlinear optimization problem

$$\underset{a}{\text{minimize}} \quad F(a) = f(a, u(a)) \,, \tag{3.1}$$

where a is a set of design variables, and $u = u(a)$ is a set of state variables. Given a, the state variables are defined implicitly by a system of state equations

$$S(a, u(a)) = 0 \tag{3.2}$$

in a and u. The equation $S(a, u) = 0$ is a system of PDEs.

An optimization-based multigrid algorithm is investigated to solve these two equations.

To use computations on a coarse discretization of (3.1)-(3.2) to improve an approximate solution of a finer-resolution problem, a multigrid algorithm, called *MG/Opt*, is proposed. MG/Opt. recursively uses coarse resolution problems to generate search directions for finer-resolution problems. Then a *line search* is used to refine the solution of each finer-resolution problem. A line search globalization technique makes it possible to prove convergence results (i.e. convergence to stationary points from arbitrary strating points) for the overall multigrid

optimization algorithm. MG/Opt. is viewed as a nonlinear adaptation of the multigrid idea. In particular, the multigrid subproblems are nonlinear optimization problems, not systems of linear or nonlinear equations.

MG/Opt. is inspired by multigrid methods for elliptic PDEs. These algorithms for elliptics are highly efficient.

Grids are taken as uniform grids which are a set of dicretizations. In the analysis of the model problems, the design variable a correspond to the discretization of a function.

Transfer operators between grids :
I_H^h : prolongation operator (coarse $\mapsto$ fine),
I_h^H : restriction operator (fine $\mapsto$ coarse).
The standard assumption as in multigrid contexts [6] is

$$I_H^h = [\text{constant}] \times \left(I_h^H\right) T.$$

MG/Opt. Algorithm [31]

a^0 : a step that the algorithm begins,
a^1 : an initial estimate of the solution on the finest grid, via :

- If on the coarsest grid,

  $$\text{minimize } F_h(a_h) = f_h(a_h, u_h(a_h))$$

  with the initial estimate a_h^0, to obtain a_h^1.

- Otherwise,

  - Partially minimize (i.e., at least one iteration of a globally convergent nonlinear optimization algorithm is applied to the optimization model) $F_h(a_h)$, with initial estimate a_h^0, to get $a_{h,1}$.
    Downdate the result to obtain $a_{H,1} = I_h^H a_{h,1}$.
  - Compute $v_H = \nabla F_H(a_{H,1}) - I_h^H \nabla F_h(a_{h,1})$.
  - Recursively apply MG/Opt. (with initial estimate $a_{H,1}$) to solve

    $$\underset{a_H}{\text{minimize}}\ F_H(a_H) - v_H^T a_H$$

subject to bound contraints

$$a_{H,low} \leq a_H \leq a_{H,up}$$

to obtain $a_{H,2}$. (Bounds are added to improve performance.)

- Compute the search direction $e_h = I_H^h (a_{H,2} - a_{H,1})$. ($e_h$ will be a descent direction, ensuring that the estimate of the solution improves at every iteration of the multigrid algorithm.)
- Use a line search to get $a_{h,2} = a_{h,1} + \alpha e_h$.
- Partially minimize $F_h(a)$, with initial estimate $a_{h,2}$, to obtain a_h^1.

Alternatives to MG/Opt

(3.1)-(3.2) can be reduced to the Karush-Kuhn Tucker (KKT) conditions as the equivalent system

$$\begin{aligned} \underset{a,u}{\text{minimize}} \quad & f(a,u) \\ \text{subject to} \quad & S(a,u) = 0 \end{aligned}$$

and then applies the multigrid to the resulting system of equations. It is not difficult to extend an approach based on KKT conditions to problems that include inequality constraints. Thus, it lacks the guarantee of convergence, and the range of applicability of MG/Opt.

Consider design variables a are fixed and only u changes with the level of discretization. Reduced Hessians based on coarser grid calculations are used as preconditioners in a sequential quadratic programming (SQP) approach. This approach is effective in accelerating an SQP algorithm for (3.1)-(3.2). Nevertheless, the algorithm makes one pass from the coarsest level to finest, using the solution of optimization problem from the previous level of discretization as a starting point for the current one which is one of the *successive refinement*. The approach lacks the multigrid feature of passing back and forth between the various levels of discretization.

Detailed observations and numerical results of model problems can be found in [31].

3.2 Recursive Multilevel Trust Region Method

A class of trust-region methods is presented for solving unconstrained nonlinear and possibly nonconvex discretized optimization problems, like those arising in systems by partial differential equations. The algorithms in the recursive class use the discretization of level to speed up the computation of the step. This leads to true multilevel optimization methods reminiscent of multigrid methods. The first study on this subject is what we will now consider [22].

General idea is to use the different levels of discretization for an infinite dimensional problem. The main idea is to use coarser grids to compute approximate solutions and then use these starting point for the optimization problem on a finer grid.

Consider the solution of the unconstrained problem

$$\min_{x\in\mathbb{R}^n} \quad f(x), \tag{3.3}$$

where f is a twice continuously differentiable objective function and maps from $\mathbb{R}^n$ to $\mathbb{R}$. Here, [22], trust region methods are iterative and produce a sequence x_k of iterates converging to a local stationary point (i.e. to a point where $g(x) = \nabla_x f(x) = 0$) for the problem with an initial point x_0. After this, the logic of basic trust region algorithm works.

To obtain sufficient decrease, the following trust-region subproblem with the quadratic model m_k is solved :

$$\min_{\|s\|\leq\Delta_k} m_k(x_k + s) = \min_{\|s\|\leq\Delta_k} f(x_k) + \langle g_k, s\rangle + \frac{1}{2}\langle s, H_k s\rangle \ ,$$

where $g_k \overset{def}{=} \nabla f(x_k)$ and H_k is a symmetric $n \times n$ approximation of $\nabla^2 f(x_k)$,
$\langle ., .\rangle$ is Euclidean inner product, $\|.\|$ is Euclidean norm.

Such methods converge to first-order critical points whenever $\{\|H_k\|\}$ is uniformly bounded. Moreover, computational cost per iteration of evaluation of $f(x_k + s_k)$ is caused by the numerical solution of the subproblem, depending on the size n of the problem. In order to reduce the cost per iteration, different levels of dicretization is used.

Now, assume that there is a collection of functions $\{f_i\}_{i=0}^r$ such that each f_i is a twice continuously differentiable function and maps from $\mathbb{R}^{n_i}$ to $\mathbb{R}$ (with $n_i \leq n_{i-1}$). And assume that for each $i = 1, \ldots, r$, f_i is 'more costly' to minimize than f_{i-1} where $f_r(x) = f(x)$.

The relation between f_i and f_{i-1} is as in multigrid algorithms by operators full-rank linear

operator $R_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i-1}$ (the restriction) and $P_i : \mathbb{R}^{n_i-1} \to \mathbb{R}^{n_i}$ (the prolongation) such that

$$P_i = \sigma_i R_i^T \ .$$

The idea is to use f_{r-1} to construct an alternative model h_{r-1} for $f_r = f$ in the neighbourhood of the current iterate that is cheaper than the quadratic model at level r and to use this alternative model, whenever suitable, to define the step in the trust-region algorithm. If $r > 1$, this can be done recursively. The approximation process stops at level 0, where the quadratic model is always used (Taylor step).

To restrict $x_{i,k}$ to create the starting iterate $x_{i-1,0}$ at level $i-1$, that is

$$x_{i-1,0} = R_i x_{i,k} \ .$$

Then the lower level model is defined as :

$$h_{i-1}(x_{i-1,0} + s_{i-1}) \overset{def}{=} f_{i-1}(x_{i-1,0} + s_{i-1}) + \langle v_{i-1}, s_{i-1} \rangle \ ,$$

where $v_{i-1} = R_i g_{i,k} - \nabla_{x_{i-1}} f_{i-1}(x_{i-1,0})$ with $g_{i,k} \overset{def}{=} \nabla_{x_i} h_i(x_{i,k})$. By convention, setting $v_r = 0$ such that, for all s_r,

$$h_r(x_{r,0} + s_r) = f(x_{r,0} + s_r) = f(x_0 + s) \text{ and } g_{r,k} = \nabla_{x_r} h_r(x_{r,k}) = \nabla_x f(x_k) = g_k \ .$$

The function h_i corresponds to a first order modification of f_i by a linear term and enforces the relation

$$g_{i-1,0} = \nabla_{x_{i-1}} h_{i-1}(x_{i-1,0}) = R_i g_{i,k} \ .$$

First-order behaviors of h_i and h_{i-1} are in harmony in a neighbourhood of $x_{i,k}$ and $x_{i-1,0}$. If $s_i = P_i s_{i-1}$:

$$\langle g_{i,k}, s_i \rangle = \langle g_{i,k}, P_i s_{i-1} \rangle = \frac{1}{\sigma_i} \langle R_i g_{i,k}, s_{i-1} \rangle = \frac{1}{\sigma_i} \langle g_{i-1,0}, s_{i-1} \rangle \ .$$

When entering level $i = 0, \ldots, r$, h_i is locally minimized starting from $x_{i,0}$. At iteration k of this minimization, at iterate $x_{i,k}$, either the model $h_{i-1}(x_{i-1,0} + s_{i-1})$ or Taylor approximation model

$$m_{i,k}(x_{i,k} + s_i) = h_i(x_{i,k}) + \langle g_{i,k}, s_i \rangle + \frac{1}{2} \langle s_i, H_{i,k} s_i \rangle \ . \tag{3.4}$$

is chosen. Here, $H_{i,k}$ is a symmetric $n_i \times n_i$ approximation to the second derivatives of h_i at $x_{i,k}$.

After the model is chosen, a step $s_{i,k}$ is computed which generates a decrease within a trust-region $\{s_i \ : \ \|s_i\|_i \leq \Delta_{i,k}\}$. The norm $\|.\|_i$ is level-dependent and for some symmetric positive-definite matrix M_i, it is defined by

$$\|s_i\|_i \overset{def}{=} \sqrt{\langle s_i, M_i s_i \rangle} \overset{def}{=} \|s_i\|_{M_i} \ .$$

If the model (3.4) is chosen, the step $s_{i,k}$ must satisfy the *sufficient decrease* or *Cauchy point* condition with $\kappa_{red} \in (0,1)$:

$$m_{i,k}(x_{i,k}) - m_{i,k}(x_{i,k} + s_{i,k}) \geq \kappa_{red} \left\|g_{i,k}\right\| \min\left[\frac{\left\|g_{i,k}\right\|}{1 + \left\|H_{i,k}\right\|}, \Delta_{i,k}\right] . \tag{3.5}$$

If the model h_{i-1} is chosen, it should produce a new minimum point $x_{i-1,*}$ such that

$$h_{i-1}(x_{i-1,*}) < h_{i-1}(x_{i-1,0}) \ ,$$

and a corresponding step $x_{i-1,*} - x_{i-1,0}$ which must then be brought back to level i by the prolongation P_i.

Since

$$\|s_i\|_i = \|s_i\|_{M_i} = \|P_i s_{i-1}\|_{M_i} = \|s_{i-1}\|_{P_i^T M_i P_i} \overset{def}{=} \|s_{i-1}\|_{M_{i-1}} = \|s_{i-1}\|_{i-1} \tag{3.6}$$

(which is well-defined since P_i is full-rank), the trust-region constraint at level $i-1$ then becomes

$$\left\|x_{i-1,*} - x_{i-1,0}\right\|_{i-1} \leq \Delta_{i,k}.$$

The lower level subproblem consists in (possibly approximately) solving

$$\min_{\|s_{i-1}\|_{i-1} \leq \Delta_{i,k}} h_{i-1}(x_{i-1,0} + s_{i-1}). \tag{3.7}$$

The model h_{i-1} is not always useful, for example if $R_i g_{i,k}$ is zero while $g_{i,k}$ is not. Therefore $\left\|g_{i-1,0}\right\| = \left\|R_i g_{i,k}\right\|$ should be large enough compared to $\left\|g_{i,k}\right\|$. The model h_{i-1} is restricted to iterations with

$$\left\|R_i g_{i,k}\right\| \geq \kappa_g \left\|g_{i,k}\right\| \quad \text{and} \quad \left\|R_i g_{i,k}\right\| > \epsilon_{i-1}^g \tag{3.8}$$

where $\kappa_g \in (0, \min[1, \min_i \|R_i\|])$ and $\epsilon_{i-1}^g \in (0,1)$ which is a measure of the first-order criticality for h_{i-1} that is judged sufficient at level $i-1$.

Description of parameters used in the following algorithm

$\Delta_i^s > 0$	:	initial trust-region radius at level i
Δ_{i+1}	:	trust-region radius at level $i+1$
$\epsilon_i^g \in (0,1)$	:	level-dependent gradient norm tolerances
$\epsilon_i^\Delta \in (0,1)$	:	trust-region tolerances for each level i
η_1, η_2	:	parameters to accept the trial point
γ_1	:	trust-region radius decreasing factor
γ_2	:	trust-region radius increasing factor
$\kappa_g \in (0,1)$	:	constant for the model choice.

Algorithm: $RMTR(i, x_{i,0}, g_{i,0}, \Delta_{i+1}, \epsilon_i^g, \epsilon_i^\Delta, \Delta_{i,0})$ [22]

Step 0: Initialization.

Compute $v_i = g_{i,0} - \nabla f_i(x_{i,0})$ and $h_i(x_{i,0})$. Set $\Delta_{i,0} = \min\left[\Delta_i^s, \Delta_{i+1}\right]$ and $k = 0$.

Step 1: Model Choice.

If $i = 0$ or if (3.8) fails, go to step 3. Otherwise, choose to go to Step 2 (recursive step) or to Step 3 (Taylor step).

Step 2: Recursive step computation.

Call algorithm $RMTR(i-1, R_i x_{i,k}, R_i g_{i,k}, \Delta_{i,k}, \epsilon_{i-1}^g, \epsilon_{i-1}^\Delta, \Delta_{i-1}^s)$, yielding an approximate solution $x_{i-1,*}$ of (3.7). Then define $s_{i,k} = P_i(x_{i-1,*} - R_i x_{i,k})$, set $\delta_{i,k} = h_{i-1}(R_i x_{i,k}) - h_{i-1}(x_{i-1,*})$ and go to Step 4.

Step 3: Taylor step computation.

Choose $H_{i,k}$ and compute a step $s_{i,k} \in \mathbb{R}^{n_i}$ that sufficiently reduces the model $m_{i,k}$ (3.4) in the sense of (3.5) and such that $\left\|s_{i,k}\right\|_i \leq \Delta_{i,k}$. Set $\delta_{i,k} = m_{i,k}(x_{i,k}) - m_{i,k}(x_{i,k} + s_{i,k})$.

Step 4: Acceptance of the trial point.

Compute $h_i(x_{i,k} + s_{i,k})$ and define

$$\rho_{i,k} \stackrel{def}{=} \frac{h_i(x_{i,k}) - h_i(x_{i,k} + s_{i,k})}{\delta_{i,k}}.$$

If $\rho_{i,k} \geq \eta_1$, then define $x_{i,k+1} = x_{i,k} + s_{i,k}$; otherwise define $x_{i,k+1} = x_{i,k}$.

Step 5: Termination.

Compute $g_{i,k+1}$. If $\left\|g_{i,k+1}\right\|_\infty \leq \epsilon_i^g$ or $\left\|x_{i,k+1} - x_{i,0}\right\|_i > (1 - \epsilon_i^\Delta)\Delta_{i+1}$, then return with the approximate solution $x_{i,*} = x_{i,k+1}$.

Step 6: Trust-region radius update.

Set

$$\Delta_{i,k}^{+} \in \begin{cases} [\Delta_{i,k}, +\infty) \ , & \text{if } \rho_{i,k} \geq \eta_2 \\ [\gamma_2 \Delta_{i,k}, \Delta_{i,k}] \ , & \text{if } \rho_{i,k} \in [\eta_1, \eta_2] \\ [\gamma_1 \Delta_{i,k}, \gamma_2 \Delta_{i,k}] \ , & \text{if } \rho_{i,k} < \eta_1 \end{cases}$$

and

$$\Delta_{i,k+1} = min \left[\Delta_{i,k}^{+}, \ \Delta_{i+1} - \left\| x_{i,k+1} - x_{i,0} \right\|_i \right] \ .$$

Increment k by one and go to Step 1.

Practical Issues of the algorithm

The efficient algorithms can be found in the theoretical shell of RMTR. For example, multigrid smoothers are very efficient in reducing the high frequency components.

At the coarsest level, where further recursion is impossible, the cost of exactly minimizing (3.4) within the trust region remains small, because of the low dimensionality of the subproblem. So, RMTR's strategy is to solve it using the method by Moré and Sorensen [28] whose very acceptable cost is then dominated by that of a small number of small-scale Cholesky factorizations.

At finer levels, the choice of using the truncated conjugate-gradient TCG [51] or generalized Lanczos trust-region GLTR algorithms [23] or an adaptation of the multigrid smoothing techniques guarantees sufficient descent inside trust region and also handles the possible nonconvexity of the model.

One of the flexible feature of RMTR is that the minimization at lower levels ($i = 1, \ldots, r-1$) can be stopped after the first successful iteration without affecting convergence properties. Therefore this opens the possibility of considering *fixed form* and *free form* recursion patterns. The fixed form recursion patterns are obtained by specifying a maximum number of successful iterations at each level, a technique directly inspired from the definitions of *V and W cycles* in multigrid algorithms.

3.2.1 The Multilevel Moré Sorensen Algorithm

This is an alternative application of the trust-region paradigm to the recursive class of problems. A multilevel numerical algorithm is presented for the exact solution of the Euclidean trust-region subproblem. Typically this subproblem is constructed when optimizating a nonlinear (possibly non-convex) objective function whose variables are discretized continuous functions. In this case, the different levels of discretization provide a natural multilevel context [56].

When multilevel techniques are considered for the exact solution of the trust-region subproblem at the highest level, that is for the finest discretization, if the objective function is convex (locally), then a suitable optimizing step is derived from the solution of (a variant of) Newton's equations. This often results in solving a positive-definite linear system. For instance, if the local Hessian is given by a discretized Laplacian or another elliptic operator (case of positive-definite system), then applying a classical multigrid solver to this system yields a very efficient method to compute the step.

In the following discussions, all norms are Euclidean norm if there is no subscript anywhere of the norm.

As before, the problem is an unconstrained optimization problem $\min_{x\in\mathbb{R}^n} f(x)$, where f is a twice continuously differentiable function mapping from $\mathbb{R}^n$ to $\mathbb{R}$ and is bounded below. The standard trust-region subproblem in the region $\mathcal{B} \stackrel{def}{=} \{x \in \mathbb{R}^n \ : \ \|x - x_k\| \leq \Delta_k\}$

$$\min_{\|s\|\leq\Delta} m_k(x_k + s) \stackrel{def}{=} \langle g_k, s\rangle + \frac{1}{2}\langle s, H_k s\rangle \ , \tag{3.9}$$

where $g_k = \nabla_x f(x_k)$ and H_k is a bounded symmetric approximation of $\nabla_{xx} f(x_k)$. The goal is to find the exact solution of this equation.

Any global minimizer s^M of (3.9) satisfies the system of linear equations [54]

$$H(\lambda^M)s^M = -g \ , \tag{3.10}$$

where $H(\lambda^M) \stackrel{def}{=} H + \lambda^M I$ is a positive semidefinite, $\lambda^M \geq 0$ and $\lambda^M(\left\|s^M\right\| - \Delta) = 0$, with s^M which is unique if $H(\lambda^M)$ is positive definite.

The Hessian curvature is induced by the additional term $\lambda^M I$. Thus, it must make s^M lie within the trust-region $\{s \in \mathbb{R}^n \ : \ \|s\| \leq \Delta\}$.

If $\lambda > -\lambda_{\min}(H)$, the smallest eigenvalue of H, then $H(\lambda)$ is positive definite and the system (3.10) has a unique solution

$$s(\lambda) = -H(\lambda)^{-1} g \,, \tag{3.11}$$

satisfying the nonlinear inequality (whenever $\lambda = 0$)

$$\|s(\lambda)\| \leq \Delta \,, \tag{3.12}$$

or the nonlinear inequality (if $\lambda > 0$)

$$\|s(\lambda)\| = \Delta \,. \tag{3.13}$$

The Moré-Sorensen method consists in finding the minimizer $s(\lambda^M)$ by solving either (3.10) if (3.12) holds for $\lambda = 0$, or (3.10) and (3.13) together otherwise. To perform the latter system, *secular equation* is used

$$\phi(\lambda) \stackrel{def}{=} \frac{1}{\|s(\lambda)\|} - \frac{1}{\Delta} = 0 \,. \tag{3.14}$$

The solution of this secular equation is found by root finding method such as Newton's method. If $H(\lambda)$ is positive definite, then the Newton step from λ [54]

$$\lambda^{new} = \lambda + \left(\frac{\|s\| - \Delta}{\Delta}\right)\left(\frac{\|s\|^2}{\|w\|^2}\right) \,, \tag{3.15}$$

where w solves $Lw = s$ with L the Lower Cholesky factor of $H(\lambda)$.

If $H(\lambda)$ is not positive definite, then λ is increased until this is the case.

In order to guarantee convergence from arbitrary starting values, the λ iterates are kept in an interval $[\lambda^L, \lambda^U]$.

The following algorithm is given for fixed values $\Delta > 0$ and $\epsilon^\Delta > 0$.

Algorithm: Outline of Moré-Sorensen Algorithm [56] & [57]

$[s^*, \lambda^*] = \text{MS}(H, g, \Delta, \epsilon^\Delta)$

Step 1. If $H(0)$ is positive definite and $\|s(0)\| \leq \Delta(1 + \epsilon^\Delta)$, terminate with $s = s(0)$.

Step 2. Determine an interval $\left[\lambda^L, \lambda^U\right]$ and an initial λ in this interval.

Step 3. Factorize $H(\lambda) = LL^T$ with Cholesky factorization.

If this succeeds, solve $LL^T s = -g$.

If $\Delta(1 - \epsilon^\Delta) \leq \|s\| \leq \Delta(1 + \epsilon^\Delta)$, i.e, if s is near the boundary of the trust-region, terminate. If not s is not near the boundary, compute λ^{new} by (3.15).

Step 4. Update the interval $\left[\lambda^L, \lambda^U\right]$:

- if $\|s\| > \Delta(1 + \epsilon^{\Delta})$, or if the factorization has failed, redefine $\lambda^L = \lambda$;
- if $\|s\| < \Delta(1 - \epsilon^{\Delta})$, redefine $\lambda^U = \lambda$.

Step 5. Choose λ sufficiently inside $\left[\lambda^L, \lambda^U\right]$ and as close as possible to λ^{new}, if it has been computed. Go to Step 3.

Despite the fact that any method can be used to solve linear system of equations (3.10), multigrid methods which have the support of a solid convergence theory are used to solve (3.11) in this study [56].

The main goal of the multilevel Moré-Sorensen method is to construct an algorithm for the solution of unconstrained problem $\min_{x \in \mathbb{R}^n} \quad f(x),\ f : \mathbb{R}^n \to \mathbb{R}$ that follows the general pattern of the Moré-Sorensen method but which, meanwhile, exploits the ideas and techniques of multigrid.

CHAPTER 4

MULTILEVEL DERIVATIVE FREE OPTIMIZATION

In many practical trust-region algorithms, the model Q_k is quadratic and obtaining sufficient decrease then amounts to (approximately) solving

$$\min_{\|s\|<\Delta} Q_k(x_k + s) = \min_{\|s\|<\Delta} f(x_k) + < g_k, s > + \frac{1}{2} < s, H_k s > \tag{4.1}$$

for s, where g_k and H_k denote approximations of $\nabla f(x_k)$ and $\nabla^2 f(x_k)$ respectively; $< \cdot, \cdot >$ is the Euclidean inner product and $\| \cdot \|$ is the Euclidean norm.

The work per iteration depends on computing the function value $f(x_k + s_k)$ and dominated by the numerical solution of the subproblem (4.1) which crucially depends on the dimension n of the problem. For optimization problems arising from the discretization of some infinite-dimensional problems like PDEs on a relatively fine grid, the solution cost is therefore often significant. Multilevel optimization algorithms are designed to reduce this cost by exploiting the knowledge of alternative simplified expressions of the objective function, when available. We consider a collection of functions twice-continuously differentiable functions from $\{f_i\}_{i=0}^r, n_i > n_{i-1}$. with $f_r(x) = f(x)$. We assume that, for each $i = 1, \cdots, r, \quad f_i$ is more costly to minimize than f_{i-1}. This situation arises often by discretization infinite dimensional objective functions, where f_i has more variables than f_{i-1} because f_i represent increasingly finer discretizations of the same infinite-dimensional objective function.

The main idea is then to use f_{r-1} to construct an alternative model h_{r-1} for $f_r = f$ in the neighbourhood of the current iterate, that is cheaper than the quadratic model at level r, and to use this alternative model to define the step in the trust region algorithm. For the convergence theory some coherence properties between f_r and its model h_{r-1} is necessary. If the function f_{r-1} is already satisfies these properties, the choice $h_{r-1} = f_r$ is possible. If not, one can choose h_{r-1} as a linear or quadratic modification of f_{r-1}. If more than two levels are available

($r > 1$); this can be done recursively, the approximation process stopping at level 0, where the quadratic model is always used.

In what follows, we use (i, k) notations in which i ($0 \leq i \leq r$), is the level index and k, the index of the current iteration within level i, and is reset to 0 each time level i is entered.

There should be some relation between f_{i-1} and f_i to be useful at all in minimizing f_i. In the context of multigrid algorithms, this relation is established by the restriction and prolongation between operators between a fine and a coarse grids. We do not need any operators between f_i's in contrast to multigrid problems. Because, we use only the information of minimum point of $(i-1)$th level. In other words, while we are in any kth iteration of the level i, we do not take any point from the previous level.

At ith level and kth iteration the quadratic model has the form

$$Q_{i,k}(x_{i,k} + s_{i,k}) = Q_{i,k}(x_{i,k}) + \langle s_{i,k}, \nabla Q_{i,k}(x_{i,k}) \rangle + \frac{1}{2} \left\langle s_{i,k}, \nabla^2 Q_{i,k}(x_{i,k}) s_{i,k} \right\rangle .$$

Then the finer level linear model is defined as:

$$h_{i+1,k}(x_{i+1,k}) = Q_i(x_{i+1,k}) + \left(\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i})\right)(x_{i+1,k} - x_{min,i}) , \tag{4.2}$$

where $x_{min,i}$ is the minimum of the ith level and Q_i is the last model function computed at ith level when $x_{min,i}$ was found and is a fixed function for every iteration k at $(i+1)$th level. Note that $x_{i+1,k} - x_{min,i} = s_{i+1,k} = s$.

The finer level model can be also defined as :

$$h_{i+1,k}(x_{i+1,k}) = f_i(x_{i+1,k}) + \left(\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i})\right)(x_{i+1,k} - x_{min,i}) , \tag{4.3}$$

where f_i is computed the ith level. In this situation, this model will be nonlinear since f_i's are nonlinear.

We choose between the quadratic Taylor model and the cheaper linear model at a finer grid $i+1$ using the following criteria

$$\|\nabla Q_{i+1}\| \geq \kappa_Q \|\nabla Q_i\| \tag{4.4}$$

and

$$\|\nabla Q_{i+1}\| > \epsilon_Q , \tag{4.5}$$

where $\kappa_Q, \epsilon_Q \in (0, 1)$.

If these are satisfied, then the new model will be chosen and the subproblem becomes :

$$\min_{||s_{i+1,k}|| \leq \Delta_{i+1,k}} h_{i+1,k}(x_{min,i} + s_{i+1,k}) \ ,$$

where $s_{i+1,k} = x_{i+1,k} - x_{min,i}$.

Otherwise, the quadratic model is used and Taylor step computation is done. In this case, trust region subproblem will be:

$$\min_{\|s\| \leq \Delta_{i+1,k}} Q_{i+1,k}(x_k + s) = \min_{\|s\| \leq \Delta_{i+1,k}} Q_{i+1,k}(x_k) + \langle s, \nabla Q_{i+1,k}(x_k) \rangle + \frac{1}{2} \left\langle s, \nabla^2 Q_{i+1,k}(x_k) s \right\rangle \ ,$$

where $s = s_{i+1,k}$, $x_k = x_{i+1,k}$.

In the programming, we used the following ways:

- The MATLAB version of DFO based on NFPs [52] is used, which is similar to the original DFO package [49] for minimization of the quadratic model.
- For minimization of the linear model (4.2) or the model (4.3), the Moré and Sorensen subroutine of the package CONDOR is used. From now on, notationally, we will show CONDOR$^+$ for this subroutine.
- When h model ((4.2) or (4.3)) is chosen at any iteration k at level $i+1$, a very important part is the construction of the interpolation set of Q_{i+1}. Two approaches to construct initial interpolation set for Q_{i+1} will be taken into account in the implementation:
  - A poised set is firstly constructed with cardinality $(n + 1)(n + 2)/2$.
    Then, the set is multiplied with initial trust region radius. The minimum value at level i is added to every elements of set.
    In other words, in this case, the minimum point of level i is taken as a starting point at $i + 1$ and used to construct the interpolation set in this way.
  - Or, the final interpolation set at the level i is used for $(i + 1)$th level.

MDFO: *Multilevel Derivative Free Optimization Algorithm*

Step 0: Initialization Given

$x^0 \in \mathbb{R}^n$	:	initial guess
Δ_0	:	initial trust-region
ϵ_Δ	:	Minimum value of the trust-region radius
η_0, η_1	:	parameters to improve quality of interpolation set
γ_0, γ_1	:	trust-region radius decreasing factors
γ_2	:	trust-region radius increasing factor
$\kappa_Q \in (0, 1)$	:	constant for the model choice
$\epsilon_Q \in (0, 1)$	:	gradient norm tolerance
ϵ_{fun}	:	function reduction tolerance.

Construct a well-poised interpolation set Y around $x^0 \in Y \subset \mathbb{R}^n$ within the initial trust region. Build the quadratic model Q_{i+1} using the interpolation set Y and the basis of NFP (Newton Fundamental Polynomial).

Step 1: Model Choice

If $i = 1$ or if the conditions (4.4) and (4.5) fail, go to Step 3 (Taylor step).

Otherwise go to Step 2.

Step 2: Recursive Step Computation

$$\text{Solve} \min_{\|s\|<\Delta} h_{i+1}(x_{min,i} + s) \,,$$

where $h_{i+1}(x_{min,i} + s)$ is in the form (4.2) or (4.3) and $x_{min,i}$ is the minimum of level i.

Step 3: Taylor step computation

To solve the constrained inner trust region sub-problem

$$\min_{\|s\|\leq\Delta_k} \nabla Q^T_{i+1,k}(x)s + \frac{1}{2}s^T\nabla^2 Q_{i+1,k}(x)s \,,$$

- either **trust()** (MATLAB subroutine) using full eigenvalue decomposition, based on the secular equation
$$\frac{1}{\Delta} - \frac{1}{\|s\|} = 0,$$
- or, **lmlib()** (MATLAB subroutine) which uses the method of the Levenberg-Marquardt algorithm with the Moré and Sorensen technique [19], [20] can be used.

Step 4: Acceptance of the trial point

Compute the ratio

$$\rho_{i+1,k} = \rho_k = \frac{f_{i+1}(x_k) - f_{i+1}(\hat{x}_k)}{\delta_{i+1,k}} ,$$

where $\hat{x}_k = \hat{x}_{i+1,k} = x_{i+1,k} + s_{i+1,k}$, $\delta_{i+1,k} = Q_{i+1}(x_k) - Q_{i+1}(\hat{x}_k)$ (if Taylor step is used at this kth iteration) or $\delta_{i+1,k} = h_{i+1}(x_k) - h_{i+1}(\hat{x}_k)$ (if recursive step works). Update the interpolation set :

- if $\rho_k \geq \eta_0$, include $\hat{x}_k$ in Y, dropping the one of the existing interpolation points.

  The procedure of dropping a point from interpolation set:

  Let x^c be the current trust region center point, x^f be the furthest point from x^c and x^w be the point with the worst function value in the interpolation point set. Then we choose a point to drop by applying the following criteria:

  if $\|x^f - x^c\| > 4\Delta$ or ($\|x^f - x^c\| > \Delta$ and $f(x^f) > \frac{1}{N}\sum_{i=1}^{N} f(x^i)$)

  - then drop = x^f,
  - else drop = x^w,

  where N is the number of points in the interpolation set Y.

- if $\rho_k < \eta_0$, include $\hat{x}_k$ in Y, if it improves the quality of model.

- if $\rho_k < \eta_0$ and there are less than $n + 1$ points in the intersection of Y and B_k, generate a new interpolation point in B_k, while preserving or improving well poisedness.

- Update the basis of NFP (Newton Fundemental Polynomials) by using updated interpolation set Y.

Step 5: Update the current iterate

Determine $\bar{x}_k$ with the best objective function value

$$f(\bar{x}_k) = \min_{x_j \in Y, x_j \neq x_k} f(x_j) .$$

If improvement (as in DFO based on NFPs) is sufficient

$$\bar{\rho}_{i+1,k} \equiv \frac{f_{i+1}(x_k) - f_{i+1}(\bar{x}_k)}{\delta_{i+1,k}} \geq \eta_0 ,$$

(where $x_k = x_{i+1,k}$) set $x_{k+1} = \bar{x}_k$.

Otherwise, set $x_{k+1} = x_k$.

Step 6: Trust-region radius update

- if $\rho_k \geq \eta_1$, increase the trust region radius

$$\Delta_{i+1,k+1} \in [\Delta_{i+1,k}, \gamma_2 \Delta_{i+1,k}] \, .$$

- if $\rho_k < \eta_0$ and the cardinality of $Y \cap B_k$ was less than $n + 1$ when $\hat{x}_k$ was computed, reduce the trust region radius

$$\Delta_{i+1,k+1} \in [\gamma_0 \Delta_{i+1,k}, \gamma_1 \Delta_{i+1,k}] \, .$$

- Otherwise set $\Delta_{i+1,k+1} = \Delta_{i+1,k}$.

Step 7: Termination

The algorithm is terminated when one of the following three criteria are satisfied:

- The radius of trust region is small enough, such that $\Delta \leq \epsilon_\Delta$.
- Final interpolation point set has the '*good geometry*' property, if
  - at least $n+1$ points are in the trust region. **checkMod** is a routine in DFO algorithm to check the quadratic model to guarantee that there are at least $n + 1$ points inside the current trust region. If there are less than $n + 1$ points, we replace the point x^f that is furthest away from the current trust region center by $\tilde{x}$ such that

$$\tilde{x} = \text{argmax} \left\{ \det(M(\phi, \bar{Y})) : \bar{Y} = Y \setminus \left\{ x^f \right\} \cup \{x\}, \|x - x^c\| \leq \Delta \right\} \, ,$$

    which is a point on the boundary of the current trust region.
  - for each point x^i,

$$\text{dist}(x^i - x^c) < 2\Delta \, ,$$

    where x^c is the current center point, Δ is the current trust region radius and the interpolation set is well poised.
- Maximum function evaluations or number of maximum iterations are reached.

Increment k by one (i.e. $k := k + 1$) and go to Step 0.

4.1 Convergence of the Recursive Multilevel Derivative Free Method

We will consider the case that the interpolation set is constructed from the last iteration of previous level i.

If the conditions (4.4) and (4.5) are satisfied the following trust region subproblem will be solved:

$$\min_{\|s\|<\Delta} h_{i+1,k}(x_{min,i}+s) = Q_i(x_{i+1,k}) + (\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}))(x_{i+1,k} - x_{min,i}) \ ,$$

where $Q_{i+1,k}$ and $x_{i+1,k}$ changes for every iteration k but Q_i as a function and $x_{min,i}$ come from the previous level are fixed and $x_{i+1,k} - x_{min,i} = s_{i+1,k}$.

Our aim is to show that MDFO converge to first-order critical points when the model (4.2) is chosen. Generally, $x_{i+1,k} - x_{min,i} = s_{i+1,k} = s$ will be denoted throughout convergence for simplicity and all norms are Euclidean norm.

The convergence analysis is based on minimization of derivative free trust region subproblem as given in [10], [13] and [53].

Assumptions :

A.1 : The objective function f is twice continuously differentiable and its Hessian is uniformly bounded over $\mathbb{R}^n$, which means that there exists a positive constant κ_1 such that,for all $x_{i,k} \in \mathbb{R}^n$,

$$\left\|\nabla^2 f(x_{i,k})\right\| \leq \kappa_1 \ ,$$

where $\kappa_1 \geq 1$ is assumed as in [53].

A.2 : The objective function f is bounded below on $\mathbb{R}^n$.

A.3 : The Hessians of all models generated are uniformly bounded, that is there exists a constant $\kappa_2 > 0$ such that

$$1 + \left\|H_{i,k}\right\| \leq \kappa_2 \ ,$$

where $H_{i,k} = \nabla^2 h_{i,k}$. The constant $\kappa_h = \max[\kappa_1, \kappa_2]$ will be used later.

Note that A.1 is like in [53] and A.2 and A.3 is given in [13].

Some definitions, lemmas and theorems for the convergence analysis are needed.

To define *adequate geometry of the interpolation set* as in [10], we will use our criterias in our usage. Therefore, we can define like in [17] :

Definition 4.1.1 *An interpolation set Y is* ***adequate*** *in* $\mathcal{B}_{i,k}(\Delta_{i,k})$ *whenever*

- *the cardinality of Y is at least* $n + 1$ *(meaning the model is at least fully linear) in the trust region where*

$$x^j \in \mathcal{B}_{i,k}(\Delta_{i,k}) \qquad \text{for all } x^j \in Y$$

- x^c *is the current center point, for each point* x^j *it holds*

$$dist(x^j - x^c) < 2\Delta_{i,k} \ ,$$

where $\Delta_{i,k}$ *is the trust region radius of any level i for every iteration k and the interpolation set is well-poised.*

We now look at the relation between the objective function and the model function h.

Theorem 4.1.2 *Assume (A.1) - (A.3) hold and if the h model is chosen, then*

$$\left|f_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k})\right| \leq 3\kappa_{ay} \max[\Delta^2_{i+1,k}, \Delta^3_{i+1,k}]$$

for all $x_{i+1,k} \in \mathcal{B}_{i+1,k}(\Delta_{i+1,k})$ *and some constant* $\kappa_{ay} > 0$*. Here,* $h_{i+1} = h_{i+1,k}$.

Proof.

Note that we can liken from DFO convergence analysis as in Theorem 4 of [10],

$$\left|f_i(x_{i+1,k}) - Q_i(x_{i+1,k})\right| \leq \kappa_{md} \max[\Delta^2_{i+1,k}, \Delta^3_{i+1,k}] \tag{4.6}$$

$$\left\|\nabla f_i(x_{i+1,k}) - \nabla Q_i(x_{i+1,k})\right\| \leq \kappa_{gd} \max[\Delta_{i+1,k}, \Delta^2_{i+1,k}] \tag{4.7}$$

and $\left\|s_{i+1,k}\right\| \leq \Delta_{i+1,k}$.

Like (4.6) we can write

$$\left|f_{i+1}(x_{i+1,k}) - Q_{i+1}(x_{i+1,k})\right| \leq \kappa_{md} \max\left[\Delta^2_{i+1,k}, \Delta^3_{i+1,k}\right] \tag{4.8}$$

for all $x_{i+1,k} \in \mathcal{B}_{i+1,k}(\Delta_{i+1,k})$ and some constant $\kappa_{md} > 0$.

In the following steps of the proof, without loss of generality, we assume that

- if the kth iteration of $i+1$th level is Taylor model, then we can guess $\left\|\nabla Q_{i+1,k}\right\|$ will go to zero since DFO convergence will work. But, if not, we will assume $\left\|\nabla Q_{i+1,k}(x_{min,i})\right\| \leq \Delta_{i+1,k}$. Actually, we will show later $\Delta_{i+1,k}$ will go to zero.
- $\left|Q_{i+1,k} - Q_{i,k}\right| \leq \Delta_{i+1,k}^2$.

At level $i+1$, kth iteration

$$\left|f_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k})\right| = \left|f_{i+1}(x_{i+1,k}) - Q_i(x_{i+1,k}) - (\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}))\, s_{i+1,k}\right|. \tag{4.9}$$

Now, we can find a bound for :

$$\begin{aligned} \left\|\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i})\right\| &\leq \left\|\nabla Q_{i+1,k}(x_{min,i})\right\| + \left\|\nabla Q_i(x_{min,i})\right\| \\ &\leq \left\|\nabla Q_{i+1,k}(x_{min,i})\right\| + \tfrac{1}{\kappa_Q}\left\|\nabla Q_{i+1,k}(x_{min,i})\right\| \\ &\leq (1+\tfrac{1}{\kappa_Q})\left\|\nabla Q_{i+1,k}(x_{min,i})\right\| \\ &\leq \kappa_m \left\|\nabla Q_{i+1,k}(x_{min,i})\right\|, \end{aligned} \tag{4.10}$$

where the condition $\left\|\nabla Q_{i+1,k}\right\| \geq \kappa_Q \left\|\nabla Q_i\right\|$ is used and $\kappa_m = 1 + 1/\kappa_Q \geq 2$.

Adding and substracting $Q_{i+1}(x_{i+1,k})$ to (4.9) and considering triangle inequality and Cauchy-Schwarz inequality and using (4.8) and the assumptions that we assumed at first

$$\begin{aligned} \left|f_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k})\right| &\leq \left|f_{i+1}(x_{i+1,k}) + Q_{i+1}(x_{i+1,k}) - Q_{i+1}(x_{i+1,k}) - Q_i(x_{i+1,k})\right| \\ &+ \left\|(\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}))\, s_{i+1,k}\right\| \\ &\leq \left|f_{i+1}(x_{i+1,k}) - Q_{i+1}(x_{i+1,k})\right| + \left|Q_{i+1}(x_{i+1,k}) - Q_i(x_{i+1,k})\right| \\ &+ \left\|(\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}))\right\| \left\|s_{i+1,k}\right\| \\ &\leq \kappa_{md} \max[\Delta_{i+1,k}^2, \Delta_{i+1,k}^3] + \kappa_m \left\|\nabla Q_{i+1,k}(x_{min,i})\right\| \Delta_{i+1,k} + \left|Q_{i+1,k} - Q_{i,k}\right| \\ &\leq \kappa_{md} \max[\Delta_{i+1,k}^2, \Delta_{i+1,k}^3] + \kappa_m \Delta_{i+1,k}^2 + \Delta_{i+1,k}^2 \\ &\leq 3\kappa_{ay} \max[\Delta_{i+1,k}^2, \Delta_{i+1,k}^3], \end{aligned}$$

where $\kappa_{ay} \overset{def}{=} \max[\kappa_{md}, \kappa_m, 1]$

■

Theorem 4.1.3 *Assume (A.1) - (A.3) hold and if the model (4.2) is chosen, then*

$$\left\|\nabla f_{i+1}(x_{i+1,k}) - \nabla h_{i+1}(x_{i+1,k})\right\| \leq \kappa_{mat} \max[\Delta_{i+1,k}, \Delta_{i+1,k}^2]$$

for some constant κ_{mat} and for all $x_{i+1,k} \in \mathcal{B}_{i+1,k}(\Delta_{i+1,k})$.

Proof.

Similar to (4.7) we can get

$$\left\|\nabla f_{i+1}(x_{i+1,k}) - \nabla Q_{i+1,k}(x_{i+1,k})\right\| \leq \kappa_{gd} \max\left[\Delta_{i+1,k}, \Delta_{i+1,k}^2\right] \tag{4.11}$$

where $x_{i+1,k} \in \mathcal{B}_{i+1,k}(\Delta_{i+1,k})$ and $\kappa_{gd} > 0$ is a constant.

$$\nabla h_{i+1,k} = \nabla Q_i(x_{i+1,k}) + \nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}) \,,$$

where $h_{i+1,k} = h_{i+1}(x_{i+1,k})$. Therefore,

$$\nabla f_{i+1,k} - \nabla h_{i+1,k} = \nabla f_{i+1}(x_{i+1,k}) - \nabla Q_i(x_{i+1,k}) - \nabla Q_{i+1,k}(x_{min,i}) + \nabla Q_i(x_{min,i}) \,. \tag{4.12}$$

Adding and substracting $\nabla Q_{i+1}(x_{i+1,k})$ to (4.12),

$$\begin{aligned} \nabla f_{i+1}(x_{i+1,k} - \nabla h_{i+1,k}(x_{i+1,k})) &= \nabla f_{i+1}(x_{i+1,k}) - \nabla Q_i(x_{i+1,k}) - \nabla Q_{i+1,k}(x_{min,i}) + \nabla Q_i(x_{min,i}) \\ &+ \nabla Q_{i+1,k}(x_{i+1,k}) - \nabla Q_{i+1,k}(x_{i+1,k}); \end{aligned}$$

taking norm of (4.12) and using triangle inequality, we get

$$\begin{aligned} \left\|\nabla f_{i+1}(x_{i+1,k} - \nabla h_{i+1,k}(x_{i+1,k}))\right\| &\leq \left\|\nabla f_{i+1}(x_{i+1,k}) - \nabla Q_{i+1,k}(x_{i+1,k})\right\| + \left\|\nabla Q_{i+1,k}(x_{i+1,k}) - \nabla Q_{i+1,k}(x_{min,i})\right\| \\ &+ \left\|\nabla Q_i(x_{i+1,k}) - \nabla Q_i(x_{min,i})\right\| \,. \end{aligned}$$

Without loss of generality, as in Assumption 10.3 in [13], we assume that ∇Q is Lipschitz continuous for all levels. Therefore,

$$\left\|\nabla Q_i(x_{i+1,k}) - \nabla Q_i(x_{min,i})\right\| \leq \kappa \left\|x_{i+1,k} - x_{min,i}\right\| = \kappa \|s\| \leq \kappa \Delta_{i+1,k}$$

$$\left\|\nabla Q_{i+1,k}(x_{i+1,k}) - \nabla Q_{i+1,k}(x_{min,i})\right\| \leq \acute{\kappa} \|s\| \leq \acute{\kappa} \Delta_{i+1,k} \,,$$

where κ and $\acute{\kappa}$ are constants independent of $\Delta_{i+1,k}$.

Thus, using last two inequality and (4.11)

$$\begin{aligned} \left\|\nabla f_{i+1}(x_{i+1,k} - \nabla h_{i+1,k}(x_{i+1,k}))\right\| &\leq \kappa_{gd} \max[\Delta_{i+1,k}, \Delta_{i+1,k}^2] + \kappa \Delta_{i+1,k} + \acute{\kappa} \Delta_{i+1,k} \\ &\leq \kappa_{mat} \max[\Delta_{i+1,k}, \Delta_{i+1,k}^2] \,, \end{aligned}$$

where $\kappa_{mat} \overset{def}{=} \kappa_{gd} + \kappa + \acute{\kappa}$.

■

- $\mathcal{S} = \{k \mid \bar{\rho}_{i+1,k} \geq \eta_0\}$ (index set of all successful iterations).

- $\mathcal{R} = \{k \mid \Delta_{i+1,k+1} < \Delta_{i+1,k}\}$ (index set of all iterations where the trust region radius is reduced).

Lemma 4.1.4 *Similar to Lemma 5 in DFO convergence part of [10],*

i For all k, if $\rho_{i+1,k} \geq \eta_0$, then $\bar{\rho}_{i+1,k} \geq \eta_0$ and thus iteration k is successful.

ii If $k \in \mathcal{R}$, then Y is adequate in $\mathcal{B}_{i+1,k}(\Delta_{i+1,k})$.

iii There are finite number of improvements of the geometry in the implementation of MDFO (like in DFO based on NFP), unless $\nabla f_{i+1}(x_{i+1,k}) = 0$.

iv There can only be a finite number of iterations such that $\rho_{i+1,k} < \eta_1$ before the trust region radius is reduced in second item of Step 6 in MDFO.

Proof.

i If $\rho_{i+1,k} \geq \eta_0$, $x_{i+1,k} + s_{i+1,k}$ is added to the interpolation set Y by Step 4. And it can be written by Step 5,

$$f(\bar{x}_{i,k}) \leq f(x_{i,k} + s_{i,k})$$

Thus

$$\begin{aligned} \bar{\rho}_{i+1,k} = \frac{f(x_{i+1,k}) - f(\bar{x}_{i+1,k})}{h_{i+1,k}(x_{i+1,k}) - h_{i+1,k}(x_{i+1,k} + s_{i+1,k})} &\geq \frac{f(x_{i+1,k}) - f(x_{i+1,k} + s_{i+1,k})}{h_{i+1,k}(x_{i+1,k}) - h_{i+1,k}(x_{i+1,k} + s_{i+1,k})} \\ &= \rho_{i+1,k} \geq \eta_0. \end{aligned}$$

So, $k \in \mathcal{S}$ and k is successful.

Proofs of (ii), (iii), (iv) is in the same way as in DFO convergence [10]. ■

When the model Q_k is linear or quadratic, 'Cauchy point decrease' condition is assumed in (AA.1) of [53] after relating with 'Cauchy point' condition. Therefore, we will also assume that for all iterations k in any level i,

$$Q_{i,k}(x_k) - Q_{i,k}(x_k + s_k) \geq \kappa_b \left\|g_{i,k}\right\| \min\left\{\frac{\left\|g_{i,k}\right\|}{1 + \left\|H_{i,k}\right\|}, \Delta_{i,k}\right\}, \tag{4.13}$$

where $g_{i,k} = \nabla_s Q_{i,k}(x_k), H_{i,k} = \nabla^2_{ss} Q_{i,k}(x_k), \kappa_b \in (0, 1)$.

In the following lemma, main requirement is to show how the 'Cauchy point condition' for the h model will be valid.

Lemma 4.1.5 *At every iteration k at level i + 1, one has*

$$h_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k} + s_{i+1,k}) \geq \frac{\kappa_Q \kappa_b}{2 + \kappa_Q} \left\|\nabla h_{i+1,k}\right\| \min\left[\frac{\kappa_Q \left\|\nabla h_{i+1,k}\right\|}{(2 + \kappa_Q)(1 + \|H_{i+1}\|)}, \Delta_{i+1,k}\right]$$

for some constant $\kappa_b \in (0, 1)$ *independent of k and* $H_{i+1} = \nabla^2 Q_{i+1}$ *and* $h_{i+1,k} = h_{i+1,k}(x_{i+1,k})$.

Proof. For simplicity, sometimes $x_k = x_{i+1,k}$, $s = s_{i+1,k} = x_{i+1,k} - x_{min,i}$ will be written.

$$\begin{aligned} h_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k} + s) &= Q_i(x_{i+1,k}) + (\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}))s - Q_i(x_{i+1,k} + s) \\ &- (\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}))\,(x_{i+1,k} + s - x_{min,i}) \\ &= Q_i(x_{i+1,k}) - Q_i(x_{i+1,k} + s) - (\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}))s. \end{aligned}$$

Note that by Taylor expansion for general forms (not vectoral)

$$Q_{i+1,k}(x_{min,i} + s) = Q_{i+1,k}(x_{min,i}) + s\nabla Q_{i+1,k}(x_{min,i}) + 1/2 s^2 \nabla^2 Q_{i+1,k}(\xi_k)\,,$$

where $\xi_k \in (x_{min,i}, x_{min,i} + s)$. This equation can be expressed equivalently as

$$Q_{i+1,k}(x_{min,i}) - Q_{i+1,k}(x_{min,i} + s) = -s\nabla Q_{i+1,k}(x_{min,i}) - 1/2 s^2 \nabla^2 Q_{i+1,k}(\xi_k)\,. \tag{4.14}$$

Assume that

- $-(\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i}))s \geq -\kappa_m \nabla Q_{i+1,k}(x_{min,i})s$
- $1/2 s^2 \nabla^2 Q_{i+1,k}(\xi_k) = 1/2 s^T \nabla^2 Q_{i+1,k}(\xi_k)s \geq 0$ and
- $\left\|\nabla Q_{i+1,k}(x_{i+1,k})\right\| \leq \left\|\nabla Q_{i+1,k}(x_{min,i})\right\|$.

By Theorem 6.3.4 in [53],

$$Q_i(x_{i+1,k}) - Q_i(x_{i+1,k} + s) \geq 0.$$

Therefore, using the assumptions above, we obtain

$$\begin{aligned} h_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k} + s) &\geq (\nabla Q_i(x_{min,i}) - \nabla Q_{i+1,k}(x_{min,i}))s \\ &\geq -\kappa_m \nabla Q_{i+1,k}(x_{min,i})s \\ &= \kappa_m \left(Q_{i+1,k}(x_{min,i}) - Q_{i+1,k}(x_{min,i} + s) + 1/2 s^2 \nabla^2 Q_{i+1,k}(\xi_k)\right) \\ &\geq Q_{i+1,k}(x_{min,i}) - Q_{i+1,k}(x_{min,i} + s), \end{aligned}$$

where $\kappa_m \geq 2$ and (4.14) is used. Using (4.13), we can write

$$\begin{aligned} h_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k}+s) &\geq Q_{i+1}(x_{min,i}) - Q_{i+1}(x_{min,i}+s) \\ &\geq \kappa_b \left\|\nabla Q_{i+1,k}(x_{min,i})\right\| \min\left[\frac{\|\nabla Q_{i+1,k}(x_{min,i})\|}{1+\|H_{i+1}\|}, \Delta_{i+1,k}\right], \end{aligned}$$

where $H_{i+1} = \nabla^2 Q_{i+1,k}(x_{min,i})$.

Taking into account $\left\|\nabla h_{i+1,k}\right\|$,

$$\begin{aligned} \left\|\nabla h_{i+1,k}\right\| &= \left\|\nabla Q_i(x_{i+1,k}) + \nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i})\right\| \\ &\leq \left\|\nabla Q_i(x_{i+1,k})\right\| + \left\|\nabla Q_{i+1,k}(x_{min,i}) - \nabla Q_i(x_{min,i})\right\| \\ &\leq \tfrac{1}{\kappa_Q}\left\|\nabla Q_i(x_{i+1,k})\right\| + \kappa_m \left\|\nabla Q_{i+1,k}(x_{min,i})\right\| \\ &\leq (\tfrac{1}{\kappa_Q} + \kappa_m)\left\|\nabla Q_{i+1,k}(x_{min,i})\right\|, \end{aligned} \tag{4.15}$$

where (4.4), (4.10) and the third assumption above is used. Putting the definition of $\kappa_m = 1 + 1/\kappa_Q$ into (4.15) gives

$$\begin{aligned} \left\|\nabla h_{i+1,k}\right\| &\leq \left(\frac{1}{\kappa_Q} + 1 + \frac{1}{\kappa_Q}\right)\left\|\nabla Q_{i+1,k}(x_{min,i})\right\| \\ \frac{\kappa_Q}{\kappa_Q+2}\left\|\nabla h_{i+1,k}\right\| &\leq \left\|\nabla Q_{i+1,k}(x_{min,i})\right\|. \end{aligned} \tag{4.16}$$

In the main discussion, taking the previous result,

$$\begin{aligned} h_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k}+s) &\geq \kappa_b \left\|\nabla Q_{i+1,k}(x_{min,i})\right\| \min\left[\frac{\|\nabla Q_{i+1,k}(x_{min,i})\|}{(1+\|H_{i+1}\|)}, \Delta_{i+1,k}\right] \\ &\geq \kappa_b \tfrac{\kappa_Q}{2+\kappa_Q}\left\|\nabla h_{i+1,k}\right\| \min\left[\frac{\kappa_Q\|\nabla h_{i+1,k}\|}{(2+\kappa_Q)(1+\|H_{i+1}\|)}, \Delta_{i+1,k}\right], \end{aligned}$$

where $\kappa_b,\ \kappa_Q/(2+\kappa_Q) \in (0,1)$ since $\kappa_Q \in (0,1)$. Observe also that $\left\|\nabla^2 h_{i+1}\right\| = \left\|\nabla^2 Q_{i+1}\right\| = \left\|H_{i+1,k}\right\|$ since

$$\left\|\nabla^2 h_{i+1,k}\right\| = \left\|\nabla^2 Q_i(x_k) + \nabla^2 Q_{i+1,k}(x_{min,i}) - \nabla^2 Q_i(x_{min,i})\right\|,$$

where $\nabla^2 Q_i(x_k) = \nabla^2 Q_i(x_{min,i})$ due to constant value result of second derivative of at most second degree function Q_i. ■

Lemma 4.1.6 *Assume that* $(A.1) - (A.3)$ *hold. Furthermore assume that* $\nabla h_{i+1,k} \neq 0$ *and that*

$$\Delta_{i+1,k} \leq \min\left[1, \frac{\kappa_Q \kappa_b \left\|\nabla h_{i+1,k}\right\|(1-\eta_1)}{(2+\kappa_Q)6\max[\kappa_h, \kappa_{ay}]}\right]. \qquad (*)$$

Then iteration k is very successful and

$$\Delta_{i+1,k+1} \geq \Delta_{i+1,k}.$$

Proof. Note that $\eta_1, \kappa_b \in (0,1)$. Thus,

$$\kappa_b(1-\eta_1) < 1$$

and by definitions of κ_{ay}, κ_h and putting these in (*),

$$\frac{\left\|\nabla h_{i+1,k}\right\|}{6\max[\kappa_h,\kappa_{ay}]} \leq \frac{\left\|\nabla h_{i+1,k}\right\|}{6\kappa_h} \leq \frac{\left\|\nabla h_{i+1,k}\right\|}{6\kappa_2} \leq \frac{\left\|\nabla h_{i+1,k}\right\|}{\kappa_2} \leq \frac{\left\|\nabla h_{i+1,k}\right\|}{(1+\|H_{i+1}\|)} ,$$

where $\kappa_2 > 1 + \|H_{i+1}\|$ and $\kappa_h = \max[\kappa_1, \kappa_2]$.

Using these two inequality, we get

$$\Delta_{i+1,k} \leq \frac{\kappa_Q\kappa_b \left\|\nabla h_{i+1,k}\right\| (1-\eta_1)}{(2+\kappa_Q)6\max\left[\kappa_h,\kappa_{ay}\right]} \leq \frac{\kappa_Q \left\|\nabla h_{i+1,k}\right\|}{(2+\kappa_Q)(1+\|H_{i+1}\|)} .$$

Combining this inequality with Lemma(4.1.5), at iteration k,

$$\begin{aligned} h_{i+1}(x_{i+1,k}) - h_{i+1}(x_{i+1,k}+s_{i+1,k}) &\geq \frac{\kappa_Q\kappa_b}{2+\kappa_Q}\left\|\nabla h_{i+1,k}\right\| \min\left[\frac{\kappa_Q\|\nabla h_{i+1,k}\|}{(2+\kappa_Q)(1+\|H_{i+1}\|)}, \Delta_{i+1,k}\right] \\ &= \frac{\kappa_Q\kappa_b\|\nabla h_{i+1,k}\|\Delta_{i+1,k}}{2+\kappa_Q} . \end{aligned}$$

Now, we can write

$$\begin{aligned} |\rho_{i+1}-1| &= \left|\frac{f_{i+1}(x_{i+1,k})-f_{i+1}(x_{i+1,k}+s)}{h_{i+1}(x_{i+1,k})-h_{i+1}(x_{i+1,k}+s)} - 1\right| \\ &\leq \left|\frac{f_{i+1}(x_{i+1,k}+s)-h_{i+1}(x_{i+1,k}+s)}{h_{i+1}(x_{i+1,k})-h_{i+1}(x_{i+1,k}+s)}\right| + \left|\frac{f_{i+1}(x_{i+1,k})-h_{i+1}(x_{i+1,k})}{h_{i+1}(x_{i+1,k})-h_{i+1}(x_{i+1,k}+s)}\right| \\ &\leq 2\frac{3(2+\kappa_Q)\kappa_{ay}\max[\Delta_{i+1,k}^2,\Delta_{i+1,k}^3]}{\kappa_Q\kappa_b\|\nabla h_{i+1,k}\|\Delta_{i+1,k}} = \frac{6(2+\kappa_Q)\kappa_{ay}\max[\Delta_{i+1,k},\Delta_{i+1,k}^2]}{\kappa_Q\kappa_b\|\nabla h_{i+1,k}\|} \\ &\leq \frac{6(2+\kappa_Q)\kappa_{ay}\Delta_{i+1,k}}{\kappa_Q\kappa_b\|\nabla h_{i+1,k}\|} \\ &\leq 1-\eta_1 , \end{aligned}$$

where last inequality comes from assumption on $\Delta_{i+1,k}$ at first. That is,

$$\Delta_{i+1,k} \leq \frac{\kappa_Q\kappa_b \left\|\nabla h_{i+1,k}\right\| (1-\eta_1)}{6(2+\kappa_Q)\max[\kappa_h,\kappa_{ay}]} \leq \frac{\kappa_Q\kappa_b \left\|\nabla h_{i+1,k}\right\| (1-\eta_1)}{6(2+\kappa_Q)\kappa_{ay}} .$$

To show how the penultimate inequality $\max[\Delta_{i+1,k}, \Delta_{i+1,k}^2] \leq \Delta_{i+1,k}$ is, note that in level $i+1$ (for simplicity, $i+1$ will not be written for now), it holds $\frac{\kappa_b\kappa_Q(1-\eta_1)}{(2+\kappa_Q)} < 1$ since $\kappa_Q, \kappa_b \in (0,1)$, and in (*), $\frac{\|\nabla h\|}{6\max[\kappa_h,\kappa_{ay}]} \leq \frac{\|\nabla h\|}{\kappa_h}$. By the way,

$$\Delta_k \leq \min\left[1, \frac{\|\nabla h\|}{\kappa_h}\right] \quad \text{and} \quad \Delta_k^2 \leq \min\left[\Delta_k, \frac{\|\nabla h\|}{\kappa_h}\Delta_k\right] .$$

Thus, we can write

$$\begin{aligned}\max\left[\Delta_k, \Delta_k^2\right] &\leq \max\left[\Delta_k, \min\left[\Delta_k, \frac{\|\nabla h\|}{\kappa_h}\Delta_k\right]\right] \\ &= \Delta_k \max\left[1, \min\left[1, \frac{\|\nabla h\|}{\kappa_h}\right]\right] \\ &= \Delta_k.\end{aligned}$$

Therefore, $\rho_{i+1} \geq \eta_1$ and the iteration is very successful. Furthermore, by the first item of step 6 in MDFO, $\Delta_{i+1,k+1} \geq \Delta_{i+1,k}$. ■

Theorem 4.1.7 *Assume that (A.1) - (A.3) hold and* $\left\|\nabla h_{i+1,k}\right\| \geq \kappa_c$. *Then there exists a constant* κ_n *such that*

$$\Delta_{i+1,k} > \kappa_n \ .$$

Proof. Assume that iteration k is the first k ($\rho_k < \eta_0$) such that

$$\Delta_{i+1,k+1} \leq \min\left[1, \frac{\gamma_0 \kappa_Q \kappa_b \kappa_c (1-\eta_1)}{6(2+\kappa_Q)\max[\kappa_h, \kappa_{ay}]}\right]. \qquad (**)$$

Then from the second item of Step 6, we have $\gamma_0 \Delta_{i+1,k} \leq \Delta_{i+1,k+1}$ and, hence,

$$\Delta_{i+1,k} \leq \min\left[1, \frac{\kappa_Q \kappa_b \kappa_c (1-\eta_1)}{6(2+\kappa_Q)\max[\kappa_h, \kappa_{ay}]}\right].$$

Assumption on $\left\|\nabla h_{i+1,k}\right\|$ implies that (*) holds and, thus, that k should be very successful and $\Delta_{i+1,k+1} \geq \Delta_{i+1,k}$ satisfied. But this contradicts the fact that iteration is the first such that (**) holds, and initial assumption is therefore impossible. So,

$$\kappa_n = \gamma_0 \min\left[1, \frac{\kappa_Q \kappa_b \kappa_c (1-\eta_1)}{6(2+\kappa_Q)\max[\kappa_h, \kappa_{ay}]}\right].$$

■

Theorem 4.1.8 *Assume (A.1) - (A.3) hold. Furthermore assume that there are only finitely many successful iterations. Then* $x_{i+1,k} = x_{i+1,*}$ *for k sufficiently large and* $\nabla f_{i+1}(x_{i+1,*}) = 0$.

Proof. In an infinite loop within Step 0, the result follows from Lemma 4.1.4(iii). Otherwise, the mechanism of the algorithm ensure that

$$x_{i+1,*} = x_{i+1,k_0+1} = x_{i+1,k_0+j} \qquad \text{for all } j > 0$$

where k_0 is the index of the last successful iteration.

Moreover, since all iterations are unsuccessful for sufficiently large k, from Lemma 4.1.4 (i) that $\rho_{i+1,k} < \eta_0 < \eta_1$ and second item of Step 6 along with Lemma 4.1.4 (iv) implies the sequence $\{\Delta_{i+1,k}\} \to 0$. Hence $\mathcal{R}$ contains at least one infinite subsequence k_j.

Further, Lemma 4.1.4 (ii) implies that Y is adequate in $\mathcal{B}_{i,k_j}$ for all j.

Now, assume that $\nabla f_{i+1}(x_{i+1,*}) \neq 0$. Using Theorem 4.1.3, for $k > k_0$, we get

$$\left\|\nabla f_{i+1}(x_{i+1,*}) - \nabla h_{i+1}(x_{i+1,k_j})\right\| \leq \kappa_{mat} \max[\Delta_{i+1,k}, \Delta_{i+1,k}^2] \, .$$

Since $\{\Delta_{i+1,k}\} \to 0$, then

$$\left\|\nabla h_{i+1}(x_{i+1,k_j})\right\| \geq \kappa_c = \frac{1}{2}\left\|\nabla f_{i+1}(x_{i+1,*})\right\| > 0$$

for $k \geq k_0$ sufficiently large. But if $\left\|\nabla h_{i+1}(x_{i+1,k_j})\right\| > 0$, Lemma 4.1.6 then implies that there must be a successful iteration of index larger than k_0, which is impossible. Hence $\nabla f_{i+1}(x_{i+1,*}) = 0$ and $x_{i+1,*}$ is first-order critical. ■

In the following Lemma as in Lemma 10.9 in [13], it will be shown that the trust region radius converges to zero.

Lemma 4.1.9 *Assume that (A.1) - (A.3) hold. Then*

$$\lim_{k \to +\infty} \Delta_k = 0.$$

Proof. Assume $\mathcal{S}$ is finite. Consider iterations that come after the last successful iteration. By Lemma 4.1.4 (iii), we can have only a finite (uniformly bounded, say by N) number of model-improving iterations before the model becomes fully linear and hence there is an infinite number of iterations that are acceptable or unsuccessful and in either case the trust region radius is reduced.

Since there are no more successful iterations, Δ_k is never increased for sufficiently large k. Moreover, Δ_k is decreased at least once every N iterations by a factor γ. Thus, Δ_k converges to zero.

Consider the case when $\mathcal{S}$ is infinite. For any $k \in \mathcal{S}$ we have

$$f_{i+1}(x_k) - f_{i+1}(x_{k+1}) \geq \eta_0 \left[h_{i+1,k}(x_k) - h_{i+1,k}(x_k + s_k)\right] \, .$$

Using the Cauchy decrease condition, we get

$$f_{i+1}(x_k) - f_{i+1}(x_{k+1}) \geq \eta_0 \frac{\kappa_Q \kappa_b}{2+\kappa_Q} \left\|\nabla h_{i+1,k}\right\| \min\left[\frac{\kappa_Q \left\|\nabla h_{i+1,k}\right\|}{(2+\kappa_Q)\left(1+\|H_{i+1}\|\right)}, \Delta_{i+1,k}\right] .$$

Considering (4.16) and the condition (4.5), we assume that

$$\left\|\nabla h_{i+1,k}\right\| \geq \frac{\epsilon_Q}{2}.$$

$$\begin{aligned} f_{i+1}(x_k) - f_{i+1}(x_{k+1}) &\geq \eta_0 \tfrac{\kappa_Q \kappa_b}{2+\kappa_Q} \tfrac{\epsilon_Q}{2} \min\left[\tfrac{\kappa_Q \epsilon_Q}{2(1+\kappa_Q)(1+\|H_{i+1}\|)}, \Delta_{i+1,k}\right] \\ &\geq \eta_0 \tfrac{\kappa_Q \kappa_b}{2+\kappa_Q} \tfrac{\epsilon_Q}{2} \min\left[\tfrac{\kappa_Q \epsilon_Q}{2(1+\kappa_Q)\kappa_2}, \Delta_{i+1,k}\right] , \end{aligned}$$

where $1 + \|H_{i+1}\| \leq \kappa_2$.

Since $\mathcal{S}$ is infinite and f is bounded from below, the right hand side of the expression above has to converge to zero. Hence, $\lim_{k \in \mathcal{S}} \Delta_k = 0$, and the proof is completed if all iterations are successful.

Recall that the trust-region radius can be increased only during a successful iteration, and it can be increased only by a ratio of at most γ_n which is a constant. Let $k \notin \mathcal{S}$ be the index of an iteration after the first successful one. Then $\Delta_k \leq \gamma_n \Delta_{s_k}$, where s_k is the index of the last successful iteration before k. Since $\Delta_{s_k} \to 0$, then $\Delta_k \to 0$ for $k \notin \mathcal{S}$. ■

When there are infinitely many successful iterations :

Theorem 4.1.10 *Assume (A.1) - (A.3) hold. Then*

$$\lim_{k \to \infty} inf \left\|\nabla h_{i+1,k}\right\| = 0 ,$$

where $h_{i+1,k} = h_{i+1,k}(x_{i+1,k})$.

Proof. For contradiction, assume that

$$\left\|\nabla h_{i+1,k}\right\| \geq \kappa_c$$

for all k and for some $\kappa_c > 0$.

Consider a successful iteration k. So, $k \in \mathcal{S}$ and this implies

$$\begin{aligned} f_{i+1}(x_{i+1,k}) - f_{i+1}(x_{i+1,k} + s_{i+1,k}) &\geq \eta_0 \left[h_{i+1,k}(x_{i+1,k}) - h_{i+1,k}(x_{i+1,k} + s_{i+1,k})\right] \\ &\geq \eta_0 \tfrac{\kappa_Q \kappa_b}{2+\kappa_Q} \kappa_c \min\left[\tfrac{\kappa_Q \kappa_c}{(2+\kappa_Q)(1+\|H_{i+1}\|)}, \Delta_{i+1,k}\right] \\ &\geq \eta_0 \tfrac{\kappa_Q \kappa_b}{2+\kappa_Q} \kappa_c \min\left[\tfrac{\kappa_Q \kappa_c}{(2+\kappa_Q)\kappa_h}, \kappa_n\right] , \end{aligned}$$

since $1 + \|H_{i+1}\| \leq \kappa_2 \leq \kappa_h$ and $\Delta_{i+1,k} > \kappa_n$.

Now, summing over all successful iterations from 0 to k, we obtain that

$$\begin{aligned} f_{i+1}(x_{i+1,0}) - f_{i+1}(x_{i+1,k+1}) &\geq \sum_{\substack{j=0 \\ j\in\mathcal{S}}}^{k} \left[f_{i+1}(x_{i+1,j}) - f_{i+1}(x_{i+1,j+1}) \right] \\ &\geq \sigma_k \frac{\kappa_Q \kappa_b}{2+\kappa_Q} \kappa_c \eta_0 \min\left[\frac{\kappa_Q \kappa_c}{(2+\kappa_Q)\kappa_h}, \kappa_n \right] , \end{aligned}$$

where σ_k is the number of successful iterations up to iteration k.

But since there are infinitely many such iterations, we have

$$\lim_{k\to\infty} \sigma_k = +\infty,$$

and the difference $f_{i+1}(x_{i+1,0}) - f_{i+1}(x_{i+1,k+1})$ is unbounded. This contradicts the fact that the objective function is bounded below. Hence, assumption is false and the desired result is obtained. ■

Lemma 4.1.11 *Assume (A.1) - (A.3) hold and that $\{k_i\}$ is a subsequence such that*

$$\lim_{i\to\infty} \left\| \nabla h_{i+1,k_i}(x_{i+1,k_i}) \right\| = 0 , \tag{4.17}$$

Then,

$$\lim_{i\to\infty} \left\| \nabla f_{i+1}(x_{i+1,k_i}) \right\| = 0 . \tag{4.18}$$

Proof. By (4.17), $\left\| \nabla h_{i+1,k_i}(x_{i+1,k_i}) \right\| \leq \epsilon_c \in (0,1)$ for sufficiently large i. In the proof of Lemma 4.1.6, we showed that $\Delta_k \leq \frac{\|\nabla h\|}{\kappa_h}$. So, now, assume that for the $\{k_i\}$ subsequence,

$$\Delta_{i+1,k_i} \leq \beta \left\| \nabla h_{i+1,k_i} \right\|$$

for all i and $\beta \in (0,1)$ (thinking $\beta := \frac{1}{\kappa_h}$).

Then, by Theorem 4.1.3, it can be deduced that

$$\begin{aligned} \left\| \nabla f_{i+1}(x_{i+1,k_i}) - \nabla h_{i+1,k_i}(x_{i+1,k_i}) \right\| &\leq \kappa_{mat} \max\left[\Delta_{i+1,k_i}, \Delta_{i+1,k_i}^2 \right] \\ &\leq \kappa_{mat} \max\left[\beta \left\| \nabla h_{i+1,k_i} \right\|, \beta^2 \left\| \nabla h_{i+1,k_i} \right\|^2 \right] \\ &\leq \kappa_{mat}\, \beta \left\| \nabla h_{i+1,k_i} \right\| . \end{aligned}$$

Hence, for sufficiently large i,

$$\left\| \nabla f_{i+1,k_i} \right\| \leq \left\| \nabla h_{i+1,k_i} \right\| + \left\| \nabla f_{i+1,k_i} - \nabla h_{i+1,k_i} \right\| \leq (1 + \kappa_{mat}\beta) \left\| \nabla h_{i+1,k_i} \right\| .$$

The limit (4.17) and this last bound then give (4.18).

■

The following two theorems are the same way as in DFO convergence.

Theorem 4.1.12 *Assume that (A.1) - (A.3) hold. Then there is at least one subsequence of successful iteration* $\{x_{i+1,k}\}$ *whose limit is a critical point, that is,*

$$\lim_{k\to\infty} inf \left\|\nabla f_{i+1}(x_{i+1,k})\right\| = 0 .$$

Theorem 4.1.13 *Assume (A.1) - (A.3) hold. Then every limit point* $x_{i+1,*}$ *of the sequence* $\{x_{i+1,k}\}$ *is critical, that is,*

$$\nabla f_{i+1}(x_{i+1,*}) = 0.$$

Observation 4.1.14 *In this section, we only observed that the chosen h model in* $i+1$ *level at any iteration k. But if after kth iteration algorithm choses (at* $k+1$*th iteration) Taylor model or the last iteration of ith level was chosen Taylor model, then convergence of DFO part will be valid in this situation. Moreover, if the model (4.3) is chosen, then the following logic will work.*

4.1.1 Handling Nonlinear Models in Trust-Region Methods

Consider our model is not a linear model and,

$$Q_k(x_k + s) = Q_k(x_k) + s^T g_k + \frac{1}{2} s^T H_k s \,, \qquad (\#)$$

$$Q_k(x_k) - Q_k(x_k + s_k) \geq \frac{\kappa_{fcd}}{2} \|g_k\| \min\left\{\frac{\|g_k\|}{\|H_k\|}, \Delta_k\right\} , \qquad (\#\#)$$

where (##) is Cauchy decrease condition (4.13) in which $\kappa_{fcd}/2 = \kappa_b$.

An approximated solution of a trust-region subproblem $\min_{s\in\mathcal{B}(0;\Delta_k)} Q_k(x_k + s)$ is defined by a nonlinear model which does not satisfy (##).

Since unapplicability of Cauchy decrease condition to nonlinear models, an equivalent condition is necessary for more general models. One way is to use a *backtracking algorithm* along

the model steepest descent direction, where the backtracking is suggested from the boundary of the trust-region [53].

Assume $Q_k(x_k + s)$ is not a quadratic function in s. Choose the smallest $j \geq 0$ such that

$$x_{k+1} = x_k + \mu^j s, \quad \text{where } s = -\frac{\Delta_k}{\|g_k\|} g_k \text{ and } \mu \in (0,1) \ . \tag{4.19}$$

By the way, a sufficient decrease is of the form

$$Q_k(x_{k+1}) \leq Q_k(x_k) + \kappa_d \mu^j s^T g_k, \qquad \kappa_d \in (0,1) \ , \tag{4.20}$$

From (4.19), (4.20) it can be written equivalently

$$Q_k(x_{k+1}) - Q_k(x_k) \leq -\kappa_d \mu^j \Delta_k \|g_k\| \ . \tag{4.21}$$

Using the mean value theorem on the left hand side,

$$-\mu^j \Delta_k \|g_k\| + \frac{1}{2} \frac{\mu^{2j} \Delta_k^2 g_k^T \nabla^2 Q_k \left(y_{k,j}\right) g_k}{\|g_k\|^2} \leq -\kappa_d \mu^j \Delta_k \|g_k\|$$

for some $y_{k,j} \in \left[x_k, x_k + \mu^j s\right]$.
Assume that $\left\|\nabla^2 Q_k(y_{k,j})\right\| \leq \kappa_{bhm}$. So, (4.21) is satisfied, provided

$$\frac{\mu^j \Delta_k}{\|g_k\|} \leq \frac{2\,(1-\kappa_d)}{\kappa_{bhm}} \ .$$

Thus, a j_k satisfying (4.21) can be found such that $\mu^{j_k} > \frac{2(1-\kappa_d)\mu\|g_k\|}{(\kappa_{bhm}\Delta_k)}$.

Define $s_k^{AC} = \mu^{j_k} s$ as the approximate Cauchy step. Then,

$$Q_k(x_k) - Q_k(x_k + s_k^{AC}) \geq \kappa_d \mu^{j_k} \Delta_k \|g_k\| \ .$$

On the other hand, if the approximate Cauchy step takes us to the boundary, it can be derived from (4.20) that the decrease in the model exceeds or is equal to $\kappa_d \Delta_k \|g_k\|$, and so

$$Q_k(x_k) - Q_k(x_k + s_k^{AC}) \geq \bar{\kappa}_d \|g_k\| \min\left\{\frac{\|g_k\|}{\kappa_{bhm}}, \Delta_k\right\}$$

for a suitably defined $\bar{\kappa}_d > 0$.

Another way of dealing with trust-region models which are not quadratic was suggested by [1].

CHAPTER 5

NUMERICAL RESULTS

Consider a *shape optimization problem* over a rectangular region with a rectangular hole $\Omega(u)$ parametrized with the coordinates $u = (P9, P12)$, as in Figure 5.1,

$$\min \quad \frac{1}{2}\int_{\Omega(u)} (y(x_1, x_2) - y_d)^2 \, dx$$

$$\text{subject to} \quad \begin{aligned} \frac{\partial^2 y}{\partial x_1^2} + \frac{\partial^2 y}{\partial x_2^2} &= 1 \quad \text{in} \quad \Omega(u) \\ y &= 0 \quad \text{on the outer boundary} \quad \partial\Omega(u) \end{aligned}$$

Here, $y(x_1, x_2)$ is state variable, y_d is the desired state and u is the control variable.

Data y_d for objective function is in such a way that for $y_d = y(u_d)$ with u_d takes the global minimum.

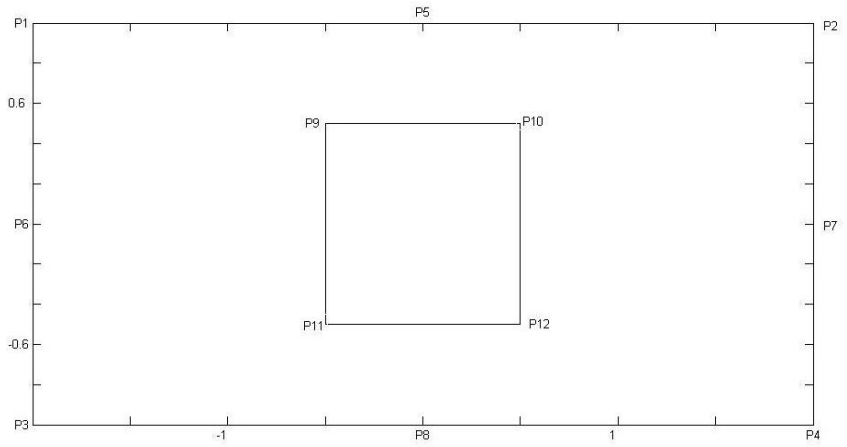

Figure 5.1: Shape Optimization Problem

Since this is a nonlinear control problem, it can be solved using nonlinear programming packages.

Before the results with MDFO is observed, CONDOR [2] and DFO [52] results for this problem can be given as follows. All results are up to 5th level because of the computational complexity at the higher levels.

When CONDOR is applied with parameters determined by user

$u^0 = [-0.5, 0.5, 0.5, -0.5]^T$: the initial guess

$\rho_{start} = 0.1$: initial distance between sample points

$\rho_{end} = 1e-4$: stopping criteria for the distance of the points,

obtained results:

Table 5.1: CONDOR results of shape optimization problem.

Level	func. eval.	func. value	u = [P9;P12]
1	131	3.0482e-014	$[-0.71, 0.22, -0.08, -0.22]^T$
2	82	3.2440e-014	$[-0.75, 0.25, -0.25, -0.25]^T$
3	60	1.0045e-012	$[-0.75, 0.25, -0.25, -0.25]^T$
4	52	1.6304e-011	$[-0.75, 0.25, -0.25, -0.25]^T$
5	61	2.8944e-012	$[-0.75, 0.25, -0.25, -0.25]^T$

If DFO based on NFPs is used with initial parameters which can be modified by any user,

$\epsilon_{trust} = 0.01$	:	minimum value of the trust region radius.
$\epsilon_{det} = 1e-12$	:	determinant tolerance.
$\Delta_0 = 0.2$	:	initial trust region radius.
$\eta_0 = 0.45$	:	parameter to accept the trial point
$\eta_1 = 0.75$	:	parameter to accept the trial point
$\gamma_1 = 0.3$	:	trust region decreasing factor.
$\gamma_2 = 1.5$	:	trust region increasing factor.
$\epsilon_{dist} = 0.001$	:	minimum distance allowed between two points.
$\epsilon_{fun} = \epsilon_{trust} \times 1e-6$	:	minimum value of the function difference.

then the following results are obtained from coarsest level to finest level:

Table 5.2: DFO results of shape optimization problem.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	35	92	1.6467e-009	$[-0.74, 0.25, -0.25, -0.25]^T$
3	36	80	2.4006e-009	$[-0.75, 0.25, -0.25, -0.25]^T$
4	32	87	1.4516e-008	$[-0.75, 0.24, -0.24, -0.24]^T$
5	34	75	2.0589e-010	$[-0.74, 0.25, -0.24, -0.25]^T$

Note that initial parameters in DFO are the same for every iteration and for every level.

MDFO without model choice

When MDFO is applied for the shape optimization problem without any model choice condition, it is explained in the following :

---

for $i = 1 : r$ **do**
  if $i = 1$ **then**
    run DFO with the initial guess $x^0 = [-0.5, 0.5, 0.5, -0.5]^T$
  else
    run DFO with the minimum point of the previous level
    OR run DFO with the interpolation set of previous level
  end if
end for

The trust region subproblem is constructed at different levels in the following ways :

- At level $i = 1$:

  The standard DFO trust region sub-problem is given by

$$\min_{\|s\| \leq \Delta} \quad \nabla Q_i(x)^T s + \frac{1}{2} s^T \nabla^2 Q_i(x) s \ .$$

- At level $i + 1$, $i = 1, \ldots, r - 1$:

  – Q_{i+1} is constructed by using interpolation set.

  – The linear model is also constructed such that

$$h_{i+1}(x_k) = Q_i(x_k) + (\nabla Q_{i+1}(x_{min,i}) - \nabla Q_i(x_{min,i}))(x_k - x_{min,i}) \ ,$$

  where $x_k = x_{i+1,k} = x_{min,i} + s$, $x_{min,i}$ is the minimum point computed at level i and Q_i is the last model function when the minimum point of i was found.

  – The constrained trust region sub-problem

$$\min_{\|s\| < \Delta} \quad h_{i+1}(x_{min,i} + s)$$

  is solved with **CONDOR**$^+$ which is Moré and Sorensen subroutine of CONDOR.

Version 1

The initial interpolation set for Q_{i+1} is constructed around the minimum point of level i. This interpolation set construction at level $i + 1$ works in the way :

- An interpolation set is constructed at the beginning of the DFO.
- The set is multiplied by initial trust region radius.
- The computed minimum point of ith level (starting point of $(i + 1)$th level) is added to the interpolation set.

From now on, all initial parameters will be taken as in DFO based on NFPs. We will emphasize only the parameters Δ_0 and ϵ_{fun}.

For this version, $\Delta_0 = 0.2$ and $\epsilon_{fun} = 1e - 08$ are taken.

Table 5.3: Results of Version 1 with **lmlib**.

Level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	16	50	1.7676e-005	$[-0.73, 0.23, -0.13, -0.23]^T$
3	16	66	2.3789e-006	$[-0.74, 0.24, -0.22, -0.24]^T$
4	15	52	2.0259e-006	$[-0.75, 0.24, -0.23, -0.24]^T$
5	12	49	1.9427e-006	$[-0.76, 0.24, -0.23, -0.24]^T$

Table 5.4: Results of Version 1 with **trust**.

Level	iterations	func. eval.	func. value	u = [P9;P12]
1	46	132	2.5013e-009	$[-0.71, 0.22, -0.04, -0.22]^T$
2	27	90	3.2511e-005	$[-0.76, 0.22, -0.09, -0.22]^T$
3	18	73	1.5347e-005	$[-0.73, 0.23, -0.17, -0.23]^T$
4	15	52	1.1001e-005	$[-0.75, 0.23, -0.19, -0.23]^T$
5	17	61	5.5734e-006	$[-0.76, 0.24, -0.21, -0.24]^T$

Version 2

The interpolation set at Q_{i+1} is constructed using the final interpolation set at level i. The final interpolation set is the set when $x_{min,i}$ was found at level i.

$\Delta_0 = 0.2$ and $\epsilon_{fun} = 1e-08$ are taken.

Table 5.5: Results of Version 2 with **lmlib**.

Level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	13	34	2.0762e-005	$[-0.71, 0.23, -0.14, -0.24]^T$
3	15	44	3.5664e-005	$[-0.72, 0.24, -0.15, -0.24]^T$
4	6	23	3.9836e-005	$[-0.72, 0.23, -0.15, -0.24]^T$
5	20	70	1.4314e-005	$[-0.74, 0.24, -0.18, -0.24]^T$

Table 5.6: Results of Version 2 with **trust**.

Level	iterations	func. eval.	func. value	u = [P9;P12]
1	46	132	2.5013e-009	$[-0.71, 0.22, -0.04, -0.22]^T$
2	9	26	6.2624e-005	$[-0.71, 0.22, -0.04, -0.22]^T$
3	10	31	1.2785e-004	$[-0.71, 0.22, -0.04, -0.22]^T$
4	10	33	1.4535e-004	$[-0.72, 0.22, -0.05, -0.23]^T$
5	20	62	1.0692e-004	$[-0.73, 0.22, -0.08, -0.22]^T$

Version 3 : Model Choice

In this version, *selection* between the standard Taylor approximation model

$$\min_{\|s\|\leq\Delta} \quad \nabla Q_i(x)^T s + \frac{1}{2} s^T \nabla^2 Q_i(x) s$$

and the model

$$\min_{\|s\|<\Delta} \quad h_{i+1}(x_{min,i} + s) = Q_i(x_k) + (\nabla Q_{i+1}(x_{min,i}) - \nabla Q_i(x_{min,i}))(x_k - x_{min,i})$$

is done.

- At level $i = 1$:

  Taylor approximation model is used. Namely, DFO trust region subproblem

  $$\min_{\|s\|\leq\Delta} \quad \nabla Q_i(x)^T s + \frac{1}{2} s^T \nabla^2 Q_i(x) s$$

  is solved.

- At level $i + 1$, $i = 1, \ldots, r - 1$:

  - Interpolation set is constructed for Q_{i+1}.
  - If $\|\nabla Q_{i+1}\| \geq \kappa_Q \|\nabla Q_i\|$ and $\|\nabla Q_{i+1}\| > \epsilon_Q$ are satisfied, then new model is used and the subproblem

    $$\min_{\|s\|<\Delta} h_{i+1}(x_{min,i} + s)$$

    will be solved with **CONDOR**$^+$. Here,

    $$h_{i+1}(x_k) = Q_i(x_k) + (\nabla Q_{i+1}(x_{min,i}) - \nabla Q_i(x_{min,i}))(x_k - x_{min,i}) \,,$$

    where $x_k = x_{i+1,k} = x_{min,i} + s$ and $x_{min,i}$ is the minimum of level i.
  - Otherwise, Taylor approximation model is used again.

In addition to the initial parameters as in DFO based on NFPs, MDFO with model choice uses the initial constants κ_Q and ϵ_Q which are chosen between 0 and 1.

This version can also be observed in two ways as in Version 1 and Version 2.

Version 3a

The minimum point of the previous level is used to construct the interpolation set for Q_{i+1}.

$\Delta_0 = 0.2$, $\epsilon_{fun} = 1e-08$, $\kappa_Q = 0.01$ and $\epsilon_Q = 0.001$ are taken.

Table 5.7: Results of Version 3a with **lmlib**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	13	48	5.4985e-010	$[-0.75, 0.24, -0.24, -0.24]^T$
3	14	51	8.1957e-010	$[-0.75, 0.24, -0.24, -0.24]^T$
4	15	51	9.2230e-010	$[-0.75, 0.24, -0.24, -0.24]^T$
5	14	50	9.5445e-010	$[-0.75, 0.24, -0.24, -0.24]^T$

Table 5.8: Results of Version 3a with **trust**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	46	132	2.5013e-009	$[-0.71, 0.22, -0.04, -0.22]^T$
2	24	83	2.0302e-009	$[-0.74, 0.24, -0.24, -0.24]^T$
3	16	51	1.1741e-010	$[-0.74, 0.25, -0.25, -0.25]^T$
4	14	50	1.2672e-010	$[-0.74, 0.25, -0.25, -0.25]^T$
5	14	50	1.2954e-010	$[-0.74, 0.25, -0.25, -0.25]^T$

Version 3b

The last interpolation set of ith level is used for constructing the initial interpolation set Q_{i+1}.

$\Delta_0 = 0.2$ and $\epsilon_{fun} = 1e-08$ are taken.

Table 5.9: Results of Version 3b with $\kappa_Q = 0.01$, $\epsilon_Q = 0.001$ and **lmlib**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	21	49	4.9240e-009	$[-0.74, 0.25, -0.24, -0.24]^T$
3	5	21	8.1538e-009	$[-0.74, 0.25, -0.24, -0.24]^T$
4	4	19	9.3654e-009	$[-0.74, 0.25, -0.24, -0.24]^T$
5	4	19	9.7371e-009	$[-0.74, 0.25, -0.24, -0.24]^T$

Table 5.10: Results of Version 3b with $\kappa_Q = 0.1$, $\epsilon_Q = 0.0001$ and **lmlib**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	13	34	2.0762e-005	$[-0.71, 0.23, -0.14, -0.24]^T$
3	20	53	9.1769e-009	$[-0.75, 0.25, -0.25, -0.25]^T$
4	4	19	9.5299e-009	$[-0.75, 0.25, -0.25, -0.25]^T$
5	4	19	9.6493e-009	$[-0.75, 0.25, -0.25, -0.25]^T$

Table 5.11: Results of Version 3b with $\kappa_Q = 0.01$, $\epsilon_Q = 0.001$ and **trust**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	46	132	2.5013e-009	$[-0.71, 0.22, -0.04, -0.22]^T$
2	23	51	1.2296e-009	$[-0.74, 0.24, -0.24, -0.25]^T$
3	4	19	1.7976e-009	$[-0.74, 0.24, -0.24, -0.25]^T$
4	4	19	2.0130e-009	$[-0.74, 0.24, -0.24, -0.25]^T$
5	4	19	2.0818e-009	$[-0.74, 0.24, -0.24, -0.25]^T$

Version 4

Now, we will use $f(x)$ instead of $Q(x)$ in h model. In other words, the numerical results of the nonlinear model (4.3) will be investigated on the shape optimization problem.

- At level $i = 1$:

  Standard DFO trust-region subproblem

  $$\min_{\|s\| \leq \Delta} \quad \nabla Q_i(x)^T s + \frac{1}{2} s^T \nabla^2 Q_i(x) s$$

  is used.

- At level $i + 1$, $i = 1, \ldots, r - 1$:

  - Interpolation set is constructed for Q_{i+1}.
  - The model is set

    $$h_{i+1}(x_k) = f_i(x_k) + (\nabla Q_{i+1}(x_{min,i}) - \nabla Q_i(x_{min,i}))(x_k - x_{min,i}) \,,$$

    where $x_k = x_{i+1,k}$, $x_{min,i}$ is the minimum point which is got from the ith level and Q_i is the last model function when the minimum point of i was found.
  - The subproblem

    $$\min_{\|s\| < \Delta} \quad h_{i+1}(x_{min,i} + s)$$

    is solved with **CONDOR**$^+$.

Moreover, the criteria to choose Taylor model or (4.3) is the same as in Version 3.

Version 4a

To construct interpolation set for Q_{i+1}, the minimum point of level i is only used.

$\Delta_0 = 0.2$, $\epsilon_{fun} = 1e-08$, $\kappa_Q = 0.01$, $\epsilon_Q = 0.001$ are taken.

Table 5.12: Results of Version 4a with **lmlib**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	13	48	5.4985e-010	$[-0.75, 0.24, -0.24, -0.24]^T$
3	14	51	8.1957e-010	$[-0.75, 0.24, -0.24, -0.24]^T$
4	14	51	9.2230e-010	$[-0.75, 0.24, -0.24, -0.24]^T$
5	14	51	9.5445e-010	$[-0.75, 0.24, -0.24, -0.24]^T$

Table 5.13: Results of Version 4a with **trust**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	46	132	2.5013e-009	$[-0.71, 0.22, -0.04, -0.22]^T$
2	14	58	6.6147e-010	$[-0.74, 0.24, -0.24, -0.24]^T$
3	15	50	6.5556e-010	$[-0.75, 0.25, -0.25, -0.25]^T$
4	14	51	7.9843e-010	$[-0.75, 0.25, -0.25, -0.25]^T$
5	14	49	8.4429e-010	$[-0.75, 0.25, -0.25, -0.25]^T$

Version 4b

Using final interpolation set of level i, interpolation set for Q_{i+1} is constructed.

$\Delta_0 = 0.2$ and $\epsilon_{fun} = 1e-08$ are taken.

Table 5.14: Results of Version 4b with $\kappa_Q = 0.01$, $\epsilon_Q = 0.001$ and **lmlib**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	21	49	4.9240e-009	$[-0.74, 0.25, -0.24, -0.24]^T$
3	5	21	8.1538e-009	$[-0.74, 0.25, -0.24, -0.24]^T$
4	4	19	9.3654e-009	$[-0.74, 0.25, -0.24, -0.24]^T$
5	4	19	9.7371e-009	$[-0.74, 0.25, -0.24, -0.24]^T$

Table 5.15: Results of Version 4b with $\kappa_Q = 0.1$, $\epsilon_Q = 0.0001$ and **lmlib**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	32	103	5.0677e-008	$[-0.71, 0.23, -0.13, -0.23]^T$
2	9	28	2.1563e-005	$[-0.71, 0.23, -0.14, -0.23]^T$
3	27	80	5.0530e-010	$[-0.74, 0.25, -0.24, -0.25]^T$
4	4	19	6.1443e-010	$[-0.74, 0.25, -0.24, -0.25]^T$
5	4	19	6.4886e-010	$[-0.74, 0.25, -0.24, -0.25]^T$

Table 5.16: Results of Version 4b with $\kappa_Q = 0.01$, $\epsilon_Q = 0.001$ and **trust**.

level	iterations	func. eval.	func. value	u = [P9;P12]
1	46	132	2.5013e-009	$[-0.71, 0.22, -0.04, -0.22]^T$
2	25	51	2.9001e-009	$[-0.75, 0.24, -0.24, -0.25]^T$
3	4	20	3.8769e-009	$[-0.75, 0.24, -0.24, -0.25]^T$
4	4	19	4.2775e-009	$[-0.75, 0.24, -0.24, -0.25]^T$
5	4	19	4.4013e-009	$[-0.75, 0.24, -0.24, -0.25]^T$

5.1 Analysis of Numerical Results

Not only to solve Taylor model but also at all sub-implementation parts of DFO, **trust**; eigenvalue decomposition based on the secular equation or **lmlib**; the Levenberg-Marquardt algorithm with the More & Sorensen technique is used. It is observed that at coarse grids, **trust** requires more iterations and function evaluations than **lmlib**. But later, on finer grids both produce the same number of iterations and function evaluations.

At all results, initial trust region radius can take any other value (recommended less than 1). This value alters the results. Similar to this, the choice between the lower level model and Taylor model critically depends on parameters like κ_Q and ϵ_Q. For instance, Table 5.9 and Table 5.14 seems to give the same results. These results can be interpreted as no new model and no choice condition. When the parameters $\kappa_Q = 0.1$ and $\epsilon_Q = 0.0001$ was taken, the results are as in Table 5.10 and Table 5.15. Therefore, our termination criteria and the initial parameters gain the importance to determine which approach will be nice.

Since the point $u = [P9; P12]$ (the control variable) that we found after any optimization procedure is not a minimization value of the problem, it is not used to compare the results. Therefore, we interpret the results with respect to function evaluations without looking minimum point u.

The construction of interpolation set at level $i + 1$ around the minimum point of the level i causes the more function evaluations than the construction of interpolation set using the last interpolation set of previous level.

When the model is chosen at any level $i > 1$ in a current iteration k and the interpolation set for the model Q_{i+1} is constructed using the last interpolation set at level i, we get the best results in terms of computational cost and time (i.e. Version 3b with trust).

The results of Version 4b are not so bad but the computation time is long when the model (4.3) works in the programming.

CHAPTER 6

CONCLUSIONS

In this thesis, we developed a new derivative free algorithm on a shape optimization problem which is a control problem. In Chapter 4 we described how our new method called Multilevel Derivative Free Optimization (MDFO) works.

Our shape optimization problem was discretized at most six discretization levels. Due to computational complexity and much time at higher levels, we analyzed the results up to fifth level. For every level, we had a different nonlinear function. Therefore, for this multilevel structure, we used the DFO to minimize functions.

We used two implementations of DFOs. One is CONDOR, and the other one is DFO based on NFPs. Both of them was a MATLAB programming package. Our main task was to modify DFO based on NFPs to form MDFO. Besides, in view of computational easiness, Moré and Sorensen subroutine of CONDOR was very useful to solve the subproblem of our new model.

The construction of interpolation sets affects the results very much in MDFO. In fact, the construction of the interpolation set using the last interpolation set of previous level gave less function evaluations than the other results. Moreover, another remarkable point is to define cheap lower level model. In summary, chosing criteria between Taylor model and cheap lower level model and the construction of interpolation sets from the previous level result in the effective function evaluations.

In the future, it can be observed how MDFO works with the other problems like fluid dynamics or the problems with much variables. Furthermore, The criteria inside the algorithm can be improved. Instead of CONDOR, the other DFO programming packages can be examined.

REFERENCES

[1] N. M. Alexandrov, J. E. Dennis, R. M. Lewis, V. Torczon. *A trust region framework for managing the use of approximation model*. Structural Optimization, 15(1):16-23, 1998.

[2] F. V. Berghen. *CONDOR: a constrained, non-linear, derivative-free parallel optimizer for continuous, high computing load, noisy objective functions*. Université Libre de Bruxelles, Belgium, 2004. PhD Thesis.

[3] F. V. Berghen, H. Bersini. *CONDOR, an new Parallel, Constrained extension of Powell's UOBYQA algorithm. Experimental results and comparison with the DFO algorithm*. Technical Report 11, Université Libre de Bruxelles, Iridia, 2004.

[4] F. V. Berghen, 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, 181:157-175, 2005.

[5] R. P. Brent. *Algorithms for Minimization Without Derivatives*. Prentice-Hall, Engelwood Cliff, USA, 1973.

[6] W. L. Briggs, V. E. Henson, S. F. McCormick. *A Multigrid Tutorial*, 2nd edition. SIAM, Philadelphia, USA, 2000.

[7] I. G. Campey, D. G. Nickols. *Simplex Minimization*, Program Specification. Imperial Chemical Industries Ltd, UK, 1961.

[8] B. Colson, Ph. L. Toint. *Exploiting Band Structure in Unconstrained Optimization Without Derivatives* . Optimization and Engineering, 2(4):399–412, December 2001.

[9] A. R. Conn, K. Scheinberg , Ph. L. Toint. *Recent progress in unconstrained nonlinear optimization without derivatives*, Mathematical Programming, 79:397–414, 1997.

[10] A. R. Conn, K. Scheinberg, Ph. L. Toint. *On the convergence of derivative-free methods for unconstrained optimization*. In Approximation Theory and Optimization: Tribute to M.J.D. Powell, editors: A. Iserles and M. Buhmann, pp.83-108. Cambridge University Press, Cambridge, UK, 1997.

[11] A. R. Conn, K. Scheinberg, L. N. Vicente. *Geometry of interpolation sets in derivative free optimization*. Mathematical Programming, Series B, 111(1-2), 2008.

[12] A. R. Conn, K. Scheinberg, L. N. Vicente. *Geometry of sample sets in derivative free optimization: polynomial regression and underdetermined interpolation* . IMA journal of numerical analysis, vol.28, pp.721-748, 2008.

[13] A. R. Conn, K. Sheinberg, L. N. Vicente. *Introduction to Derivative-Free Optimization*. SIAM Series on Optimization, 2009.

[14] R. Cosentino, Z. Alsalihi, R. Van Den Braembussche. *Expert System for Radial Impeller Optimisation*. In Fourth European Conference on Turbomachinery, ATI-CST-039/01, Florence, Italy, 2001.

[15] J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM Society for Industrial and Applied Mathematics, Berkeley, California, 1997.

[16] J. E. Dennis Jr., R. B. Schnabel. *Numerical Methods for unconstrained Optimization and nonlinear Equations*. Prentice-Hall, Englewood Cliffs, USA, 1983.

[17] E. Fan. *Global Optimization Of Lennard-Jones Atomic Clusters* , McMaster University, February 2002. Master Thesis.

[18] M. Fisher. *Minimization algorithms for variational data assimilation*. In Recent Developments in Numerical Methods for Atmospheric Modelling, pp.364-385, ECMWF, 1998.

[19] R. Fletcher. *Practical Methods of optimization*. A Wiley-Interscience publication, Great Britain, 1987.

[20] D. M. Gay. *Computing optimal locally constrained steps*. SIAM Journal Sci. Stat. Comp., 2(2):186 - 197, 1981.

[21] S. Gratton, A. Sartenaer, Ph. L. Toint. *Numerical Experience with a recursive trust-region method for multilevel nonlinear optimization*. Technical Report No.06/01, Department of Mathematics, University of Namur, Namur, Belgium, 2006.

[22] S. Gratton, A. Sartenaer, Ph. L. Toint. *Recursive Trust-Region Methods for Multiscale Nonlinear Optimization*. SIAM Journal on Optimization, 19:414-444, 2008.

[23] N. I. M. Gould, S. Lucidi, M. Roma, Ph. L. Toint. *Solving the trust-region subproblem using the Lanczos method*. SIAM J. Optim., 9:504-525, 1999.

[24] B. Karasözen. *Survey of Trust-Region Derivative Free Optimization Methods*. AIMS Journals, 3(2):321–334, 2007.

[25] D. Kincaid, W. Cheney. *Numerical Analysis*. Mathematics of Scientific Computing, Brooks/Cole - Thomson Learning, The University of Texas at Austin, USA, 2002.

[26] M. Marazzi, J. Nocedal. *Wedge trust region methods for derivative free optimization*. Mathematical Programming, Series A, 91:289–305, 2002.

[27] J. L. Morales, G. Fasano, J. Nocedal. *On the Geometry Phase in Model-Based Algorithms for Derivative-Free Optimization*. Optimization Methods and Software, 24:145–154, 2009.

[28] J. J. Moré, D. C. Sorenson. *On the use of directions of negative curvature in a modified Newton Method*. Mathematical Programming, 16(1):1 - 20, 1979.

[29] E. D. Dolan and J. J. Moré.*Benchmarking optimization software with performance profiles*. Mathematical Programming, 91: 201 - 213, 2002.

[30] J. J. Moré, S. M. Wild. *Benchmarking Derivative-Free Optimization Algorithm*. Argonne National Laboratory, Mathematics and Computer Science Division, Argonne, Illinois, December 2007. Preprint ANL/MCS-P1471-1207.

[31] S. G. Nash, M. Lewis. *Model problems for the multigrid optimization of systems governed by differential equations*. SIAM Journal on Optimization,(to appear), 2005.

[32] J. A. Nelder, R. Mead. *A Simplex method for function minimization*. Computer Journal, 7:308 - 313, 1965.

[33] R. Oeuvray. *Trust-Region Methods Based On Radial Basis Functions with Application to Biomedical Imaging* . Operations Research Group (ROSO), Institute of Mathematics, Swiss Federal Insitute of Technology, Luisanns, Switzerland, March 2005. PhD Thesis.

[34] R. Oeuvray, M. Bierlaire. *BOOSTERS: a derivative-free algorithm based on radial basis functions*. International Journal of Modelling and Simulation, 2008.

[35] R. Oeuvray, M. Bierlaire. *A New Derivative-Free Algorithm For The Medical Image Registration Problem*. In Proc. 4th IASTED International Conference on Modelling, Simulation, and Optimization, 2004.

[36] Ö. Uğur, M. Schäfer, B. Karasözen, Y. Uludağ, K. Yapıcı. *Numerical Method for Optimizing Stirrer Configurations*. Computers & Chemical Engineering, 30:183–190, 2005.

[37] M. J. D. Powell. *An Efficient Method for Finding the Minimum of A Function of Several Variables Without Calculating Derivatives* . Computer Journal, 17:155 - 162, 1964.

[38] M. J. D. Powell. *A method for minimizing a sum of squares of nonlinear functions without calculating derivatives*. Computer Journal, 7:303 - 307, 1965.

[39] M. J. D. Powell. *A new algorithm for unconstrained optimization*. In Nonlinear Programming, editors:J. B. Rosen, O. L. Mangasarian, K. Ritter, pp.31 - 65. Academic Press, London, 1970.

[40] M. J. D. Powell. *Unconstrained Minimization Algorithms Without Computation of Derivatives*. Bollettino della Unione Matematica Italiana, 9:60 - 69, 1974.

[41] M. J. D. Powell. *A view of unconstrained minimization algorithms that do not require derivatives* . ACM Transactions on Mathematical Software, 1(2):97 - 107, 1975.

[42] M. J. D. Powell. *A direct search optimization method that models the objective and constraint functions by linar interpolation*. In Advances in Optimization and Numerical Analysis, Proceedings of the sixth Workshop on Optimization and Numerical Analysis, Oaxaca, Mexico, pp.51-67. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994.

[43] M. J. D. Powell. *A direct search optimization method that models the objective by quadratic interpolation*. Presentation at the 5th Stockholm Optimization Days, 1994.

[44] M. J. D. Powell. *UOBYQA : Unconstrained Optimization By Quadratic Approximation*. Technical Report No.DAMTP2000/14. Department of Applied Mathematics and Theoretical Physics, University of Cambridge, England, 2000.

[45] M. J. D. Powell. *On the Lagrange functions of quadratic models that are defined by interpolation*. Optimization Methods Software, 16:555 - 582, 2001.

[46] M. J. D. Powell. *Developments of NEWUOA for minimization without derivatives*. IMA Journal of Numerical Analysis, 28(4):649–664, February 2008.

[47] R. T. Rockefeller. *Convex analysis*. Princeton Univ. Press, Princeton, New Jersey, 1970.

[48] K. Scheinberg. *Derivative free optimization method*. Technical Report CS 4/6-TE3, SFW ENG 4/6-TE3, IBM Watson Research Center, 2000.

[49] K. Scheinberg. *Derivative free optimization method*. Department Computing and Software, McMaster University, 2000.

[50] W. Spendley, G. R. Hext, F. R. Himsworth. *Sequential application of simplex designs in optimization and evolutionary operation*. Technometrics, 4:441 - 461, 1962.

[51] T. Steihaug. *The conjugate gradient method and trust regions in large scale optimization*. SIAM J. Numer. Anal., 20:626-637, 1983.

[52] T. Terlaky. *Algorithms for Continuous Optimization DFO-Trust Region Interpolation Algorithm*. Computer and Software, McMaster University, Hamilton, Canada, January 2004. 4TE3/6TE3.

[53] Ph. L. Toint, A. R. Conn, N. I. M. Gould. *Trust-Region Methods*. SIAM Society for Industrial and Applied Mathematics, Englewood Cliffs, New Jersey, mps-siam series on optimization edition, 2000. Chapter 6: Global Convergence of the Basic Algorithm, pp.115 - 168.

[54] Ph. L. Toint, A. R. Conn, N. I. M. Gould. *Trust-Region Methods*. Number 01 in MPS-SIAM Series on Optimization, SIAM, Philadelphia, USA, 2000.

[55] Ph. L. Toint, F. Bastin, V. Malmedy, M. Mouffe, D. Tomanos. *A Retrospective Trust-Region Method for Unconstrained Optimization*. Technical Report 07/08, Department of Mathematics, University of Namur, Namur, Belgium, October 2007.

[56] Ph. L. Toint, D. Tomanos, M. Weber Mendonça. *A multilevel algorithm for solving the trust-region subproblem*, 2008. To appear in Optimization Methods and Software.

[57] Ph. L. Toint, D. Tomanos, M. Weber Mendonça. *A multilevel algorithm for solving the trust-region subproblem*. Optimization Methods and Software, 24(2):299 - 311, April 2009.

[58] Ph. L. Toint, K. Scheinberg. *Self-Correcting Geometry In Model-Based Algorithms For Derivative-Free Unconstrained Optimization*. Technical Report 09/06, Department of Mathematics, University of Namur, Namur, Belgium, February, 2009.

[59] V. Torczon. *On the convergence of multidirectional search algorithm* . SIAM Journal on Optimization, 1(1):123 - 145, 1991.

[60] A. R. Conn, K. Scheinberg, L. N. Vicente. *Global Convergence of General Derivative-Free Trust-Region Algorithms To First and Second Order Critical Points*. SIAM Journal on Optimization, 2006.

[61] S. M. Wild. *MNH: A Derivative-Free Optimization Algorithm Using Minimal Norm Hessians*. In Tenth Copper Mountain Conference on Iterative Methods, Cornell University, School of Operations Research and Information Engineering, April 2008. Available at http://grandmaster.colorado.edu/ copper/2008/SCWinners/Wild.pdf

[62] S. M. Wild, R. G. Regis, C. A. Shoemaker. *ORBIT: Optimization By Radial Basis Function Interpolation In Trust-Regions*. SIAM Journal on Scientific Computing, 30:3197-3219, 2008.

[63] D. Winfield. *Function and functional optimization by interpolation in data tables*. Harvard University, Cambridge, USA, 1969. PhD Thesis.

[64] D. Winfield. *Function minimization by interpolation in a data table*. Journal of the Institute of Mathematics and its Applications, 12:339-347, 1973.

[65] M. H. Wright. *Direct search methods: once scorned, now respectable*. In proceedings of the 1995 Dundee Biennal Conference in Numerical Analysis, editors:D. F. Griffiths, G. A. Watson. Addison Wesley Longman, UK, 1996.