Titre: Title:	A Stochastic Levenberg-Marquardt Method for Nonsmooth Regularized Inverse Problems
Auteur: Author:	Valentin Sylvain Bernard Dijon
Date:	2024
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Dijon, V. S. B. (2024). A Stochastic Levenberg-Marquardt Method for Nonsmooth Regularized Inverse Problems [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/61111/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/61111/
Directeurs de recherche: Advisors:	Youssef Diouane, & Dominique Orban
Programme: Program:	Maîtrise recherche en mathématiques appliquées

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

A Stochastic Levenberg-Marquardt Method for Nonsmooth Regularized Inverse Problems

VALENTIN SYLVAIN BERNARD DIJON
Département de mathématiques et de génie industriel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Mathématiques appliquées

Novembre 2024

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

A Stochastic Levenberg-Marquardt Method for Nonsmooth Regularized Inverse Problems

présenté par **Valentin Sylvain Bernard DIJON**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Charles AUDET, président
Youssef DIOUANE, membre et directeur de recherche
Dominique ORBAN, membre et codirecteur de recherche
Fabian BASTIN, membre

DEDICATION

To my family, my girlfriend and my friends who always supported me,
I love you.

ACKNOWLEDGEMENTS

In these few lines, I would like to thank all the people who made this master's thesis possible and who helped me get through the various challenges I faced over the last 2 years.

First of all, I give my thanks to the jury members Mr. Charles AUDET, Mr. Fabian BASTIN, Mr. Youssef DIOUANE and Mr. Dominique ORBAN for their appreciation of my work.

Then, I would like to thank my supervisors Mr. Youssef DIOUANE and Mr. Dominique ORBAN, who supported me and believed in my abilities throughout this project.

I would also like to warmly thank my family, who, even on the other side of the Atlantic, have never stopped believing in me and have always supported and encouraged me in what I wanted to do.

Finally, I would like to thank all my friends, both those who stayed in France and those I was lucky enough to meet in Montreal who have been with me through good and bad times.

RÉSUMÉ

On conçoit une variante stochastique avec régularisation non lisse de l'algorithme de Levenberg-Marquardt utilisé pour résoudre des problèmes de moindres carrés non linéaires. Notre algorithme est stochastique puisqu'il permet des évaluations inexactes du résidu moindres carrés et de sa jacobienne par un échantillonnage de ces derniers.

La dimension non lisse du problème empêche d'utiliser des métriques de stationnarité usuelles en optimisation car celles-ci reposent sur le gradient. Par ailleurs, l'aspect stochastique lié aux estimations basées sur un échantillonnage impacte la manière de sélectionner le prochain itéré ainsi que la convergence de l'algorithme. Nous avons conçu une preuve de convergence de notre variante de l'algorithme de Levenberg-Marquardt accompagnée d'une étude de complexité. Nous avons testé notre méthode avec une implémentation dans le langage de programmation Julia sur des problèmes de grande dimension provenant de l'intelligence artificielle et de la vision par ordinateur.

Les résultats de l'analyse théorique montrent que la borne de complexité de cette variante est analogue à d'autres déjà existantes, aussi bien lisses que non lisses. Les tests numériques révèlent que l'échantillonnage permet d'accélérer la vitesse de convergence en réduisant le coût numérique de l'optimisation pour certains problèmes. Ces tests montrent également que la régularisation non lisse permet d'obtenir une solution plus creuse et donc moins volumineuse à stocker.

ABSTRACT

We designed a stochastic and nonsmooth variant of the Levenberg-Marquardt algorithm used to solve nonlinear least squares. With large-scale problems in mind, our algorithm is stochastic in the sense that it allows inexact evaluations of both the least-squares residual and its Jacobian by sampling them.

The nonsmooth dimension of the problem prevents the use of common stationarity metrics in optimization because of the inability to access the gradient. Furthermore, the stochastic estimates based on sampling have an impact on how to select the next iterate and on the convergence of the algorithm. We designed a convergence analysis for the Levenberg-Marquardt algorithm, along with a complexity study. We tested our method with an implementation in Julia on high-dimensional problems that arise from artificial intelligence and computer vision.

The results of the theoretical analysis show that the complexity bound of this variant is similar to other methods that already exist, both smooth and nonsmooth. Numerical tests illustrate that sampling accelerates the speed of convergence by reducing the numerical cost of optimization for some problems. These tests also show that nonsmooth regularization enables the return of a sparser solution that is cheaper to store.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

$f \in \mathcal{C}^k$	$f : \mathbb{R}^n \to \mathbb{R}$ is of class $\mathcal{C}^k$, i.e., k times differentiable and k^{th} derivative is continuous on all the definition domain of f
$\mathbb{R}_+$	All the nonnegative reals, i.e., $\mathbb{R}_+ = [0, +\infty)$
$\|\cdot\|$	Without any precision, the Euclidean norm for vectors and spectral norm for matrices
I_n	Identity matrix of size n
r	Residual function such that $r : \mathbb{R}^n \to \mathbb{R}^m$
J	Jacobian of r where $J \in \mathbb{R}^{m \times n}$
A^T	Transposition of matrix A
$\text{rank}(A)$	Rank of matrix A
$\det(A)$	Determinant of matrix A
$\text{cond}(A)$	Conditioning number of matrix A
$\|S\|$	Cardinality of a set S
LM	Levenberg-Marquardt
TR	Trust Region
SVM	Support Vector Machine
MNIST	Dataset of numbers for image recognition by neural networks
IJCNN1	Dataset for neural network competition

CHAPTER 1 INTRODUCTION

We introduce two kinds of large-scale problems, arising from data science, artificial intelligence, and computer vision, that motivate us to devise a new LM variant. These problems result in challenges that necessitate the development of state-of-the-art methods to address them.

1.1 Context

In this section, we describe two well-known families of large-scale problems that share a common structure.

1.1.1 Nonlinear SVM

Figure 1.1 MNIST dataset © Suvanjanprasai CC BY-SA 4.0.

In the field of artificial intelligence, Support Vector Machines (SVMs) are a type of learning algorithm that has been the subject of a great deal of research due to their performance in object recognition and their statistical properties [53]. SVMs can be used to recognize handwritten numbers, such as those in the MNIST dataset [46], with a high degree of confidence.

After training the SVM, we evaluate its predictive ability on a set of unknown data, called the test set. The aim, therefore, is to maximize the number of correct predictions (or minimize the number of errors) on this test set in order to obtain the most generalizable model possible.

The binary classification problem can be formulated as follows:

$$\min_{x \in \mathbb{R}^n} \tfrac{1}{2} \|\mathbf{1} - \tanh\left(b \odot A^\top x\right)\|^2, \quad \text{where } \mathbf{1} = [1, 1, \ldots, 1]^\top, \tag{1.1}$$

where $A \in \mathbb{R}^{n \times m}$ is the matrix of embedded images that represent two different digits and b is a vector of labels where $b \in \{-1, 1\}^m$. Note that b is composed of 1 and -1 since we exclude all digits except two from the dataset to be in a binary classification context. The classification then determines whether a digit is in one group or the other. The operator $\odot$ is the pointwise product, i.e., taking two vectors $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$, the pointwise product of x and y is $x \odot y = [x_1 y_1, x_2 y_2, \ldots, x_n y_n]^\top$.

However, the challenge in training these neural networks is that they have to process a large amount of data, and the calculations can prove quite costly. For MNIST, after embedding, the dimensions of the data matrix are 12665×784. For another problem called IJCNN1 (taken from the LIBSVM library [26]) whose data was used for competition between binary classifiers, the dimensions of the data matrix are 49990×22.

The considerable amount of data thus poses a real challenge to the effectiveness of algorithms that seek to train on such datasets. Moreover, the solutions returned can also be substantial. For storage efficiency, we aim to find sparse solutions, i.e., with many zeros.

1.1.2 Bundle Adjustment Problems

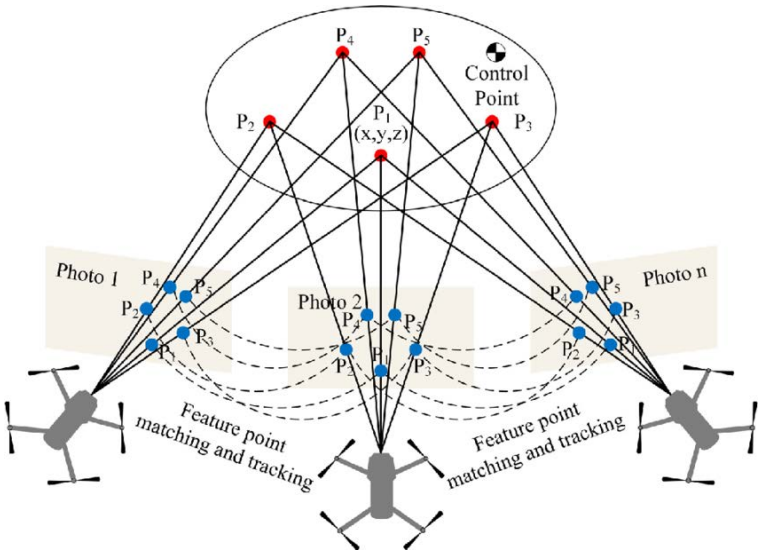


Figure 1.2 Concept behind Bundle Adjustment problems © Dongliang Huang, CC BY 4.0.

Bundle Adjustment problems consist of reconstructing structures in computer vision from

photographs. The goal is to generate 3D points from a collection of 2D images. We want to minimize the error between the observations and the data. This process is illustrated by Fig. 1.2 from [30].

The problem can be formulated as follows:

$$\min_{X,C} \sum_{i=1}^{N_{points}} \sum_{j=1}^{N_{cams}} v_{i,j} \|P(X_i, C_j) - b_{i,j}^{obs}\|^2, \tag{1.2}$$

where N_{points} and N_{cams} are the number of points and the number of cameras respectively and

- $X_i = [x_i, y_i, z_i]^T \in \mathbb{R}^3$ is the 3D coordinates of point $i \in \{1, \ldots, N_{points}\}$,
- $C_j \in \mathbb{R}^9$ represents the 9 parameters of camera $j \in \{1, \ldots, N_{cams}\}$,
- $v_{i,j} \in \{0, 1\}$ indicates if camera j observes point i,
- $P(X_i, C_j)$ is the projection function of point X_i on a camera C_j,
- $b_{i,j}^{obs}$ is 2D real observation for point i and camera j.

With solutions of these problems, it is possible to reconstruct an entire city or famous monuments in computer vision [1]. However, they are extremely large, and generating the full Jacobian may cause out-of-memory errors on a standard computer. Fortunately, it turns out that these problems have very sparse structures, that need to be exploited by optimization strategies.

1.2 Research question

Regarding those two sorts of problems, we would like to develop an algorithm able to return sparse solutions (for storing advantages). We aim for it to handle large amounts of data and compute efficiently. Besides, (1.1) and (1.2) share a common problem structure called nonlinear least squares as they aim for minimizing a finite squared sum. An algorithm devised by Levenberg [47] and Marquardt [50] exploits this structure efficiently and is today the main reference for methods solving nonlinear least squares.

As we consider large-scale problems, returned solutions may be large as well. That is why, we could draw advantages from computing sparse solutions as their storage is more economical. A strategy used in numerical optimization, called regularization, can allow algorithms to select a solution with specific properties like sparsity. However, many regularizers that return sparse solutions are nonsmooth (i.e., not differentiable) [3].

In the same spirit, since the amount of data may be too large for the entire residual to fit in memory, we will treat only a sample. This involves a stochastic framework, as the method uses estimates instead of exact quantities [9].

The main framework here is to merge nonsmooth regularization and sample schemes. But these two strategies altogether impact both the stopping criterion and the quality of the solution returned by the algorithm. Hence, to what extent can we include a nonsmooth regularization and a stochastic framework to the Levenberg-Marquardt algorithm to solve more efficiently nonlinear least squares and return specific solutions?

The thesis is organized as follows. In Chapter 2 we present the necessary mathematical background and recent works in nonsmooth and stochastic algorithms. We develop in Chapter 3 the theory of our LM variant of the algorithm and conduct numerical tests. In Chapter 4, we summarize our theoretical and numerical results, along with the limits and perspectives of this approach.

CHAPTER 2 LITERATURE REVIEW

In this literature review, we present the key notions needed to understand the problem framework. Sections 2.1 and 2.2 present the basic background to understand what mathematical optimization and nonlinear least squares are where Section 2.3 deals with the Levenberg-Marquardt (LM) algorithm which solves nonlinear least-squares problems. Eventually, Sections 2.4 and 2.5 introduce notions covering specificity added to the LM algorithm with this research framework.

2.1 Mathematical optimization

Mathematical optimization or mathematical programming is a field of mathematics that arises from several mathematical aspects like linear algebra, analysis, and programming. We provide here the main concepts of mathematical optimization stated in [54].

2.1.1 Basic mathematical background

In mathematics, an unconstrained optimization problem can be formulated as follows:

$$\min_{x \in \mathbb{R}^n} f(x), \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is called the *objective function* to minimize. We then search for an argument $x \in \mathbb{R}^n$ satisfying 2.1 written as

$$x \in \operatorname*{argmin}_{x \in \mathbb{R}^n} f(x). \tag{2.2}$$

Note first here the difference between min representing the minimum *real value* attained by f, and argmin which is the *set* of all x such that $f(x)$ attains its minimum value. A constrained problem can be formulated as

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} f(x), \\ &\text{s.t.} \quad x \in C, \end{aligned} \tag{2.3}$$

where C is a nonempty set. The way of managing the constraints is not central to our subject, as unconstrained optimization is mainly treated here. Adding constraints to an optimization problem represents more realistic scenarios but complicates their resolution.

Minimization is implicit in optimization problems, as it is a standard convention in the

literature. For maximization problems, there is the following proposition:

Proposition 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$. Then*

$$\max_{x\in\mathbb{R}^n} f(x) = -\min_{x\in\mathbb{R}^n} -f(x)$$
$$\operatorname*{argmax}_{x\in\mathbb{R}^n} f(x) = \operatorname*{argmin}_{x\in\mathbb{R}^n} -f(x).$$

There exist different types of minima with the first distinction of *global* and *local* minima.

Definition 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$. Then x^* is a global minimizer of f if $f(x^*) \leq f(x)$ for all $x \in \mathbb{R}^n$.*

Definition 2. *Let $f : \mathbb{R}^n \to \mathbb{R}$. Then x^* is a local minimizer of f if there is a neighborhood $\mathcal{N}$ of x^* such that $f(x^*) \leq f(x)$ for all $x \in \mathcal{N}$.*

Another very useful tool in optimization is Taylor's theorem.

Theorem 1. *[55, Theorem 2.1] Suppose that $f \in \mathcal{C}^1$ and that $p \in \mathbb{R}^n$. Then we have that*

$$\forall x \in \mathbb{R}^n, \quad f(x+p) = f(x) + p^\top \nabla f(x+tp), \tag{2.4}$$

for some $t \in (0,1)$. If $f \in \mathcal{C}^2$, we have that

$$\forall x \in \mathbb{R}^n, \quad f(x+p) = f(x) + p^\top \nabla f(x+tp) + \tfrac{1}{2} p^\top \nabla^2 f(x+tp) p, \tag{2.5}$$

for some $t \in (0,1)$.

Here, $\nabla f(x)$ and $\nabla^2 f(x)$ represent the gradient and the Hessian of f at x respectively. They are defined as

$$\nabla f(x) \coloneqq \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix} \quad \text{and} \quad \nabla^2 f(x) \coloneqq \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}. \tag{2.6}$$

To characterize if x is a minimum, there exists a necessary first order criterion.

Theorem 2. *[54, Theorem 2.2]* ***First Order Necessary conditions:*** *If x^* is a local minimizer and f is continuously differentiable on a neighborhood of x^*, then $\nabla f(x^*) = 0$.*

This criterion can then be used by an algorithm as a stopping condition.

2.1.2 Algorithmic notions

An optimization algorithm is a method that generates iterates x_j at each iteration j to be closer and closer to a local minimum. Its goal, as described in [55], is twofold. It has to be

1. Efficient: The algorithm has to compute a solution with the least operations possible and as fast as possible.
2. Accurate: We require the returned solution to be sufficiently close to the real solution according to a given threshold.

Efficiency is more inherent to the way the algorithm is designed, while accuracy can be measured relative to the optimality criterion stated earlier. Consider the notion of global convergence.

Definition 3. *Let a sequence* $\{x_j\}$ *and* $f : \mathbb{R}^n \to \mathbb{R}$ *such that* $f \in \mathcal{C}^1$*. Then* $\{x_j\}$ *converges globally if*

$$\lim_{j \to +\infty} \|\nabla f(x_j)\| = 0, \tag{2.7}$$

for any $x_0 \in \mathbb{R}^n$*.*

Not all algorithms converge globally or at the same speed. The convergence rate can be stated as follows:

Definition 4. *The sequence* $\{x_j\}$ *converging to* x^* *is said to converge at rate* r *if there exist* $r > 0$ *and* $c \geq 0$*, such that for all* $j \in \{0, 1, \dots\}$

$$\frac{\|x_{j+1} - x^*\|}{\|x_j - x^*\|^r} \leq c. \tag{2.8}$$

The convergence rate is said linear if $r = 1$ *and* $c \in (0, 1)$*, super linear if* $r \in (1, 2)$*, and quadratic if* $r = 2$ *and* $\|x_0 - x^*\| \leq \frac{1}{c}$*.*

A higher convergence rate results in faster convergence to a local minimum. Note finally the analysis notion of Lipschitz-continuity.

Definition 5. *A function* $f : \mathbb{R}^n \to \mathbb{R}^m$ *is said Lipschitz-continuous with Lipschitz constant* $L > 0$ *if for all* $x, y \in \mathbb{R}^n$*,*

$$\|f(x) - f(y)\| \leq L\|x - y\|. \tag{2.9}$$

2.2 Nonlinear least squares

2.2.1 Linear least-squares problems

As presented in Chapter 1, we minimize problems with a very specific structure, and it is crucial to present it in this section to familiarize the reader with it.

Before introducing nonlinear least squares, it is essential to first explain linear least squares, as their understanding is necessary for grasping their nonlinear counterparts. A least-squares problem is formulated as

$$\min_{x \in \mathbb{R}^n} f(x) \coloneqq \tfrac{1}{2}\|Ax - b\|^2, \tag{2.10}$$

where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. For this structure, we have

$$\nabla f(x) = A^\top (Ax - b) \quad \text{and} \quad \nabla^2 f(x) = A^\top A. \tag{2.11}$$

The necessary first order optimality condition can be written as

$$A^\top A x = A^\top b, \tag{2.12}$$

and are known as the *normal equations* of the main problem. In practice, solving explicitly the normal equations is not recommended due to matrix conditioning issues. Indeed, the conditioning number of an invertible matrix $A \in \mathbb{R}^{n \times n}$ is defined by

$$\text{cond}(A) \coloneqq \|A^{-1}\|\|A\|, \tag{2.13}$$

and, $\text{cond}(A^\top A) = \text{cond}\,(A)^2$ and may result in numerical instability. Methods to solve linear least-squares problems are presented in Section 2.3.2.

2.2.2 Nonlinear least-squares problems

Consider now the nonlinear least-squares problem formulated as follows:

$$\min_{x \in \mathbb{R}^n} f(x) \coloneqq \tfrac{1}{2}\|r(x)\|^2, \tag{2.14}$$

where $r : \mathbb{R}^n \to \mathbb{R}^m$ such that $r(x) = [r_1(x), \ldots, r_m(x)]$. For all $i \in \{1, \ldots, m\}$, $r_i(x)$ are nonlinear. Like for linear least squares, we have explicit expressions of the gradient and the

Hessian:

$$\nabla f(x) = J(x)^\top r(x) \quad \text{and} \quad \nabla^2 f(x) = J(x)^\top J(x) + \sum_{i=1}^{m} \nabla^2 r_i(x) r_i(x), \tag{2.15}$$

where $J(x)$ is the Jacobian of $r(x)$, also denoted $J(x) = \nabla r(x)$. To solve (2.14), we can use an iterative approach, called the Gauss-Newton method, derived from the Newton method which is used to find the roots of a function. The goal of the Gauss-Newton method is to find iteratively at each computed point x_j a step s_j^{GN} which is the solution of the following linear system:

$$\nabla^2 f(x_j) s_j^{GN} = -\nabla f(x_j). \tag{2.16}$$

The Gauss-Newton step is unique as long as $\nabla^2 f(x_j)$ is invertible and, if so, we have that $s_j^{GN} = -\nabla^2 f(x_j)^{-1} \nabla f(x_j)$.

According to (2.15), the computation of $\nabla^2 f(x_j)$ requires all the $\nabla^2 r_i(x_j)$, which is computationally expensive. Thus, a common assumption is to consider for all $i \in \{1, \ldots, n\}$ that $r_i(x_j)$ is very small or nearly affine to neglect the right-hand term of the Hessian in (2.15). Consequently, the Hessian of the objective function f at x_j can be approximated by $J^\top(x_j) J(x_j)$. Thus, (2.16) comes down to solve the linear system

$$J(x_j)^\top J(x_j) s_j^{GN} = -J(x_j)^\top r(x_j) \iff J(x_j)^\top (J(x_j) s_j^{GN} + r(x_j)) = 0. \tag{2.17}$$

We recognize here the normal equations of the following linear least-squares problem.

$$\min_{s \in \mathbb{R}^n} \tfrac{1}{2} \|J(x_j) s + r(x_j)\|^2. \tag{2.18}$$

In other words, solving a nonlinear least-squares problem involves iteratively solving linear least-squares problems. One limitation of the Gauss-Newton method is that there is no guarantee that $J(x_j)^\top J(x_j)$ is invertible, particularly when $J(x_j)$ is not full rank. Additionally, Gauss-Newton is highly dependent on the starting point x_0. Therefore, an improved method for solving nonlinear least-squares problems would be highly beneficial.

2.3 Levenberg-Marquardt algorithm

We present in this section the method designed by Levenberg [47] and Marquardt [50] to solve problem (2.14).

2.3.1 Presentation of the method

The key modification to ensure (2.17) has a unique solution is to introduce a positive regularization parameter σ_j to solve instead the system

$$(J(x_j)^\top J(x_j) + \sigma_j I_n)s_j = -J(x_j)^\top r(x_j), \tag{2.19}$$

which are the normal equations of a regularized linear least-squares problem:

$$\min_{s\in\mathbb{R}^n} \tfrac{1}{2}\|J(x_j)s + r(x_j)\|^2 + \tfrac{\sigma_j}{2}\|s\|^2 \iff \min_{s\in\mathbb{R}^n} \frac{1}{2}\left\| \begin{bmatrix} J(x_j) \\ \sqrt{\sigma_j} I_n \end{bmatrix} s + \begin{bmatrix} r(x_j) \\ 0 \end{bmatrix} \right\|^2. \tag{2.20}$$

This formulation ensures with $\begin{bmatrix} J(x_j)^\top & \sqrt{\sigma_j} I_n \end{bmatrix}^\top$, which is always full rank, that (2.20) has a unique solution. For clarification, introduce the notation corresponding to the model associated to (2.20):

$$m(s; x_j, \sigma_j) \coloneqq \tfrac{1}{2}\|J(x_j)s + r(x_j)\|^2 + \tfrac{\sigma_j}{2}\|s\|^2. \tag{2.21}$$

The LM algorithm is detailed in the pseudo-code of Algorithm 1.

Algorithm 1 Levenberg-Marquardt algorithm.

Initialization: Define $r : \mathbb{R}^n \to \mathbb{R}^m$, $J(x) = \nabla r(x)$, $0 < \eta_1 < \eta_2 < 1$, $\sigma_{\min} > 0$, $\lambda_1 < 1$, $\lambda_2 > 1$,$\varepsilon_a > 0$, $\varepsilon_r > 0$.
Choose x_0 and $\sigma_0 \geq \sigma_{\min}$.
Compute $r(x_0)$ and $J(x_0)$.
for $j = 0, 1, 2, \ldots$ **do**
    if $\|J(x_j)^\top r(x_j)\| \leq \varepsilon_a + \varepsilon_r \|J(x_0)^\top r(x_0)\|$ **then**
        Stop algorithm
    end if
    Compute an approximate minimizer s_j of (2.20)
    Compute the quality ratio

$$\rho_j \coloneqq \frac{f(x_j) - f(x_j + s_j)}{m(0; x_j, \sigma_j) - m(s_j; x_j, \sigma_j)}. \tag{2.22}$$

    if $\rho_j \geq \eta_1$ **then**, set $x_{j+1} = x_j + s_j$
        if $\rho_j \geq \eta_2$ **then**, set $\sigma_{j+1} = \max\{\lambda_1 \sigma_j, \sigma_{\min}\}$.
        end if
    else, set $x_{j+1} = x_j$ and $\sigma_{j+1} = \lambda_2 \sigma_j$.
    end if
end for

The stopping criterion of Algorithm 1 depends on both an absolute tolerance ε_a and a relative one $\varepsilon_r|J(x_j)^\top r(x_j)|$. The absolute tolerance ensures the norm of the gradient decreases beneath a specified threshold, while the relative criterion ensures the norm of the gradient decreases enough compared to its initial value.

At each iteration, the quality ratio ρ_j, defined in (2.22), is computed based on the actual reduction of the objective function,

$$A_{red} \coloneqq f(x_j) - f(x_j + s_j) \tag{2.23}$$

and the reduction of the objective function predicted by the model,

$$P_{red} \coloneqq m(0 : x_j, \sigma_j) - m(s_j; x_j, \sigma_j). \tag{2.24}$$

As s_j is an approximate minimizer of $m(s; x_j, \sigma_j)$, we have

$$0 < A_{red} \leq P_{red}. \tag{2.25}$$

Therefore, $\rho_j = \frac{A_{red}}{P_{red}} \in (0, 1]$. A ρ_j value closer to 1 indicates that the step s_j significantly reduces the objective function, while a value near 0 suggests the opposite. Consequently, ρ_j categorizes the computed step as successful, very successful, or failed. The step s_j is then accepted or rejected compared to threshold η_1, while if η_2 is fulfilled, the current iteration is considered very successful.

The LM algorithm converges globally with superlinear convergence [66], whereas the Gauss-Newton method achieves quadratic but local convergence. Other globalization techniques though exist, with trust-region (TR) schemes as the most famous [28]. The primary difference in the LM approach lies in solving the subproblem, which is transformed into a constrained optimization problem involving a trust-region radius $\Delta_j > 0$. The subproblem (2.20) becomes for TR strategies

$$\begin{aligned} &\min_{s \in \mathbb{R}^n} \tfrac{1}{2}\|J(x_j)s + r(x_j)\|^2, \\ &\text{s.t.} \quad \|s\| \leq \Delta_j. \end{aligned} \tag{2.26}$$

The main drawback of trust-region approaches is they transform the unconstrained subproblem into a constrained one, which is more complicated to solve. On the other hand, the quadratic regularization introduced in (2.20) is an explicit modification of the subproblem and involves an additional inaccuracy, while trust-region schemes do not change the subproblem objective to minimize. Each approach has its strengths and weaknesses, which must be considered

when designing an algorithm based on either scheme.

Thus, one of the main issues of Algorithm 1 is how to compute s_j. We present in the following section different ways to do this.

2.3.2 Methods to solve the subproblem

There are two main ways to solve approximately (2.20): direct methods and iterative (or Krylov) methods.

A direct method: QR decomposition

A way to solve the LM subproblem is to use a direct method, i.e., using a matrix factorization for the linear subproblem. The very used approach is the *QR factorization* which aims to find for a matrix $A \in \mathbb{R}^{m\times n}$ an upper triangular matrix $R \in \mathbb{R}^{m\times n}$ and an orthogonal one $Q \in \mathbb{R}^{m\times m}$ such that $A = QR$. Note an orthogonal matrix satisfies $Q^{-1} = Q^\top$.

This factorization is commonly used to solve linear systems as it can be applied to any matrix $A \in \mathbb{R}^{m\times n}$ and is unique as long as the diagonal terms of R are nonnegative. Besides, it reveals the rank of A as $\text{rank}(A) = |\{R_{i,i} \neq 0,\, i \in \{1,\dots,\min(m,n)\}\}|$. Note $A \in \mathbb{R}^{m\times n}$, overdetermined matrix ($m \geq n$), is full (column) rank when all its columns are linearly independent.

To solve (2.20), QR factorization of the leftmost matrix in (2.19) is computed. Let $\hat{Q}_j$ and $\hat{R}_j$ be the matrices from the QR decomposition of $J(x_j)^\top J(x_j) + \sigma_j I_n$:

$$\hat{Q}_j \hat{R}_j s = -\nabla f(x_j) \iff \hat{R}_j s = -\hat{Q}_j{}^\top \nabla f(x_j). \tag{2.27}$$

Since $\hat{R}_j$ is upper triangular, we can finish the resolution of the system (2.27) easily. It is possible to compute this QR factorization by different methods, but one of the most numerically robust and mainstream approaches uses Householder reflections [41]. A Householder reflection is a matrix of the form

$$H(v) \coloneqq I_n - 2vv^\top, \tag{2.28}$$

where $v \in \mathbb{R}^n$ is an unitary vector. For any unitary vector v, $H(v)$ is orthogonal. For all matrix $A = \left[a_{j,k}\right]_{1\leq j\leq m, 1\leq k\leq n} \in \mathbb{R}^{m\times n}$, we denote any submatrix of A by $A_{i:m,i:n} = \left[a_{j,k}\right]_{i\leq j\leq m, i\leq k\leq n} \in \mathbb{R}^{(m-i+1)\times(n-i+1)}$.

Considering then $A = \left[a_1, \cdots, a_m\right]$, for $u_1 = a_1 - \|a_1\|e_1$ and $v_1 = \frac{u_1}{\|u_1\|}$, $H(v_1) = \|a_1\|e_1$. So, using Householder reflections, we can build iteratively an upper triangular matrix $\bar{R}$, which

results in the QR decomposition of A:

$$A = Q_1^\top Q_2^\top \cdots Q_t^\top \bar{R}, \tag{2.29}$$

where $t = \min\{m-1, n\}$ and $Q_i = \begin{bmatrix} I_{i-1} & \\ & H(v_i) \end{bmatrix}$ with $H(v_i)$ the Householder reflection associated to $v_i \coloneqq \frac{u_i}{\|u_i\|}$ where $u_i \coloneqq a_i - \|a_i\| e_1^i$ with $e_1^i = [1, 0, \ldots, 0]^\top \in \mathbb{R}^{m-i+1}$ and a_i is the first column of $A_{i:m,i:n}$.

Another way of computing a QR factorization is to use Given's rotations [37] by nullifying subdiagonal elements of J using matrices of the form

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} j_k \\ j_l \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}. \tag{2.30}$$

Note that it is possible to compute the QR factorization of $\begin{bmatrix} J(x_j)^\top & \sqrt{\sigma_j} I_n \end{bmatrix}^\top$ economically by using Householder reflections for $J(x_j)$ and Given's rotations on the diagonal submatrix $\sqrt{\sigma_j} I_n$.

Iterative methods

One of the main disadvantages of direct methods is the need for an explicit matrix to compute the factorization, while iterative methods can approximate the inverse without constructing the matrix. Focus here on Krylov methods.

Let $A \in \mathbb{R}^{n \times n}$ and consider its characteristic polynomial $p_A(X) \coloneqq \det(A - XI_n)$. Write down the full expression of $p_A(X)$:

$$p_A(X) \coloneqq X^n + p_{n-1} X^{n-1} + \ldots p_1 X + p_0 I_n. \tag{2.31}$$

The Cayley-Hamilton theorem states that $p_A(A) = 0$, then, if $p_0 \neq 0$, we have

$$A(A^{n-1} + p_{n-1} A^{n-2} + \ldots p_1 I_n) = -p_0 I_n \iff -\frac{A}{p_0}(A^{n-1} + p_{n-1} A^{n-2} + \ldots p_1 I_n) = I_n. \tag{2.32}$$

In other words, if $p_0 \neq 0$, A is invertible and

$$A^{-1} = -\frac{1}{p_0}(A^{n-1} + p_{n-1} A^{n-2} + \ldots p_1 I_n). \tag{2.33}$$

It means that it is possible to iteratively construct the inverse of A with its successive powers.

Let $b \in \mathbb{R}^n$, the Krylov subspace generated by $\{b, Ab, \ldots, A^{n-1}b\}$ is denoted $\mathcal{K}_n(A, b)$. According to (2.33), if x is the solution of $Ax = b$, then $x \in \mathcal{K}_n(A, b)$. The basic idea behind Krylov methods for solving the linear system $Ax = b$ is to build a sequence of iterates $\{x_j\}_{j\geq 0}$ such that, for all $j \geq 0$, $x_j \in \mathcal{K}_n(A, b)$.

A Krylov method suitable for our framework is LSMR, as described in detail by Fong and Saunders [35].

2.4 Nonsmooth optimization

In this section, we introduce essential nonsmooth analysis notions presented in [58].

2.4.1 Background

Consider functions h that can take infinite values. Define $\overline{\mathbb{R}} \coloneqq \mathbb{R} \cup \{\pm\infty\}$ and let $h : \mathbb{R}^n \to \overline{\mathbb{R}}$. Notice that if $h(\bar{x}) = -\infty$ for $\bar{x} \in \mathbb{R}^n$, then $\bar{x} \in \operatorname*{argmin}_{x\in\mathbb{R}^n} h(x)$. Besides, if $\inf h = +\infty$, then $\operatorname*{argmin}_{x\in\mathbb{R}^n} h(x) = \emptyset$.

Yet, allowing functions to take infinite values is useful in an optimization framework. Indeed, in constrained optimization, one way to handle constraints is to add an extreme barrier relative to the feasible set C. Consider the problem

$$\min_{x\in C} h(x), \tag{2.34}$$

which can be transformed into an unconstrained optimization problem by rewriting it as

$$\min_{x\in\mathbb{R}^n} h(x) + \chi_C(x), \tag{2.35}$$

where $\chi_C(x)$ is the indicator function of set C defined by

$$\chi_C(x) \coloneqq \begin{cases} 0 & \text{if } x \in C \\ +\infty & \text{otherwise.} \end{cases} \tag{2.36}$$

In the following, consider functions that are nonsmooth, i.e., not differentiable on parts of their domain. In such cases, the gradient may no longer be accessible, necessitating the introduction of new concepts to handle nonsmooth issues.

Definition 6. *The effective domain of a function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$, denoted* $\operatorname{dom}(h)$*, is the set*

defined by

$$\mathrm{dom}(h) \coloneqq \{x \in \mathbb{R}^n \ : \ h(x) < +\infty\}. \tag{2.37}$$

Definition 7. *[58] A function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ is said proper if $h(x) > -\infty$ for all x and $f(x) < +\infty$ for at least one $x \in \mathbb{R}^n$.*

A proper function is then a function whose effective domain is not empty. Requiring a function to be proper in an optimization context simply demands that h can take finite values somewhere on $\mathbb{R}^n$. To introduce more variational concepts, it is necessary to extend the notion of limit. Denoting $\mathbb{B}(\bar{x}, r)$ the (closed) ball of center $\bar{x}$ of radius $r > 0$ for the norm $\|\cdot\|_2$ the set

$$\mathbb{B}(\bar{x}, r) \coloneqq \{x \in \mathbb{R}^n \ : \ \|x - \bar{x}\|_2 \leq r\}. \tag{2.38}$$

Definition 8. *[58, Definition 1.5] The limit inferior of a function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ at $\bar{x}$ is the value in $\overline{\mathbb{R}}$ defined by*

$$\liminf_{x \to \bar{x}} h(x) \coloneqq \lim_{\delta \searrow 0} \left(\inf_{x \in \mathbb{B}(\bar{x}, \delta)} h(x) \right). \tag{2.39}$$

The function h is lower semicontinuous (lsc) at $\bar{x}$ if

$$\liminf_{x \to \bar{x}} h(x) \geq h(\bar{x}) \quad \textit{or equivalently} \quad \liminf_{x \to \bar{x}} h(x) = h(\bar{x}). \tag{2.40}$$

Note that we have the equivalence since $\liminf_{x \to \bar{x}} h(x) \leq h(\bar{x})$ is always satisfied for all $\bar{x}$. With a sequential approach, considering a sequence $\{x_j\}$, $\liminf_{j \to +\infty} x_j$ represents the lowest value of the sequence after a specific integer N.

The *(lower) level set* of a function h, denoted $\mathrm{lev}_{\leq \alpha} h$ for $\alpha \in \mathbb{R}$, is defined by

$$\mathrm{lev}_{\leq \alpha} h \coloneqq \{x \in \mathbb{R}^n \ : \ h(x) \leq \alpha\}. \tag{2.41}$$

One has the following definition,

Definition 9. *[58, Definition 1.8] A function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ is (lower) level-bounded if for every $\alpha \in \mathbb{R}$ the set $\mathrm{lev}_{\leq \alpha} h$ is bounded (possibly empty).*

With these new notions, we have the following theorem.

Theorem 3. *[58, Theorem 1.9] Assume $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ proper, lsc, and level-bounded. Then the value of $\inf h$ is finite and the set $\operatorname{argmin}_{x \in \mathbb{R}^n} h(x)$ is nonempty and compact.*

This theorem is crucial as it guarantees the existence of a subsequence of iterates that converges to a minimum of h, a key feature for devising algorithms within a nonsmooth framework.

To introduce a useful result for our framework, we define "envelope" functions.

Definition 10. *[58, Definition 1.22] For a proper, lsc function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ and a parameter value $\lambda > 0$, the Moreau envelope function $e_\lambda h$ and the proximal mapping $P_\lambda h$ are defined by*

$$e_\lambda h(x) \coloneqq \inf_{u \in \mathbb{R}^n} \left\{ h(u) + \frac{1}{\lambda}\|x - u\|^2 \right\} \leq h(x) \tag{2.42}$$

$$P_\lambda h(x) \coloneqq \operatorname*{argmin}_{u \in \mathbb{R}^n} \left\{ h(u) + \frac{1}{\lambda}\|x - u\|^2 \right\}. \tag{2.43}$$

Graphically, Moreau envelopes bound below the function h with a quadratic penalty of parameter $1/\lambda$. They are useful to define a crucial notion in our context.

Definition 11. *[58, Definition 1.23] A function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ is prox-bounded if there exists $\lambda > 0$ such that $e_\lambda h(x) > -\infty$ for some $x \in \mathbb{R}^n$. The supremum of the set of all such λ is the threshold λ_h of prox-boundedness for h.*

This property of prox-boundedness can be interpreted as it imposes any quadratic regularization of h to be bounded below. Since LM introduces a quadratic regularization in the subproblem, this prox-boundedness feature is key to devising nonsmooth Levenberg-Marquardt algorithms. We finally state the following theorem.

Theorem 4. *[58, Theorem 1.25] Let $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ proper, lsc and prox-bounded with threshold $\lambda_h > 0$. Then for every $\lambda \in (0, \lambda_h)$ the set $P_\lambda h(x)$ is nonempty and compact and $e_\lambda h(x)$ is finite and depends continuously on (λ, x) with*

$$e_\lambda h(x) \nearrow h(x) \quad \textit{for all } x \textit{ as } \lambda \searrow 0. \tag{2.44}$$

This last theorem endorses a similar role as Theorem 3 with the weaker assumption of prox-boundedness rather than level-boundedness for quadratic regularized nonsmooth functions. Since a quadratic regularization naturally appears in the subproblem with the LM approach, Theorem 4 prevails for the analysis made in Chapter 3.

2.4.2 Subderivatives and proximal methods

Since nonsmoothness implies the inability to compute gradients, it is necessary to extend this notion to handle functions without gradients. For this reason, we introduce the notion of subderivatives and subdifferential.

Definition 12. *[58, Definition 8.1] For a function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ and a point $\bar{x}$ with $h(\bar{x})$ is finite, the subderivative function $dh(\bar{x}) : \mathbb{R}^n \to \overline{\mathbb{R}}$ is defined by*

$$dh(\bar{x})(\bar{\omega}) \coloneqq \liminf_{\tau \searrow 0,\, \omega \to \bar{\omega}} \frac{h(\bar{x} + \tau\omega) - f(\bar{x})}{\tau}. \tag{2.45}$$

This definition, alike Definition 8, is an extension of the definition of the common directional derivative in $\bar{\omega}$ including ω in the lim inf as well. As the derivative is extended, the gradient is as well,

Definition 13. *[3, Definition 2.1] Consider a function $\psi : \mathbb{R}^n \to \overline{\mathbb{R}}$ and a point $x^* \in \mathbb{R}^n$ with $\psi(x^*)$ finite. For a vector $v \in \mathbb{R}^n$, one says that v is a regular subgradient of ψ at x^*, written $v \in \hat{\partial}\psi(x^*)$ if*

$$\liminf_{x \to x^*,\, x \neq x^*} \frac{\psi(x) - \psi(x^*) - v^\top (x - x^*)}{\|x - x^*\|} \geq 0.$$

The set of regular subgradients is also called the Fréchet subdifferential. One says that $v \in \mathbb{R}^n$ is a (general) subgradient of ψ at x^*, and we write $v \in \partial\varphi(x^*)$, if there are sequences $\{x_k\}$ and $\{v_k\}$ such that

$$x_k \xrightarrow[k\to\infty]{} x^*, \quad \psi(x_k) \xrightarrow[k\to\infty]{} \psi(x^*), \quad v_k \in \hat{\partial}\psi(x_k) \quad \text{and} \quad v_k \xrightarrow[k\to\infty]{} v.$$

The set of general subgradients is called the limiting subdifferential.

We say that x^* is first order stationary for ψ if $0 \in \partial\psi(x^*)$. Besides, subdifferentials have a property similar to linearity.

Proposition 2. *[58, Theorem 10.1] Considering $g : \mathbb{R}^n \to \overline{\mathbb{R}}$ such that $g(x) = \varphi(x) + \psi(x)$ with $\varphi : \mathbb{R}^n \to \mathbb{R}$ where $\varphi \in \mathcal{C}^1$ and $\psi : \mathbb{R}^n \to \overline{\mathbb{R}}$. If $\partial\psi(\bar{x})$ exists at point $\bar{x} \in \mathbb{R}^n$, then*

$$\partial g(\bar{x}) = \nabla\varphi(\bar{x}) + \partial\psi(\bar{x}). \tag{2.46}$$

Furthermore, for any local minimizer x^* of $\varphi + \psi$ with $\varphi(x) = \frac{1}{2}\|r(x)\|^2$, using Proposition 2, we have

$$0 \in J(x^*)^\top r(x^*) + \partial\psi(x^*). \tag{2.47}$$

Introduce now a central operator used in many nonsmooth methods.

Definition 14. *Let a function $\psi : \mathbb{R}^n \to \overline{\mathbb{R}}$, a point x^* and a parameter $\nu > 0$. The proximal operator of $\nu\psi$ at $\bar{x}$ is defined as*

$$prox_{\nu\psi}(\bar{x}) \coloneqq \underset{u\in\mathbb{R}^n}{\operatorname{argmin}} \tfrac{1}{2}\|u - \bar{x}\|^2 + \nu\psi(u). \tag{2.48}$$

The prox finds the best compromise between minimizing the function $\nu\psi$ and staying as close as possible to the input vector $\bar{x}$. This notion is the base of prox-based optimization methods. One of the first of them is the prox-gradient method [36]. Consider the optimization problem

$$\min_{s\in\mathbb{R}^n} \varphi(s) + \psi(s), \tag{2.49}$$

where φ is smooth and ψ is nonsmooth. Then, the prox-gradient method generates a sequence $\{s_k\}$ of iterates such that

$$s_{k+1} \in \operatorname{prox}_{\nu\psi}(s_k - \nu_k\nabla\varphi(s_k)), \tag{2.50}$$

where $\nu_k > 0$ is the prox-gradient steplength for iteration k. The intuition behind (2.50) is we want to have a sequence that minimizes both ψ while being as close as possible to the steepest descent direction for φ. Prox-gradient methods also share the following property.

Proposition 3. *[14, Lemma 2] Let $\nabla\varphi$ be Lipschitz-continuous with Lipschitz constant $L > 0$, ψ be proper, lsc and $\inf\psi > -\infty$. Let $s_k \in \operatorname{dom}\psi$, $0 < \nu < 1/L$ and s_{k+1} be defined according to (2.50). Then,*

$$(\varphi + \psi)(s_{k+1}) \leq (\varphi + \psi)(s_k) - \tfrac{1}{2}(\nu^{-1} - L)\|s_{k+1} - s_k\|^2.$$

Consequently, iterates generated by a prox-gradient method ensure a decrease of $\varphi + \psi$.

2.4.3 Nonsmooth nonconvex optimization

Composite nonsmooth optimization has been studied for a long time. Several methods minimize $f(x) + h(c(x))$, where f and c are smooth, and h is convex and nonsmooth [34]. A function f is said convex when for all $\alpha \in [0, 1]$ and for all $x, y \in \mathbb{R}^n$ we have

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y). \tag{2.51}$$

Convexity is a strong property in optimization because, if f is convex, any local minimizer x^*

of f is a global minimizer. Recent works devised methods to minimize $f(x) + h(c(x))$ where f and h are convex or Lipschitz-continuous [25, 38, 59] but, in our framework, f and h are nonconvex.

Following recent work on proximal-based methods, we assume c is the identity function. Trust-region schemes pioneered [21, 25, 57] with recent complexity analyses [32, 33, 48, 52] to solve the problem where c is the identity function. Leconte and Orban [44] establish a worst-case complexity bound related to unbounded Hessian approximations. Other works developed proximal gradient methods using diagonal quasi-Newton updates (e.g., `TRDH` [45]) or modified quasi-Newton methods with possibly unbounded Hessians (e.g., `R2N` [31]). In our framework, we exploit the specific structure of nonlinear least squares to emancipate from second order models and assumptions.

Previous works minimizing $f(x) + h(x)$ assume f and h are nonconvex, such as [49], which designed an accelerated prox-gradient method requiring $f + h$ to be coercive for convergence. The definition of coercivity may vary depending on the function, but simply put, a function is coercive if it grows rapidly at the boundaries of its domain. Stella et al. [60] devised the algorithm `PANOC`, based on a line search limited-memory BFGS, that converges if $f+h$ respects the Kurdyka-Łojasiewicz (KŁ) assumption [13, 43]. Similarly, Bolte et al. [14] designed a proximal alternating linearized minimization (`PALM`) for partitioned variables, i.e., $x = (x_1, x_2)$ and $h(x) = h_1(x_1) + h_2(x_2)$ where h_1 and h_2 are nonconvex. Themelis et al. [62] also designed `ZeroFPR`, a non-monotone line search proximal quasi-Newton method, which still needs the KŁ assumption for global convergence similar to Boţ et al. [19] who devised a proximal method with momentum.

Our approach, laid out by nonsmooth `LM` and `LMTR` [3], aims to exploit the least-squares structure of (3.1) and curvature information to propose a method requiring neither coercivity nor KŁ assumptions.

Consider the problem

$$\min_{s \in \mathbb{R}^n} f(x) + h(x), \tag{2.52}$$

where $f(x) \coloneqq \frac{1}{2}\|r(x)\|^2$ with $r : \mathbb{R}^n \to \mathbb{R}^m$ and $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ nonsmooth. Both f and h are nonconvex. To solve the subproblem associated to (2.52) at iteration j, consider the current iterate x_j, the regularization parameter $\sigma_j > 0$, and models of f and h as follows:

$$m(s; x_j, \sigma_j) \coloneqq \varphi(s; x_j) + \psi(s; x_j) + \tfrac{1}{2}\sigma_j\|s\|^2, \tag{2.53a}$$

$$\varphi(s; x_j) \coloneqq \tfrac{1}{2}\|J(x_j)^\top s + r(x_j)\|^2, \tag{2.53b}$$

$$\psi(s; x_j) \approx h(x_j + s). \tag{2.53c}$$

The model described by (2.53a) is (2.20) with the model of $h(x+s)$ in addition. By construction, $\varphi(0; x_j) = f(x_j)$ and $\nabla_s \varphi(0; x_j) = \nabla f(x_j)$. Consider for the rest of this section that $\psi(\cdot; x_j)$ is proper, lsc, and prox-bounded of threshold λ_x. Assume as well that $\psi(0; x_j) = h(x_j)$ and $\partial\psi(0; x_j) = \partial h(x_j)$. These assumptions are automatically satisfied as long as h is proper, lsc, prox-bounded, and $\psi(s; x_j) = h(x_j + s)$ for all $s \in \mathbb{R}^n$. Introduce now

$$p(x_j, \sigma_j) \coloneqq \min_{s \in \mathbb{R}^n} m(s; x_j, \sigma_j), \tag{2.54a}$$

$$P(x_j, \sigma_j) \coloneqq \operatorname*{argmin}_{s \in \mathbb{R}^n} m(s; x_j, \sigma_j). \tag{2.54b}$$

Notice problem P is the subproblem associated to (2.52). We have then,

$$s \in P(x_j, \sigma_j) \iff 0 \in \nabla\varphi(s; x_j) + \sigma_j s + \partial\psi(s; x_j). \tag{2.55}$$

Since we do not yet have a stationarity measure for problem (2.52), we define

$$\xi(x_j, \sigma_j) = f(x_j) + h(x_j) - \varphi(s_j; x_j) - \psi(s_j; x_j), \quad \text{where} \quad s_j \in P(x_j, \sigma_j), \tag{2.56}$$

which represents the error between the objective function and its shifted model by a step solving (2.54b). With (2.56), we have the following proposition.

Proposition 4. *[3, Lemma 3.1] Let $x_j \in \mathbb{R}^n$ and $\sigma_j \geq \lambda_x^{-1}$. Then*

$$\xi(x_j, \sigma_j) = 0 \iff 0 \in P(x_j, \sigma_j) \Rightarrow x_j \text{ is first order stationary for problem (2.52).} \tag{2.57}$$

In addition, x_j is first order stationary for (2.52) if and only if $s_j = 0$ is first order stationary for (2.53a).

Thus, $\xi(x_j, \sigma_j)$ can be used to determine if the solution of the subproblem s_j does not allow any descent of the current point x_j. However, finding $s_j \in P(x_j, \sigma_j)$ exactly is nearly impossible. Therefore, we introduce a simpler model and its associated parametrized problems.

$$m_{cp}(s; x_j, \sigma_j) \coloneqq \varphi_{cp}(s; x_j) + \psi(s; x_j) + \tfrac{1}{2}\sigma_j \|s\|^2, \tag{2.58a}$$

$$\varphi_{cp}(s; x_j) \coloneqq f(x_j) + \left(J(x_j)^\top r(x_j)\right)^\top s, \tag{2.58b}$$

$$p_{cp}(x_j, \sigma_j) \coloneqq \min_{s \in \mathbb{R}^n} m_{cp}(s; x_j, \sigma_j), \tag{2.58c}$$

$$P_{cp}(x_j, \sigma_j) \coloneqq \operatorname*{argmin}_{s \in \mathbb{R}^n} m_{cp}(s; x_j, \sigma_j). \tag{2.58d}$$

The "cp" in (2.58a)-(2.58d) stands for *Cauchy point* as they aim for generalizing this notion from trust-region methods where it provides a sufficient decrease of the TR model. Here, our Cauchy Point is determined by the first step of a proximal gradient method that minimizes both $\varphi(s;x_j)+\psi(s;x_j)$ and $\varphi_{cp}(s;x_j)+\psi(s;x_j)$. Indeed, let $x_j\in\mathbb{R}^n$ and $\nu_j>0$,

$$\begin{aligned} s_{j,cp} &\in P_{cp}(x_j,\nu_j^{-1}) \\ &= \operatorname*{argmin}_{s\in\mathbb{R}^n} m_{cp}(s;x_j,\nu_j^{-1}) \\ &= \operatorname*{argmin}_{s\in\mathbb{R}^n} f(x_j)+\left(J(x_j)^\top r(x_j)\right)^\top s+\psi(s;x_j)+\tfrac{1}{2}\nu_j^{-1}\|s\|^2 \\ &= \operatorname*{argmin}_{s\in\mathbb{R}^n} \tfrac{1}{2}\|s+\nu_j J(x_j)^\top r(x_j)\|^2+\nu_j\psi(s;x_j) \\ &= \operatorname{prox}_{\nu_j\psi(\cdot;x_j)}(-\nu_j J(x_j)^\top r(x_j)). \end{aligned}$$

Thus, computing $s_{j,cp}$ requires only the computation of a prox. If $\nu_j^{-1}\geq\sigma_j$, we have $m(s;x_j,\sigma_j)\leq m(s;x_j,\nu_j^{-1})$ and a decrease is ensured. We define the following metric

$$\xi_{cp}(x_j,\nu_j^{-1})=f(x_j)+h(x_j)-\varphi_{cp}(s_{j,cp};x_j)-\psi(s_{j,cp};x_j),\quad \text{for}\quad s_{j,cp}\in P_{cp}(x_j,\nu_j^{-1}). \tag{2.59}$$

Analogously as Proposition 4, we can establish

Proposition 5. *[3, Lemma 3.3] Let $x_j\in\mathbb{R}^n$ and $\sigma_j>0$. Then*

$$\xi_{cp}(x_j,\nu_j^{-1})=0 \iff 0\in P_{cp}(x_j,\nu_j^{-1})\Rightarrow x_j \text{ is first order stationary for problem } (2.52). \tag{2.60}$$

In addition, x_j is first order stationary for (2.52) *if and only if $s_{j,cp}=0$ is first order stationary for* (2.58a)*.*

We have then an easier way to determine if x_j is a first-order stationary point for (2.52). In a smooth case framework, i.e., $h=0$, $s_{j,cp}=-\nu_j\nabla f(x_j)$ and

$$\xi_{cp}(x_j,\nu_j^{-1})=f(x_j)-\varphi_{cp}(s_{j,cp};x_j)=\nu_j\|\nabla f(x_j)\|^2. \tag{2.61}$$

Therefore, it invites us to consider $\nu_j^{-1/2}\xi_{cp}(x_j,\nu_j^{-1})^{1/2}$ as stationarity metric for an algorithm solving (2.52).

2.5 Stochastic optimization

We now address the second key feature of our research framework: a sampling scheme. We will first review the essential probability background, followed by existing methods for stochastic optimization.

2.5.1 Probability background

The problems described in Chapter 1 involve massive datasets and incur high computational costs for both r and J. All required information for nonlinear least squares reads in its residuals and the Jacobian of its residuals. Hence, reducing the computational cost of r and J will necessarily improve the overall efficiency and speed of the LM algorithm. The goal of this stochastic framework is to find a more computationally efficient way to calculate r and J. One simple approach is to compute them on a reduced sample of them.

Consider a problem p consisting of m equations and n variables, with the residual function $r(x) = (r_1(x), \ldots, r_m(x))$ and its Jacobian $J(x) \in \mathbb{R}^{m\times n}$ for all $x \in \mathbb{R}^n$. Let $\mathcal{S} \subseteq \{1, \ldots, m\}$ a random set of indexes. Then, considering a nonlinear least-squares framework , i.e., $f(x) \coloneqq \frac{1}{2}\|r(x)\|^2 = \sum_{i=1}^{m} r_i(x)^2$, the sampled version of f denoted $f_{\mathcal{S}}$ would be

$$f_{\mathcal{S}}(x) \coloneqq \sum_{i\in\mathcal{S}} r_i(x)^2. \tag{2.62}$$

This strategy immediately improves computation speed as long as $|\mathcal{S}| < m$. But, as f is a sum of nonnegative terms,

$$f_{\mathcal{S}}(x) \leq f(x). \tag{2.63}$$

As a consequence, considering $\mathcal{S}_1 \subseteq \mathcal{S}_2$, for all $x \in \mathbb{R}^n$, we have $f_{\mathcal{S}_1} \leq f_{\mathcal{S}_2}$. Thus, comparing two estimates with different samples is flawed.

Moreover, an algorithm relying on estimates may provide descent directions based on approximations, potentially accepting a step that increases the deterministic objective function. However, it is convenient to allow stochasticity in optimization, like noisy r and/or J, as it may lead the optimization to a different local minimum, which might be better than the solution returned by a deterministic approach. A stochastic scheme can also lead to faster convergence due to reduced computational costs.

Review then the essentials of probabilistic background to familiarize to specific notations and concepts stated in [15–17].

Definition 15. *A probability space is a triple* $(\Omega, \mathcal{F}, \mathbf{P})$ *where*

- Ω *is a set representing all possible outcomes,*
- $\mathcal{F}$ *is a* σ*-algebra representing the set of all events. An event is a subset of* Ω*,*
- $\mathbf{P} : \mathcal{F} \to [0,1]$ *is a probability measure such that* $\mathbf{P}(\Omega) = 1$ *and, for disjoint events denoted* $E_1, E_2, \cdots \in \mathcal{F}$*, we have*

$$\mathbf{P}\left(\bigcup_{i=1}^{+\infty} E_i\right) = \sum_{i=1}^{+\infty} \mathbf{P}(E_i). \tag{2.64}$$

The probability measure $\mathbf{P}$ possesses many properties that are closely related to set properties. To cite some of them, consider events $A, B \in \mathcal{F}$,

- $\mathbf{P}(A \cup B) = \mathbf{P}(A) + \mathbf{P}(B) - \mathbf{P}(A \cap B)$.
- Denoting A^c the contrary event of A, $\mathbf{P}(A^c) = 1 - \mathbf{P}(A)$, and consequently, $\mathbf{P}(\emptyset) = 0$.
- If $A \subseteq B$, then $\mathbf{P}(A) \leq \mathbf{P}(B)$.

Another crucial notion in probabilities is the definition of random variables and expectation.

Definition 16. *A random variable* X *is a measurable function from a probability space* $(\Omega, \mathcal{F}, \mathbf{P})$ *to the reals such that for any set* B*,*

$$X^{-1}(B) = \{\omega \in \Omega \ : \ X(\omega) \in B\} \in \mathcal{F}. \tag{2.65}$$

We use in the sequel the common simplified notation:

$$\{X = x\} \coloneqq \{\omega \in \Omega \ : \ X(w) = x\}, \tag{2.66}$$

for any realization x of X.

Definition 17. *The expectation of a random variable* X *given on a probability space* $(\Omega, \mathcal{F}, \mathbf{P})$ *is defined as the quantity*

$$\mathbb{E}[X] \coloneqq \int_{\Omega} X(\omega) d\mathbf{P}(d\omega), \tag{2.67}$$

where the integral is the Lebesgue integral.

Even if Definition 17 may appear very abstract, it simply tells the expectation represents the *mean* or the *average* of possible outcomes of a random variable. For instance, considering a discrete variable X that takes a finite set of possible values $\{a_1, a_2, \ldots, a_n\}$, then,

$$\mathbb{E}[X] = \sum_{i=1}^{n} a_i \mathbf{P}(X = a_i). \tag{2.68}$$

In probability, it is common to encounter events conditioned by previous ones. This notion of conditioning is closely related to independence too.

Definition 18. *Let $(\Omega, \mathcal{F}, \mathbf{P})$ be a probability space and A and B two arbitrary events. A and B are independent if*

$$\mathbf{P}(A \cap B) = \mathbf{P}(A)\mathbf{P}(B). \tag{2.69}$$

Definition 18 can be generalized to a larger collection of events and to random variables' framework too. We now define the notion of conditioned events.

Definition 19. *Let $(\Omega, \mathcal{F}, \mathbf{P})$ be a probability space and A and B two arbitrary events. If $\mathbf{P}(B) > 0$, the conditional probability of the event A given B has occurred is denoted $\mathbf{P}(A \mid B)$ and is defined by*

$$\mathbf{P}(A \mid B) \coloneqq \frac{\mathbf{P}(A \cap B)}{\mathbf{P}(B)}. \tag{2.70}$$

As an immediate consequence, considering two independent events A and B on a probability space $(\Omega, \mathcal{F}, \mathbf{P})$, then

$$\mathbf{P}(A \mid B) = \mathbf{P}(A). \tag{2.71}$$

In other words, if two events are independent, the probability that one occurs does not impact the realization of the other one. This notion can be extended to expectations as well.

Definition 20. *Let X be a random variable defined on a probability space $(\Omega, \mathcal{F}, \mathbf{P})$ and $B \in \mathcal{F}$ such that $\mathbf{P}(B) > 0$. The conditional expectation of X given B is defined by*

$$\mathbb{E}[X \mid B] \coloneqq \frac{\int_B X(\omega) d\mathbf{P}(d\omega)}{\mathbf{P}(B)}. \tag{2.72}$$

Definition 20 combines the ideas of Definitions 17 and 19. With this background, we are now prepared to address the general framework of an LM algorithm with nonsmooth regularization based on estimates.

2.5.2 Existing methods for stochastic optimization

Many problems encounter a nonlinear least-squares structure like inverse problems [29, 61, 63], but may have a huge amount of data so they face both storing and numerical issues like in Machine Learning field [18] and Bundle Adjustment problems [23, 64]. As a result, sampling strategies emerged to reduce the computational cost of evaluating the objective function and its derivatives. The first works came out with Gauss-Newton approaches [20]. They have been later improved thanks to Cartis et al. [24] by setting a derivative-free framework and Herrmann et al. [40] who designed a Gauss-Newton algorithm with a $l1$-penalized subproblem. Besides, Zhou et al. [67] designed an incremental Gauss-Newton (`IGN`) approach to solve $f(x) = 0$. It also uses a mini-batch approach (`MB-IGN`) with a Hölder continuity assumption on the Jacobian impacting the convergence rate. El Houcine et al. recently used an unbiased stochastic sketching operator, similar to stochastic gradient descent [39], to provide the iteratively regularized Gauss-Newton method with a general stochastic framework (`SIRGNM`) [10].

Bellavia et al. [6] developed an LM method that implements a dynamic accuracy threshold for r and J estimations. As this scheme imposes estimates to respect this specific accuracy threshold, the algorithm has to be able to reduce the noise if necessary. Blanchet et al. [27] designed a trust-region algorithm that handles noisy functions through probabilistic models. This TR scheme has a complexity bound in $O(\varepsilon^{-2})$ in expectation for the norm of the gradient to be lower than a given accuracy ε [27]. As it can involve two sources of randomness, the analysis we carry out in Chapter 3 accounts for different estimates whether for the model or the objective function evaluations.

Like stochastic `LM` [9], we incorporate estimates to compute the least-squares residuals and their Jacobian.

CHAPTER 3 A STOCHASTIC LEVENBERG-MARQUARDT METHOD FOR NONSMOOTH REGULARIZED INVERSE PROBLEMS

In this chapter, we introduce a variant of the LM algorithm adding stochastic estimates and a nonsmooth regularizer. Section 3.1 presents the modified LM algorithm using probabilistic models and nonsmooth accuracy metrics. The main assumptions and helpful lemmas are shown in Section 3.2. Both convergence and complexity analysis are detailed in Section 3.3 while Section 3.4 displays the numerical tests carried out on nonlinear SVMs and Bundle Adjustment problems.

3.1 A probabilistic LM algorithm based on estimated values

We consider the nonsmooth regularized nonlinear least-squares problem

$$\min_{x\in\mathbb{R}^n} \; f(x) + h(x), \tag{3.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is defined by $f(x) \coloneqq \frac{1}{2}\|r(x)\|^2$, with $r : \mathbb{R}^n \to \mathbb{R}^m$ continuously differentiable, and the regularizer $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is proper, lower semi-continuous (lsc) and prox-bounded with $\lambda_x > 0$ the *threshold of prox-boundedness*. Both f and h may be nonconvex.

We consider a context where r and its Jacobian cannot be exactly computed because of a numerical cost that is too high. The framework of the LM variant we want to design is a merge of both a stochastic `LM` [9] and a nonsmooth regularized `LM` [3].

3.1.1 Models and estimates

Let Ω_j represents all the possible executions of our algorithm at iteration j and $\omega_j \in \Omega_j$.

At iteration j, the algorithm only has access to stochastic estimates of the residual and the Jacobian

$$r_j \coloneqq r(x_j, \omega_j) \approx r(x_j), \quad J_j \coloneqq J(x_j, \omega_j) \approx J(x_j), \quad f_j \coloneqq \tfrac{1}{2}\|r_j\|^2, \quad \text{and} \quad g_j \coloneqq J_j^\top r_j. \tag{3.2}$$

We introduce the LM regularization parameter $\sigma_j > 0$. We compute a trial step s_j as an

approximate minimizer of a model of $f + h$ about x_j,

$$m(s; x_j, \sigma_j) \coloneqq \varphi(s; x_j) + \psi(s; x_j) + \tfrac{1}{2}\sigma_j \|s\|^2, \tag{3.3a}$$

$$\varphi(s; x_j) \coloneqq \tfrac{1}{2}\|J_j^\top s + r_j\|^2, \tag{3.3b}$$

$$\psi(s; x_j) \approx h(x_j + s). \tag{3.3c}$$

By construction, $\varphi(0; x_j) = f_j$ and $\nabla_s \varphi(0; x_j) = g_j$. The model $\psi(\cdot\,; x_j)$ must satisfy the following assumption.

Assumption 1. *For all $x_j \in \mathbb{R}^n$, $\psi(\cdot\,; x_j)$ is proper, lsc and prox-bounded with threshold $\lambda_x \in \mathbb{R}_+ \cup \{+\infty\}$. In addition, $\psi(0; x_j) = h(x_j)$ and $\partial\psi(0; x_j) = \partial h(x_j)$.*

We consider a simpler first-order model that allows us to define a stationary measure, set minimal requirements on steps computed during the iterations of the algorithm, and derive a complexity analysis. This first-order model generalizes the concept of *Cauchy point* ("cp"). For fixed $\nu_j > 0$, we define

$$\varphi_{\text{cp}}(s; x_j) \coloneqq f_j + {g_j}^\top s, \tag{3.4a}$$

$$m_{\text{cp}}(s; x_j, \nu_j^{-1}) \coloneqq \varphi_{\text{cp}}(s; x_j) + \psi(s; x_j) + \tfrac{1}{2}\nu_j^{-1}\|s\|^2, \tag{3.4b}$$

$$p_{\text{cp}}(x_j, \nu_j^{-1}) \coloneqq \min_{s \in \mathbb{R}^n} m_{\text{cp}}(s; x_j, \nu_j^{-1}), \tag{3.4c}$$

$$P_{\text{cp}}(x_j, \nu_j^{-1}) \coloneqq \operatorname*{argmin}_{s \in \mathbb{R}^n} m_{\text{cp}}(s; x_j, \nu_j^{-1}), \tag{3.4d}$$

$$\xi_{\text{cp}}(x_j, \nu_j^{-1}) \coloneqq f_j + h(x_j) - (\varphi_{\text{cp}}(s_{j,\text{cp}}; x_j) + \psi(s_{j,\text{cp}}; x_j)) \geq 0, \tag{3.4e}$$

where $s_{j,\text{cp}} \in P_{\text{cp}}(x_j, \nu_j^{-1})$. Hence, using [14, Lemma 2],

$$\xi_{\text{cp}}(x_j, \nu_j^{-1}; \omega_j) \geq \tfrac{1}{2}\nu_j^{-1}\|s_{j,\text{cp}}\|^2 \geq 0. \tag{3.5}$$

In the smooth case, i.e., $h = 0$ and $\psi = 0$, $s_{j,\text{cp}} = -\nu_j g_j$ and

$$\xi_{\text{cp}}(x_j, \nu_j^{-1}; \omega_j) = \tfrac{1}{2}\nu_j^{-1}\|s_{j,\text{cp}}\|^2 = \tfrac{1}{2}\nu_j\|g_j\|^2,$$

which suggests $\nu_j^{-1/2}\xi_{\text{cp}}(x_j, \nu_j^{-1}; \omega_j)^{1/2}$ as a stationarity measure that generalizes the norm of the gradient to the nonsmooth setting. In the sequel, we set

$$\xi_j \coloneqq \nu_j^{-1/2}\xi_{\text{cp}}(x_j, \nu_j^{-1}; \omega_j)^{1/2}. \tag{3.6}$$

We will use a "star" superscript to indicate quantities obtained deterministically, e.g.,

$P^*_{\text{cp}}(x_j, \nu_j^{-1})$, $s^*_{j,\text{cp}}$, $\xi^*_{\text{cp}}(x_j, \nu_j^{-1})$, etc.

To derive complexity bounds in the smooth case (i.e., $h = 0$), Bergou et al. [9] set σ_j as a multiple of the gradient norm. By analogy, we set

$$\sigma_j \coloneqq \mu_j \xi_j, \tag{3.7}$$

where μ_j is updated following the classical LM mechanism update based on an achieved decrease in stochastic estimates of f. We set the step length ν_j to

$$\nu_j \coloneqq \theta(\|J_j\|^2 + \mu_{\min})^{-1}, \tag{3.8}$$

where $\theta \in (0, 1)$ and $\mu_{\min} > 0$ are preset parameters. The complete procedure is summarized in Algorithm 2.

The step s_j is required to satisfy *Cauchy decrease*, which we define as in [2, 3, 31]:

$$\varphi(0; x_j) + \psi(0; x_j) - (\varphi(s_j; x_j) + \psi(s_j; x_j)) \geq \kappa_{mdc} \xi_{\text{cp}}(x_j, \nu_j^{-1}), \tag{3.9}$$

for $\kappa_{mdc} \in (0, 1)$ given. In other words, s_j must result in a decrease in $\varphi(\cdot; x_j) + \psi(\cdot; x_j)$ that is at least a fraction of that achieved in the first-order model $\varphi_{\text{cp}}(\cdot; x_j) + \psi(\cdot; x_j)$ by the Cauchy step s_{cp} and a well-chosen step length ν_j.

At Line 13 of Algorithm 2, the ratio ρ_j assesses the quality of s_j by comparing the decrease achieved in $f(\cdot; \omega_j) + h$ to that predicted by the model. A generalization of our approach consists in using a different estimate of $f(x_j)$ than $f_j = \varphi(0; x_j)$ in the numerator of ρ_j, denoted f_j^0 while preserving convergence and complexity properties [8]. To simplify in practice, we use the readily available estimate f_j, which is also what would happen in most implementations.

3.2 Probabilistic properties of models and function estimates

The framework of Algorithm 2 allows for approximations of the objective function and its derivatives to construct stochastic estimates for both the model and the function values. In this section, we consider that the function values and the derivatives can only be accessed through noisy approximations, and we define accuracy formulas in a deterministic and probabilistic sense.

Definition 21. *[9, Definition 3.1] Consider a realization of Algorithm 2, and the model m of f defined around the iterate x_j, and let $\kappa_f, \kappa_g, \kappa_\xi > 0$. Then, the model m is called*

Algorithm 2 PLM: A probabilistic LM method using random models and estimates.

Require: $\eta_1 \geq 1 > \eta_2 > 0$, $\eta_3 > 0$, $\mu_{\min} > 0$, $\theta \in (0,1)$, $\lambda > 1$, $x_0 \in \mathbb{R}^n$, and $\mu_0 \geq \mu_{\min}$.

1: **for** $j = 0, 1, 2, \ldots$ **do**
2: Select $\omega_j \in \Omega_j$ and a random sample $\mathcal{S}_j \subset \{1, \ldots, m\}$.
3: Compute estimates r_j and J_j and corresponding f_j and g_j as in (3.2).
4: Compute estimate f_j^0 of $f(x_j)$ by sampling using $\mathcal{S}_j$
5: Set $\nu_j := \theta(\|J_j\|^2 + \mu_{\min})^{-1}$.
6: Compute $s_{j,cp} \in P_{\rm cp}(x_j, \nu_j^{-1})$.
7: Compute ξ_j as in (3.6) and set $\sigma_j \coloneqq \mu_j \xi_j$.
8: Compute an approximate minimizer s_j of $m(\cdot\,; x_j, \sigma_j)$ that satisfies (3.9).
9: **if** $\|s_j\| > \eta_1 \|s_{j,\rm cp}\|$ **then**
10: Set $s_j \coloneqq s_{j,\rm cp}$.
11: **end if**
12: Compute an estimate f_j^+ of $f(x_j + s_j)$ by sampling using $\mathcal{S}_j$.
13: Set

$$\rho_j \;\coloneqq\; \frac{f_j^0 + h(x_j) - (f_j^+ + h(x_j + s_j))}{\varphi(0; x_j) + \psi(0; x_j) - (\varphi(s_j; x_j) + \psi(s_j; x_j))}. \tag{3.10}$$

14: **if** $\rho_j \geq \eta_2$ and $\xi_j \geq \eta_3/\mu_j$ **then** set $x_{j+1} = x_j + s_j$ and $\mu_{j+1} = \max\{\mu_j/\lambda, \mu_{\min}\}$.
15: **else**, set $x_{j+1} = x_j$ and $\mu_{j+1} = \lambda \mu_j$.
16: **end if**
17: **end for**

$(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate *with respect to* (x_j, μ_j) *if the following properties hold:*

$$\left\|\nabla f(x_j) - g_j^m\right\| \leq \frac{\kappa_g}{\mu_j}, \quad \left|\nu_j^{-1/2} \xi_{\rm cp}^*(x_j, \nu_j^{-1})^{1/2} - \xi_j\right| \leq \frac{\kappa_\xi}{\mu_j} \quad \text{and} \quad \left|f(x_j) - f_j\right| \leq \frac{\kappa_f}{\mu_j^2}. \tag{3.11}$$

In smooth case, i.e., $h = 0$, $\xi_{j,cp}^m(x_j, \nu_j^{-1}) = \nu_j \|g_j^m\|^2$ and $\xi_{j,cp}^*(x_j, \nu_j^{-1}) = \nu_j \|\nabla f(x_j)\|^2$, then,

$$\begin{aligned}
\left|\nu_j^{-1/2} \xi_{\rm cp}^*(x_j, \nu_j^{-1})^{1/2} - \xi_j\right| &= \left|\sqrt{\frac{\nu_j \|\nabla f(x_j)\|^2}{\nu_j}} - \sqrt{\frac{\nu_j \|g_j^m\|^2}{\nu_j}}\right| \\
&= \left|\|\nabla f(x_j)\| - \|g_j^m\|\right| \\
&\leq \|\nabla f(x_j) - g_j^m\| \\
&\leq \frac{\kappa_g}{\mu_j}.
\end{aligned}$$

Definition 22. *[9, Definition 3.3] Consider a realization of Algorithm 2, and the residual estimates f_j^0 and f_j^+ computed at iteration j. Given $\varepsilon_f > 0$, we say that f_j^0 and f_j^+ are*

ε_f-accurate estimates of $f(x_j)$ and $f(x_j + s_j)$ *with respect to* (x_j, μ_j) *if*

$$\left|f_j^0 - f(x_j)\right| \leq \frac{\varepsilon_f}{\mu_j^2} \quad \text{and} \quad \left|f_j^+ - f(x_j + s_j)\right| \leq \frac{\varepsilon_f}{\mu_j^2}. \tag{3.12}$$

3.2.1 Probabilistic properties

The randomness introduced by the stochastic variable ω_j implies that the iterates, regularization parameters, and trial steps become stochastic processes. We now introduce random variables that represent the process of our algorithm along its iterations as it computes inexact residuals and Jacobian [9]. Let X_j, S_j, Σ_j, M_j, V_j and Ξ_j be random variables whose realizations are x_j, s_j, σ_j, μ_j, ν_j and ξ_j, respectively. Let also R_j and $\mathcal{J}_j$ be random variables whose realizations are r_j and J_j respectively, and denote $F_j = \frac{1}{2}\|R_j\|$ and $G_j = \mathcal{J}_j^\top R_j$. Analogously, we define F_j^0 and F_j^+ as the stochastic process generating f_j^0 and f_j^+ respectively. The two following definitions generalize Definitions 21 and 22 to the probabilistic setting.

Definition 23. *[9, Definition 3.4] Let $p \in (0, 1]$, $\kappa_f > 0$, $\kappa_g > 0$, $\kappa_\xi > 0$, and consider the sequence $\{(X_j, M_j)\}$. For a sequence of random variables $\{(R_j, \mathcal{J}_j)\}$ and denote $F_j = \frac{1}{2}\|R_j\|^2$, $G_j = \mathcal{J}_j^\top R_j$ and $\xi_j = V_j^{-1/2}\xi_{cp}(X_j, V_j^{-1})^{1/2}$. Define the events $U_{F,j} \coloneqq \{|f(X_j) - F_j| \leq \kappa_f / M_j^2\}$, $U_{G,j} \coloneqq \{\|\nabla f(X_j) - G_j\| \leq \kappa_g / M_j\}$, and $U_{\xi,j} \coloneqq \{|V_j^{-1/2}\xi_{cp}^*(X_j, V_j^{-1})^{1/2} - \xi_j| \leq \kappa_\xi / M_j\}$. We say that $\{(F_j, G_j, \xi_j)\}$ is p-probabilistically $\{\kappa_f, \kappa_g, \kappa_\xi\}$-first-order accurate with respect to $\{(X_j, M_j)\}$ if the event $U_j \coloneqq U_{F,j} \cap U_{G,j} \cap U_{\xi,j}$ satisfy*

$$p_j^* \;\coloneqq\; \mathbf{P}(U_j \mid \mathcal{F}_{j-1}) \;\geq\; p, \tag{3.13}$$

where $\mathcal{F}_{j-1}$ is the σ-algebra generated by $\{R_0, \mathcal{J}_0, \ldots, R_{j-1}, \mathcal{J}_{j-1}\} \cup \{F_0^0, F_0^+, \ldots, F_{j-1}^0, F_{j-1}^+\}$.

Definition 24. *[9, Definition 3.5] Let $\varepsilon_f > 0$, $q \in (0, 1]$, and consider the sequence $\{(X_j, M_j)\}$. Define the events $W_{0,j} \coloneqq \{|F_j^0 - f(X_j)| \leq \varepsilon_f / M_j^2\}$ and $W_{+,j} \coloneqq \{|F_j^+ - f(X_j)| \leq \varepsilon_f / M_j^2\}$. A sequence of random quantities $\{(F_j^0, F_j^+)\}$ is called q-probabilistically ε_f-accurate with respect to $\{(X_j, M_j)\}$ if the event $W_j \coloneqq W_{0,j} \cup W_{+,j}$ satisfy*

$$q_j^* \coloneqq \mathbf{P}(W_j \mid \mathcal{F}_{j-1/2}) \geq q, \tag{3.14}$$

where $\mathcal{F}_{j-1/2}$ is the σ-algebra generated by $\{R_0, \mathcal{J}_0, \ldots, R_j, \mathcal{J}_j\} \cup \{F_0^0, F_0^+, \ldots, F_{j-1}^0, F_{j-1}^+\}$.

We can then establish the following proposition.

Proposition 6. *Let $\{X_j\}_{j \geq 0}$ be a sequence of random variables generated by Algorithm 2 and consider the j^{th} iteration. Assume that $\psi(\cdot\,; X_j)$ satisfies Assumption 1 and $V_j > 0$. For*

any $\alpha \in (0, 1]$,

$$\mathbf{P}(\Xi_j = 0) \geq \alpha \iff \mathbf{P}(0 \in P_{\text{cp}}(X_j, V_j^{-1})) \geq \alpha \tag{3.15}$$

$$\implies \mathbf{P}(X_j \textit{ is a stationary point for } (3.1)) \geq \alpha. \tag{3.16}$$

Proof. Let x_j be a realization of X_j. Since $\psi(\cdot\,; x_j)$ satisfies Assumption 1, we have

$$\xi_{\text{cp}}(x_j, \nu_j^{-1}) = 0 \iff 0 \in P_{\text{cp}}(x_j, \nu_j^{-1}). \tag{3.17}$$

Hence, $\mathbf{P}(\xi_{\text{cp}}(X_j, V_j^{-1}) = 0) \geq \alpha \iff \mathbf{P}(0 \in P_{\text{cp}}(X_j, V_j^{-1})) \geq \alpha$. In addition,

$$s \in P_{\text{cp}}(x_j, \nu_j^{-1}) \implies 0 \in \partial m_{\text{cp}}(s; x_j, \nu_j^{-1}) = \nabla \varphi_{\text{cp}}(s; x_j) + \partial \psi(s; x_j) + \nu_j^{-1} s.$$

Thus,

$$\begin{aligned}\left\{0 \in P_{\text{cp}}(X_j, V_j^{-1})\right\} &\subseteq \left\{0 \in \nabla \varphi_{j,cp}(0; X_j) + \partial \psi(0; X_j) = J(X_j)^\top r(X_j) + \partial h(X_j)\right\} \\ &= \left\{X_j \text{ is a stationary point}\right\}.\end{aligned}$$

Therefore, we have $\mathbf{P}(0 \in P_{j,cp}(X_j, V_j^{-1})) \leq \mathbf{P}(\left\{X_j \text{ is a stationary point}\right\})$. As $\mathbf{P}(0 \in P_{j,cp}(X_j, V_j^{-1})) \geq \alpha$,

$$\mathbf{P}(\left\{X_j \text{ is a stationary point}\right\}) \geq \alpha$$

□

3.2.2 Deterministic useful results

We state our deterministic assumptions.

Assumption 2. *f is continuously differentiable on an open set containing the level set $\mathcal{L}(f(x_0)+h(x_0)) \coloneqq \{x \in \mathbb{R}^n | f(x) + h(x) \leq f(x_0) + h(x_0)\}$, with Lipschitz continuous gradient of Lipschitz constant L.*

Assumption 3. *There exists $\kappa_J > 0$ such that for all j, $\|J_j\| \leq \kappa_J$.*

Assumption 4. *There exists $\kappa_m > 0$ such that for all j, the step s_j computed at Line 8 of Algorithm 2 satisfies*

$$|f_j + {g_j}^\top s_j + h(x_j + s_j) - (\varphi(s_j; x_j) + \psi(s_j; x_j))| \leq \kappa_m \|s_j\|^2. \tag{3.18}$$

If we set $\psi(s; x_j) = h(x_j + s)$ for all $s \in \mathbb{R}^n$, (3.18) holds with $\kappa_m = \frac{1}{2}\kappa_J^2$ using Assumption 3. The following assumption states that the trial step satisfies desirable bounds on its norm.

Assumption 5. *There exists $\kappa_s > 0$ such that for s_j computed at Line 8 of Algorithm 2 satisfies*

$$\|s_j\| \leq \kappa_s/\mu_j \tag{3.19}$$

Assumption 5 generalizes [9, Assumption 4.5] to the nonsmooth setting. In fact, when $h = 0$, one can show that $\|s_j\| \leq \frac{\|g_j\|}{\sigma_j} = \frac{1}{\mu_j}$, e.g., [7, Lemma 5.1].

The next lemmas describe useful results for our complexity analysis. The obtained results hold for any realization of Algorithm 2.

Lemma 1. *Let Assumptions 1, 2, 4 and 5 hold for a realization of Algorithm 2. Consider the j-th iteration of that realization, and suppose that the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate. Then,*

$$\left|f(x_j + s_j) + h(x_j + s_j) - \varphi(s_j; x_j) - \psi(s_j; x_j)\right| \;\leq\; \kappa_1/\mu_j^2, \tag{3.20}$$

with $\kappa_1 \coloneqq \kappa_f + \kappa_g\kappa_s + \kappa_s^2\left(\kappa_m + \frac{L}{2}\right)$.

Proof. Using Assumptions 1, 2, 4 and 5 and Taylor expansion of f around x_j, we have

$$\begin{aligned}
&\left|f(x_j + s_j) + h(x_j + s_j) - \varphi(s_j; x_j) - \psi(s_j; x_j)\right| \\
&\qquad\leq \left|f(x_j + s_j) - f_j - {g_j}^\top s_j\right| + \kappa_m\|s_j\|^2 \\
&\qquad\leq \left|f(x_j) + \nabla f(x_j)^\top s_j - f_j - {g_j}^\top s_j\right| + \frac{L + 2\kappa_m}{2}\|s_j\|^2 \\
&\qquad\leq \left|f(x_j) - f_j\right| + \|s_j\|\|\nabla f(x_j) - g_j\| + \frac{L + 2\kappa_m}{2}\|s_j\|^2 \\
&\qquad\leq \frac{2\kappa_f + 2\kappa_g\kappa_s + \kappa_s^2(L + 2\kappa_m)}{2\mu_j^2}.
\end{aligned}$$

□

The next lemma shows a decrease on $f + h$ with respect to the inexact stationarity measure.

Lemma 2. *Let Assumptions 1 to 5 hold for a realization of Algorithm 2, and consider iteration j. If the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate and*

$$\mu_j \geq \frac{\kappa_2}{\xi_j}, \quad \textit{with} \quad \kappa_2 \coloneqq \sqrt{\frac{3(\kappa_f + \kappa_1)(\kappa_J^2 + \mu_{\min})}{2\kappa_{mdc}\theta}}, \tag{3.21}$$

then the trial step s_j satisfies

$$(f+h)(x_j) - (f+h)(x_j+s_j) \geq \kappa_3 \frac{\xi_j}{\mu_j}, \quad \textit{with} \quad \kappa_3 \coloneqq \frac{\kappa_1+\kappa_f}{2\kappa_2}. \tag{3.22}$$

Proof. Since the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate, using Assumptions 1 to 5 and Lemma 1 we have

$$\begin{aligned}
&(f+h)(x_j+s_j) - (f+h)(x_j) \\
&\quad = \Big((f+h)(x_j+s_j) - \varphi(s_j;x_j) - \psi(s_j;x_j)\Big) + \varphi(s_j;x_j) + \psi(s_j;x_j) - (f+h)(x_j) \\
&\quad \leq \frac{\kappa_1}{\mu_j^2} + \Big(\varphi(s_j;x_j) + \psi(s_j;x_j) - \varphi(0;x_j) - \psi(0;x_j)\Big) + f_j - f(x_j) + \psi(0;x_j) - h(x_j) \\
&\quad \leq \frac{\kappa_1}{\mu_j^2} - \kappa_{mdc}\xi_{cp}(x_j;\nu_j^{-1}) + \frac{\kappa_f}{\mu_j^2} \\
&\quad \leq \frac{\kappa_1+\kappa_f}{\mu_j^2} - \kappa_{mdc}\nu_j\xi_j^2 \\
&\quad \leq \frac{\kappa_1+\kappa_f}{\mu_j^2} - \frac{\kappa_{mdc}\theta}{\kappa_J^2+\mu_{\min}}\xi_j^2 = \left(\frac{\kappa_1+\kappa_f}{\xi_j\mu_j} - \frac{\kappa_{mdc}\theta}{\kappa_J^2+\mu_{\min}}\xi_j\mu_j\right)\frac{\xi_j}{\mu_j}.
\end{aligned}$$

Using (3.21), we get $\xi_j\mu_j \geq \kappa_2$, hence

$$(f+h)(x_j+s_j) - (f+h)(x_j) \leq \left(\frac{\kappa_1+\kappa_f}{\kappa_2} - \frac{\kappa_{mdc}\theta}{\kappa_J^2+\mu_{\min}}\kappa_2\right)\frac{\xi_j}{\mu_j} = -\frac{\kappa_1+\kappa_f}{2\kappa_2}\frac{\xi_j}{\mu_j}.$$

□

The next result is a consequence of Lemma 2, which shows a decrease on $f+h$ with respect to the exact stationarity measure.

Lemma 3. *Let Assumptions 1 to 5 hold for a realization of Algorithm 2, and consider iteration j. If the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate and*

$$\mu_j \geq \Big(\kappa_\xi + \kappa_2\Big)\sqrt{\frac{\nu_j}{\xi_{cp}^*(x_j;\nu_j^{-1})}}, \tag{3.23}$$

where κ_2 and κ_ξ are defined in Lemma 2 and Definition 21 respectively. Then the trial step s_j satisfies

$$(f+h)(x_j+s_j) - (f+h)(x_j) \leq -C_1\frac{1}{\mu_j}\nu_j^{-1/2}\xi_{\text{cp}}^*(x_j,\nu_j^{-1})^{1/2}, \quad \textit{with} \quad C_1 \coloneqq \frac{\kappa_2\kappa_3}{\kappa_\xi+\kappa_2}. \tag{3.24}$$

Proof. Using (3.23) and the fact that the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate, we have

$$\frac{\kappa_\xi + \kappa_2}{\mu_j} \leq \nu_j^{-1/2} \xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2} = \nu_j^{-1/2} \xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2} - \xi_j + \xi_j \leq \frac{\kappa_\xi}{\mu_j} + \xi_j. \tag{3.25}$$

Hence, $\mu_j \geq \frac{\kappa_2}{\xi_j}$ and $\xi_j \geq \frac{\kappa_2}{\kappa_\xi + \kappa_2} \nu_j^{-1/2} \xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2}$. Thus, by using Lemma 2, we get

$$(f+h)(x_j + s_j) - (f+h)(x_j) \leq -\kappa_3 \frac{\xi_j}{\mu_j} \leq -\frac{\kappa_2 \kappa_3}{\kappa_\xi + \kappa_2} \frac{\nu_j^{-1/2} \xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2}}{\mu_j}.$$

□

The next lemma guarantees a least decrease of the objective function for successful iterations.

Lemma 4. *Let Assumptions 1 to 5 hold. For a given realization of Algorithm 2, let j be the index of a successful iteration. Assuming that (f_j^0, f_j^+) is ε_f-accurate and*

$$\eta_3 > \sqrt{\frac{2\varepsilon_f(\kappa_J^2 + \mu_{\min})}{\eta_2 \kappa_{mdc} \theta}}, \tag{3.26}$$

we have

$$(f+h)(x_j + s_j) - (f+h)(x_j) \leq -\frac{C_2}{\mu_j^2}, \tag{3.27}$$

where $C_2 \coloneqq \eta_2 \eta_3^2 \kappa_{mdc} \theta (\kappa_J^2 + \mu_{\min})^{-1} - 2\varepsilon_f > 0$.

Proof. Let j be the index of a successful iteration, then $\rho_j \geq \eta_2$, hence,

$$f_j^+ + h(x_j + s_j) - (f_j^0 + h(x_j)) \leq -\eta_2(\varphi(0; x_j) + \psi(0; x_j) - (\varphi(s; x_j) + \psi(s_j; x_j))).$$

Thus, as the estimated residuals are ε_f-accurate, we get

$$\begin{aligned}
(f+h)(x_j + s_j) - (f+h)(x_j) &= f(x_j + s_j) - f_j^+ + f_j^+ + h(x_j + s_j) - f_j^0 - h(x_j) - f(x_j) + f_j^0 \\
&\leq \frac{2\varepsilon_f}{\mu_j^2} - \eta_2(\varphi(0; x_j) + \psi(0; x_j) - (\varphi(s; x_j) + \psi(s_j; x_j))) \\
&\leq \frac{2\varepsilon_f}{\mu_j^2} - \eta_2 \kappa_{mdc} \xi_{cp}(x_j; \nu_j^{-1}) \\
&\leq \frac{2\varepsilon_f}{\mu_j^2} - \frac{\eta_2 \kappa_{mdc} \theta}{\kappa_J^2 + \mu_{\min}} \xi_j^2.
\end{aligned}$$

As iteration j is successful, $\mu_j \xi_j \geq \eta_3$. Then,

$$(f+h)(x_j+s_j) - (f+h)(x_j) \leq -\left(\frac{\eta_2 \eta_3^2 \kappa_{mdc} \theta}{\kappa_J^2 + \mu_{\min}} - 2\varepsilon_f\right) \frac{1}{\mu_j^2} = -\frac{C_2}{\mu_j^2}.$$

Note that by (3.26), $C_2 > 0$. □

Lemma 5. *Let Assumptions 1 to 5 hold. Consider a realization of an iteration j of Algorithm 2. Assume that the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate, (f_j^0, f_j^+) is ε_f-accurate, and*

$$\mu_j \geq \frac{\kappa_4}{\xi_j}, \quad \textit{with} \quad \kappa_4 \coloneqq \max\left\{\sqrt{\frac{(\kappa_1 + 2\varepsilon_f + \kappa_f)(\kappa_J^2 + \mu_{\min})}{(1-\eta_2)\kappa_{mdc}\theta}}, \eta_3\right\}. \tag{3.28}$$

Then, the iteration j is successful, i.e., $\rho_j \geq \eta_2$ and $\xi_j \geq \frac{\eta_3}{\mu_j}$.

Proof. Using Lemma 1, and the fact that the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate and f_j^+ is ε_f-accurate,

$$\begin{aligned}
\left|1-\rho_j\right| &= \left|1 - \frac{f_j^0 + h(x_j) - f_j^+ - h(x_j+s_j)}{\varphi(0;x_j) + \psi(0;x_j) - \varphi(s_j;x_j) - \psi(s_j;x_j)}\right| \\
&= \frac{\left|\varphi(0;x_j) + \psi(0;x_j) - \varphi(s_j;x_j) - \psi(s_j;x_j) - f_j^0 - h(x_j) + f_j^+ + h(x_j+s_j)\right|}{|\varphi(0;x) + \psi(0;x) - \varphi(s;x) - \psi(s;x)|} \\
&= \frac{\left|h(x_j) - \varphi(s_j;x_j) - \psi(s_j : x_j) + f_j^+ + f_j - f_j^0\right|}{|\varphi(0;x) + \psi(0;x) - \varphi(s;x) - \psi(s;x)|} \\
&\leq \frac{\frac{\kappa_1}{\mu_j^2} + \frac{2\varepsilon_f}{\mu_j^2} + \frac{\kappa_f}{\mu_j^2}}{\kappa_{mdc}\xi_{j,cp}(x_j, \nu_j^{-1})} \\
&\leq \frac{\kappa_1 + 2\varepsilon_f + \kappa_f}{\kappa_{mdc}\theta(\kappa_J^2 + \mu_{\min})^{-1}} \frac{1}{\mu_j^2 \xi_j^2}.
\end{aligned}$$

Thus, by using (3.28), we get $\left|1-\rho_j\right| \leq 1-\eta_2$, hence $\rho_j \geq \eta_2$. Since (3.28) implies also $\xi_j \geq \frac{\eta_3}{\mu_j}$, we conclude that the iteration j is successful. □

3.2.3 Probabilistic useful results

We now formulate assumptions regarding the probabilistic properties satisfied by our method. We formulate here a similar variance condition as from [56, Assumption 2.4] using our notations.

Assumption 6. *Considering $p \in (0,1]$ and $q \in (0,1]$, we assume that*

(i) the random sequence $\{(F_j, G_j, \xi_j)\}$ is p-probabilistically $\{\kappa_f, \kappa_g, \kappa_\xi\}$-first-order accurate for $\kappa_f, \kappa_g, \kappa_\xi > 0$;

(ii) the sequence of random function estimates $\{(F_j^0, F_j^+)\}$ is q-probabilistically ε_f-accurate for $\varepsilon_f > 0$;

(iii) there exists $\kappa_v > 0$ such that

$$\mathbb{E}\left[|f(X_j) - F_j^0|^2 \mid \mathcal{F}_{j-1}\right] \leq \frac{\kappa_v^2}{M_j^4}(1-pq), \tag{3.29}$$

$$\mathbb{E}\left[|f(X_j + S_j) - F_j^+|^2 \mid \mathcal{F}_{j-1/2}\right] \leq \frac{\kappa_v^2}{M_j^4}(1-pq). \tag{3.30}$$

Finally, we formulate an assumption on η_3 for convergence purposes.

Assumption 7. *The constant η_3 is chosen such as*

$$\eta_3 \;\geq\; \max\left\{\sqrt{\frac{2\varepsilon_f(\kappa_J^2 + \mu_{\min})}{\eta_2 \kappa_{mdc}\theta}}, \quad \sqrt{\frac{3(\kappa_f + \kappa_{efs})(\kappa_J^2 + \mu_{\min})}{2\kappa_{mdc}\theta}}\right\}. \tag{3.31}$$

The next lemma holds almost immediately using (3.29) and Hölder's inequality.

Lemma 6. *Let (iii) from Assumption 6 hold and consider that $pq \in (0,1]$. Then,*

$$\mathbb{E}\left[\left|f(X_j) - F_j^0\right| \mid \mathcal{F}_{j-1}\right] < \frac{\kappa_v}{M_j^2} \quad \textit{and} \quad \mathbb{E}\left[\left|f(X_j + S_j) - F_j^+\right| \mid \mathcal{F}_{j-1/2}\right] < \frac{\kappa_v}{M_j^2}. \tag{3.32}$$

Proof. Using Hölder's inequality and (3.29),

$$\mathbb{E}\left[\frac{\left|f(X_j) - F_j^0\right|}{\kappa_v / M_j^2} \mid \mathcal{F}_{j-1}\right] \leq \mathbb{E}\left[1 \mid \mathcal{F}_{j-1}\right]^{\frac{1}{2}} \mathbb{E}\left[\left|\frac{f(X_j) - F_j^0}{\kappa_v / M_j^2}\right|^2 \mid \mathcal{F}_{j-1}\right]^{\frac{1}{2}} \leq \sqrt{1-pq} < 1.$$

We use the same arguments to get the second inequality, replacing $f(X_j)$, F_j^0 and $\mathcal{F}_{j-1}$ by $f(X_j + S_j)$, F_j^+ and $\mathcal{F}_{j-1/2}$ respectively. □

We have consequently the following lemma.

Lemma 7. *Let (iii) from Assumption 6 hold. Assume that the iteration j of Algorithm 2 is successful. Then,*

$$\mathbb{E}\left[(f+h)(X_j+S_j) - (f+h)(X_j) \mid \mathcal{F}_{j-1/2}\right] \leq \frac{2\kappa_v}{M_j^2}. \tag{3.33}$$

Proof. Using the fact that $\mathcal{F}_{j-1} \subseteq \mathcal{F}_{j-1/2}$ and Lemma 6, we have

$$\begin{aligned}
&\mathbb{E}\left[f(X_j+S_j) + h(X_j+S_j) - f(X_j) - h(X_j) \mid \mathcal{F}_{j-1/2}\right] \\
&\leq \frac{2\kappa_v}{M_j^2} + \mathbb{E}\left[F_j^+ - F_j^0 + h(X_j+S_j) - h(X_j) \mid \mathcal{F}_{j-1/2}\right].
\end{aligned}$$

As iteration j is successful, using Assumption 4,

$$\begin{aligned}
&\mathbb{E}\left[F_j^+ - F_j^0 + h(X_j+S_j) - h(X_j) \mid \mathcal{F}_{j-1/2}\right] \\
&\leq -\eta_2 \mathbb{E}\left[\varphi(0;X_j) + \psi(0;X_j) - \varphi(S_j;X_j) - \psi(S_j;X_j) \mid \mathcal{F}_{j-1/2}\right] \\
&\leq -\eta_2 \kappa_{mdc} \mathbb{E}\left[\xi_{j,cp}(X_j, V_j^{-1}) \mid \mathcal{F}_{j-1/2}\right] \leq 0.
\end{aligned}$$

Hence,

$$\mathbb{E}\left[f(X_j+S_j) + h(X_j+S_j) - f(X_j) - h(X_j) \mid \mathcal{F}_{j-1/2}\right] \leq \frac{2\kappa_v}{M_j^2}.$$

□

Note that Lemma 7 still holds for any conditioning event as long as it is included in $\mathcal{F}_{j-1/2}$. For our complexity analysis, we will assume that $pq \neq 1$, since if $pq = 1$ then for all j we have

$$p_j^* \;=\; P(U_j \mid \mathcal{F}_{j-1}) \;=\; p \;=\; q_j^* \;=\; P(W_j \mid \mathcal{F}_{j-1}) \;=\; q \;=\; 1,$$

and the behavior of the algorithm reduces to that of a deterministic method.

3.3 Complexity analysis

3.3.1 Key theorem

Similarly to existing analyses, e.g., [8, 9, 12], for a given iteration j of Algorithm 2, we consider the following random variable

$$\Pi_j = \tau(f(X_j) + h(X_j)) + \frac{1-\tau}{M_j^2}, \tag{3.34}$$

for an appropriately chosen $\tau \in (0,1)$ that respects

$$\frac{\tau}{1-\tau} \;>\; \max\left\{4\frac{\left(\lambda^2-\frac{1}{\lambda^2}\right)}{C_1\zeta},\; \frac{\left(\lambda^2-\frac{1}{\lambda^2}\right)}{C_2},\; \frac{2\left(\lambda^2-\frac{1}{\lambda^2}\right)}{\kappa_f+\kappa_{efs}}\right\}, \tag{3.35}$$

where ζ is a parameter such that

$$\zeta \;\geq\; \kappa_\xi + \max\left\{\kappa_4,\; \kappa_J^2,\; \kappa_2\right\}. \tag{3.36}$$

In addition, the probabilities p and q are required to satisfy:

$$\frac{pq-\frac{1}{2}}{(1-p)(1-q)} \geq \frac{C_3}{C_1}, \quad \text{where} \quad C_3 \coloneqq \tfrac{2\kappa_v}{\zeta}. \tag{3.37}$$

The bound (3.37) demands so that $pq \geq \frac{1}{2}$. Moreover,

$$(1-p)(1-q) \leq \frac{(1-\tau)\left(1-\frac{1}{\lambda^2}\right)}{4(\tau C_3\zeta + (1-\tau)(\lambda^2-1))}. \tag{3.38}$$

Denote conditions (3.37) and (3.38) are satisfied for p and q sufficiently close to 1. The main task in deriving our convergence and complexity result consists in proving the following theorem.

Theorem 5. *Let Assumptions 1 to 7 hold where p and q are chosen respecting* (3.35)–(3.38). *Then, there exists $\gamma > 0$ such that, for all j, one has*

$$\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid \mathcal{F}_{j-1/2}\right] \;\leq\; -\frac{\gamma}{M_j^2}, \tag{3.39}$$

where the expectation is taken over the product trace σ-algebra generated by all models and function value estimates. We point out that the right-hand side is measurable relative to $\mathcal{F}_{j-1/2}$.

Proof. The proof focuses on a realization of the process Π_j, and divides the iterations into two subsets, depending on whether the following condition holds:

$$\nu_j^{-1/2}\xi^*_{\text{cp}}(x_j,\nu_j^{-1})^{1/2} \geq \frac{\zeta}{\mu_j}. \tag{3.40}$$

This condition is strongly related to the requirements on μ_j in the previous lemmas.

Consider a realization of Algorithm 2, and let π_j be the corresponding realization of Π_j. If j

is the index of a successful iteration, then $x_{j+1} = x_j + s_j$, and $\mu_{j+1} \geq \frac{\mu_j}{\lambda}$. Then

$$\pi_{j+1} - \pi_j \;\leq\; \tau((f+h)(x_{j+1}) - (f+h)(x_j)) + (1-\tau)\frac{\lambda^2 - 1}{\mu_j^2}. \tag{3.41}$$

If j is the index of an unsuccessful iteration, $x_{j+1} = x_j$ and $\mu_{j+1} = \lambda\mu_j$, leading to

$$\pi_{j+1} - \pi_j \;=\; \frac{1-\tau}{\mu_j^2}\left(\frac{1}{\lambda^2} - 1\right) < 0. \tag{3.42}$$

For both types of iterations, we will consider four possible outcomes, involving the quality of the model and the estimates.

Case 1: $V_j^{-1/2}\xi^*_{cp}(X_j, V_j^{-1})^{1/2} \geq \frac{\zeta}{M_j}$.

1. **Both m and (f_j^0, f_j^+) are accurate.** Because the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate, using (3.36),

$$\xi_j \geq \nu_j^{-1/2}\xi^*_{\rm cp}(x_j, \nu_j^{-1})^{1/2} - \frac{\kappa_\xi}{\mu_j} \geq \frac{\zeta - \kappa_\xi}{\mu_j} \geq \frac{\kappa_4}{\mu_j}.$$

   Therefore (3.28) holds. Since the estimates are also accurate, the iteration is necessarily successful by Lemma 5. Moreover,

$$\nu_j^{-1/2}\xi^*_{\rm cp}(x_j, \nu_j^{-1})^{1/2} \;\geq\; \frac{\zeta}{\mu_j} \;\geq\; \left(\kappa_\xi + \kappa_2\right)\tfrac{1}{\mu_j},$$

   where κ_2 is defined in Lemma 3. So the condition (3.23) is satisfied and, by Lemma 3, we have (3.24) and it leads to

$$\pi_{j+1} - \pi_j \leq -\tau C_1 \frac{1}{\mu_j}\nu_j^{-1/2}\xi^*_{\rm cp}(x_j, \nu_j^{-1})^{1/2} + (1-\tau)(\lambda^2 - 1)\frac{1}{\mu_j^2}. \tag{3.43}$$

2. **Only m is accurate.** From Lemma 3, (3.24) is still valid. Therefore, if the iteration is successful, then (3.43) holds. We also have using (3.35) and $\lambda \geq 1$

$$\begin{aligned}
\pi_{j+1} - \pi_j &\leq -\tau C_1 \frac{1}{\mu_j}\nu_j^{-1/2}\xi^*_{\rm cp}(x_j, \nu_j^{-1})^{1/2} + (1-\tau)(\lambda^2 - 1)\frac{1}{\mu_j^2} \\
&\leq \frac{1-\tau}{\mu_j^2}\left(\frac{4}{\lambda^2} - 4\lambda^2 + \lambda^2 - 1\right) \\
&\leq \frac{1-\tau}{\mu_j^2}\left(\frac{1}{\lambda^2} - 1\right) < 0.
\end{aligned}$$

As we know that (3.42) also holds whenever iteration j is unsuccessful, then, whether iteration j is successful or not, we have

$$\pi_{j+1} - \pi_j \leq \frac{1-\tau}{\mu_j^2}\left(\frac{1}{\lambda^2} - 1\right). \tag{3.44}$$

3. **Only (f_j^0, f_j^+) is accurate.** If the iteration is unsuccessful, then we have (3.42). Otherwise, by Lemma 4 we have (3.27). Hence,

$$\begin{aligned} \pi_{j+1} - \pi_j &\leq \left[-\tau C_2 + (1-\tau)(\lambda^2 - 1)\right] \tfrac{1}{\mu_j^2} \\ &\leq (1-\tau)\left(\frac{1}{\lambda^2} - 1\right) \tfrac{1}{\mu_j^2}, \end{aligned}$$

where we used (3.35) to obtain the last inequality. Thus, (3.44) also holds if the iteration is successful.

4. **Both m and (f_j^0, f_j^+) are inaccurate.** We treat this last case later in the proof.

Summarizing the four cases, we have that the bound (3.43) on $\pi_{j+1} - \pi_j$ holds for Case 1 and the bound (3.44) holds for Cases 2 and 3. Denoting $E_j = \mathcal{F}_{j-1/2} \cap \left\{ V_j^{-1/2} \xi_{cp}^*(X_j, V_j^{-1})^{1/2} \geq \frac{\zeta}{M_j} \right\}$ and A_j the event "m and (r_j^0, r_j^s) are accurate", we have

$$\begin{aligned} &\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j\right] \\ &\leq pq\left(-\tau C_1 \frac{1}{M_j} V_j^{-1/2} \xi_{cp}^*(X_j, V_j^{-1})^{1/2} + \frac{(1-\tau)(\lambda^2-1)}{M_j^2}\right) \\ &+ ((1-p)q + p(1-q))\left(\frac{1-\tau}{M_j^2}\left(\frac{1}{\lambda^2} - 1\right)\right) \\ &+ (1-p)(1-q)\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j \cap A_j^c\right]. \end{aligned}$$

We have using Lemma 7 that

$$\begin{cases} \mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j \cap A_j^c\right] \leq \frac{1-\tau}{M_j^2}\left(\frac{1}{\lambda^2} - 1\right) & \text{if iteration } j \text{ is unsuccessful} \\ \mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j \cap A_j^c\right] \leq \frac{1-\tau}{M_j^2}(\lambda^2 - 1) + \tau \frac{2\kappa_v}{\mu_j^2} & \text{otherwise.} \end{cases}$$

As $\frac{1-\tau}{\mu_j^2}(\frac{1}{\lambda^2} - 1) \leq \frac{1-\tau}{\mu_j^2}(\lambda^2 - 1)$ and $\tau \frac{2\kappa_v}{\mu_j^2} \geq 0$, then, whether iteration j is successful or not, we have

$$\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j \cap A_j^c\right] \leq \frac{(1-\tau)}{M_j^2}(\lambda^2 - 1) + \tau \frac{2\kappa_v}{M_j^2}. \tag{3.45}$$

Then,

$$\begin{aligned}\mathbb{E}\left[\Pi_{j+1}-\Pi_j \mid E_j\cap A_j^c\right] &\le 2\tau\frac{\kappa_v}{M_j^2}+\frac{1-\tau}{M_j^2}(\lambda^2-1)\\ &\le \frac{\tau}{M_j}\frac{2\kappa_v}{\zeta}V_j^{-1/2}\xi_{cp}^*(X_j,V_j^{-1})^{1/2}+(1-\tau)(\lambda^2-1)\frac{1}{M_j^2}\\ &= \tau\frac{C_3}{M_j}V_j^{-1/2}\xi_{cp}^*(X_j,V_j^{-1})^{1/2}+(1-\tau)(\lambda^2-1)\frac{1}{M_j^2}.\end{aligned}$$

Following the lines of [9, Theorem 4.14] for the rest of the proof, we have then

$$\begin{aligned}&\mathbb{E}\left[\Pi_{j+1}-\Pi_j \mid E_j\right]\\ &\le pq\left(-\tau C_1V_j^{-1/2}\xi_{cp}^*(X_j,V_j^{-1})^{1/2}\frac{1}{M_j}+\frac{(1-\tau)(\lambda^2-1)}{M_j^2}\right)\\ &+((1-p)q+p(1-q))\left(\frac{1-\tau}{M_j^2}\left(\frac{1}{\lambda^2}-1\right)\right)\\ &+(1-p)(1-q)\left(\tau C_3\frac{1}{M_j}V_j^{-1/2}\xi_{cp}^*(X_j,V_j^{-1})^{1/2}+(1-\tau)(\lambda^2-1)\frac{1}{M_j^2}\right)\\ &=(-C_1pq+(1-p)(1-q)C_3)\frac{\tau}{M_j}V_j^{-1/2}\xi_{cp}^*(X_j,V_j^{-1})^{1/2}\\ &+(pq-\tfrac{1}{\lambda^2}(p(1-q)+(1-p)q)+(1-p)(1-q))(1-\tau)(\lambda^2-1)\frac{1}{M_j^2}\\ &\le(-C_1pq+(1-p)(1-q)C_3)\frac{\tau}{M_j}V_j^{-1/2}\xi_{cp}^*(X_j,V_j^{-1})^{1/2}+(1-\tau)(\lambda^2-1)\frac{1}{M_j^2},\end{aligned}$$

where the last line uses

$$pq-\tfrac{1}{\lambda^2}(p(1-q)+(1-p)q)+(1-p)(1-q)\le(p+(1-p))(q+(1-q))=1.$$

Since p and q are chosen such that

$$\frac{pq-\frac{1}{2}}{(1-p)(1-q)}\ge\frac{C_3}{C_1} \tag{3.46}$$

holds and $\lambda\ge1$, by combining (3.46) and (3.35),

$$-C_1pq+(1-p)(1-q)C_3\le-\frac{C_1}{2}\le-2\frac{(1-\tau)(\lambda^2-\frac{1}{\lambda^2})}{\tau\zeta}\le-2\frac{(1-\tau)(\lambda^2-1)}{\tau\zeta}. \tag{3.47}$$

Since $V_j^{-1/2}\xi^*_{cp}(X_j, V_j^{-1})^{1/2} \geq \frac{\zeta}{M_j}$,

$$\frac{(1-\tau)(\lambda^2-1)}{M_j^2} \;\leq\; -\tfrac{1}{2}\left(-C_1pq + (1-p)(1-q)C_3\right)\frac{\tau}{M_j}V_j^{-1/2}\xi^*_{cp}(X_j, V_j^{-1})^{1/2}.$$

Using (3.47), it leads to

$$\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j\right] \;\leq\; -\frac{1}{4}C_1\frac{\tau}{M_j}V_j^{-1/2}\xi^*_{cp}(X_j, V_j^{-1})^{1/2} \leq -\frac{C_1\tau\zeta}{4M_j^2}.$$

Using (3.47), we finally have

$$\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j\right] \;\leq\; -\frac{(1-\tau)\left(\lambda^2-1\right)}{M_j^2} \leq -\frac{(1-\tau)\left(1-\frac{1}{\lambda^2}\right)}{4M_j^2}. \tag{3.48}$$

Case 2: $V_j^{-1/2}\xi^*_{cp}(X_j, V_j^{-1})^{1/2} < \frac{\zeta}{M_j}$.

Whenever $\xi_j < \frac{\eta_3}{M_j}$, iteration j is necessarily unsuccessful and (3.44) holds. We thus assume in what follows that $\xi_j \geq \frac{\eta_3}{M_j}$, and consider again four cases.

1. **Both m and (f_j^0, f_j^+) are accurate.** Unlike in the previous case, it is now possible for the iteration to be unsuccessful and in that case we have (3.44). Otherwise, if the iteration is successful, we can use (3.27) from Lemma 4. We can thus apply the same reasoning as in Case 1.3, which implies that (3.44) also holds when the iteration is successful.

2. **Only m is accurate.** If the iteration is unsuccessful, it is clear that (3.44) holds. Otherwise, we apply the same argument as in the proof of Lemma 4. Since the model is $(\kappa_f, \kappa_g, \kappa_\xi)$-first-order accurate, using Lemma 1,

$$\begin{aligned}
&f(x_j) + h(x_j) - f(x_j + s_j) - h(x_j + s_j) \\
&\quad = f(x_j) - f_j + f_j + h(x_j) - \varphi(s_j; x_j) - \psi(s_j; x_j) \\
&\qquad +\varphi(s_j; x_j) + \psi(s_j; x_j) - f(x_j + s_j) - h(x_j + s_j) \\
&\quad \geq -\frac{\kappa_f}{\mu_j^2} + \kappa_{mdc}\frac{\xi_j^2\theta}{\kappa_J^2 + \mu_{min}} - \frac{\kappa_{efs}}{\mu_j^2} \\
&\quad \geq -\frac{\kappa_f}{\mu_j^2} + \kappa_{mdc}\frac{\eta_3^2\theta}{(\kappa_J^2 + \mu_{min})\mu_j^2} - \frac{\kappa_{efs}}{\mu_j^2} \\
&\quad \geq \left(\kappa_{mdc}\eta_3^2\theta(\kappa_J^2 + \mu_{min})^{-1} - (\kappa_f + \kappa_{efs})\right)\frac{1}{\mu_j^2} \;\geq\; \frac{(\kappa_f + \kappa_{efs})}{2}\frac{1}{\mu_j^2},
\end{aligned}$$

where the last line comes from (3.31). Using (3.35), we have

$$\begin{aligned} \pi_{j+1} - \pi_j &\leq \left[-\tau\frac{\kappa_f + \kappa_{efs}}{2} + (1-\tau)(\lambda^2 - 1)\right]\tfrac{1}{\mu_j^2} \\ \iff \pi_{j+1} - \pi_j &\leq (1-\tau)\left(\frac{1}{\lambda^2} - 1\right)\tfrac{1}{\mu_j^2}. \end{aligned} \tag{3.49}$$

3. **Only (f_j^0, f_j^+) is accurate.** We can reason as in Case 1 to show that (3.44) holds regardless if the iteration is successful or unsuccessful.

4. **Both m and (f_j^0, f_j^+) are inaccurate.** As explained earlier and denoting $E_j' = \mathcal{F}_{j-1/2} \cap \left\{V_j^{-1/2}\xi_{cp}^*(X_j, V_j^{-1})^{1/2} < \frac{\zeta}{M_j}\right\}$, we still have (3.45). Since $\zeta C_3 = 2\kappa_v$,

$$\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j'\right] \leq \tau\frac{\zeta C_3}{M_j^2} + \frac{(1-\tau)(\lambda^2-1)}{M_j^2}. \tag{3.50}$$

Thus, (3.44) is satisfied for each subcase except for Case 4 where (3.50) holds. Then we have

$$\begin{aligned} & \mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j'\right] \\ \leq\ & [pq + p(1-q) + q(1-p)](1-\tau)\left(\frac{1}{\lambda^2} - 1\right)\frac{1}{M_j^2} \\ +\ & (1-p)(1-q)\left[\tau C_3\zeta + (1-\tau)(\lambda^2-1)\right]\frac{1}{M_j^2} \\ \leq\ & -pq(1-\tau)\left(1 - \frac{1}{\lambda^2}\right)\frac{1}{M_j^2} + (1-p)(1-q)\left[\tau C_3\zeta + (1-\tau)(\lambda^2-1)\right]\frac{1}{M_j^2}. \end{aligned}$$

As p and q have been chosen such that $pq \geq \frac{1}{2}$ and

$$(1-p)(1-q) \leq \frac{(1-\tau)\left(1 - \frac{1}{\lambda^2}\right)}{4(\tau C_3\zeta + (1-\tau)(\lambda^2-1))} \tag{3.51}$$

holds, using (3.51), we have

$$\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid E_j'\right] \leq -\frac{(1-\tau)\left(1 - \frac{1}{\lambda^2}\right)}{4}\frac{1}{M_j^2}, \tag{3.52}$$

which is the same bound as in (3.48). Let $\gamma \coloneqq \frac{(1-\tau)\left(1-\frac{1}{\lambda^2}\right)}{4} > 0$, we have then established that for every iteration j,

$$\mathbb{E}\left[\Pi_{j+1} - \Pi_j \mid \mathcal{F}_{j-1/2}\right] < -\frac{\gamma}{M_j^2},$$

which concludes the proof. □

From this theorem, we have the following corollary.

Corollary 1. *Let Assumptions 1 to 7 hold with probabilities p and q chosen as for Theorem 5. Assume also that f and h are bounded from below. Then,*

$$\mathbf{P}\left(\sum_{j=0}^{\infty} \frac{1}{M_j^2} \;<\; +\infty\right) \;=\; 1. \tag{3.53}$$

From Corollary 1, we have

$$\mathbf{P}\left(\frac{1}{M_j} \to 0\right) = 1 \iff \mathbf{P}\left(M_j \to \infty\right) = 1. \tag{3.54}$$

Following the lines of the proof of [27, Theorem 4.16], Algorithm 2 generates $\left(X_j, V_j\right)$ sequences which almost surely satisfy

$$\liminf_{j\to\infty} V_j^{-1/2} \xi_{cp}^*(X_j, V_j^{-1})^{1/2} = 0. \tag{3.55}$$

3.3.2 Complexity bound

General result

In this section, we introduce the necessary tools to derive the convergence rate from a general result established by the stochastic trust-region algorithm in [12]. Given a stochastic process $\{X_j\}$, we say that T is a *stopping time* for $\{X_j\}$, if, for all $j \geq 1$, the event $\{T \leq j\}$ belongs to the σ-algebra associated with $X_1, X_2, \ldots, X_j$.

Let Λ_j be a birth and death stochastic process, defined on the same probability space as $\{\Pi_j, \Delta_j\}_{j\geq 0}$, with $\Lambda_0 = 1$ and, for all $j \geq 0$

$$\begin{cases} \mathbf{P}(\Lambda_{j+1} = 1 \mid \mathcal{F}_j) & = \beta, \\ \mathbf{P}(\Lambda_{j+1} = -1 \mid \mathcal{F}_j) & = 1 - \beta, \end{cases} \tag{3.56}$$

where $\mathcal{F}_j$ is the σ-algebra generated by $(\Pi_0, \Delta_0, \Lambda_0), \ldots, (\Pi_j, \Delta_j, \Lambda_j)$. Let $\{T_{\varepsilon'}\}_{\varepsilon'\geq 0}$ be a family of stopping times with respect to $\mathcal{F}_j$. Then, we have the following theorem from [12].

Theorem 6. *[12, Theorem 2] Consider the following assumptions*

i.) There exists a constant $\hat{\gamma} > 0$ *such that there exists a* $\Delta_{max} = \Delta_0 e^{\hat{\gamma} j_{max}}$ *where* $j_{max} \in \mathbb{Z}$ *and such that, for all* $j,\ \Delta_j \leq \Delta_{max}$.

ii.) There exists a constant $\Delta_{\varepsilon'} = \Delta_0 e^{\hat{\gamma} j_{\varepsilon'}}$ *where* $j_{\varepsilon'} \in \mathbb{Z}_-$ *and, for all* $j \geq 0$,

$$\mathbf{1}_{\{\mathbf{T}_{\varepsilon'} > \mathbf{j}\}} \Delta_{j+1} \geq \mathbf{1}_{\{\mathbf{T}_{\varepsilon'} > \mathbf{j}\}} \min\left\{\Delta_{\varepsilon'}, \Delta_j e^{\hat{\gamma}\Lambda_{j+1}}\right\},$$

where Λ_{j+1} *satisfies* (3.56) *with* $\beta > \frac{1}{2}$.

iii.) There exists a non-decreasing function g *and a constant* $\Theta > 0$ *such that*

$$\mathbf{1}_{\{\mathbf{T}_{\varepsilon'} > \mathbf{j}\}} \mathbb{E}\left[\Pi_{j+1} \mid \mathcal{F}_j\right] \leq \mathbf{1}_{\{\mathbf{T}_{\varepsilon'} > \mathbf{j}\}} \left(\Pi_j - \Theta g\left(\Delta_j\right)\right).$$

Then we have

$$\mathbb{E}\left[T_{\varepsilon'}\right] \leq \frac{\beta}{2\beta - 1} \frac{\Pi_0}{\Theta g(\Delta_{\varepsilon'})} + 1. \tag{3.57}$$

Convergence rate for stochastic nonsmooth regularized LM

The result given in Section 3.3.2 is a general complexity bound satisfied for any stopping time $T_{\varepsilon'}$ respecting the filtering of $\mathcal{F}_j$. We will then prove that we can construct a stochastic process such that Algorithm 2 ensures $V_j^{-1/2} \xi^*_{cp}(X_j, V_j^{-1})^{1/2} \leq \varepsilon$ for a given $\varepsilon > 0$.

Let $\varepsilon > 0$, we define a random time T_ε by

$$T_\varepsilon \coloneqq \inf\left\{j \geq 0 \mid V_j^{-1/2} \xi^*_{cp}(X_j, V_j^{-1})^{1/2} \leq \varepsilon\right\},$$

where T_ε is a stopping time for Algorithm 2 with respect to the σ-algebra $\mathcal{F}_{j-1/2}$. We will prove the assumptions of *Theorem* 6 are satisfied with $\Delta_j = \frac{1}{M_j}$. Hence, the assumptions can be alternatively written as follows

i.) There exists a constant $\hat{\gamma} > 0$ such that there exists a $M_{min} = M_0 e^{-\hat{\gamma} j_{max}}$ where $j_{max} \in \mathbb{Z}$ and such that, $\forall j,\ M_j \geq M_{min}$.

ii.) There exists a constant $M_\varepsilon = M_0 e^{-\hat{\gamma} j_\varepsilon}$ where $j_\varepsilon \in \mathbb{Z}_-$ and

$$\mathbf{1}_{\{\mathbf{T}_\varepsilon > \mathbf{j}\}} M_{j+1} \leq \mathbf{1}_{\{\mathbf{T}_\varepsilon > \mathbf{j}\}} \max\left\{M_\varepsilon, M_j e^{-\hat{\gamma}\Lambda_j}\right\},$$

where Λ_j satisfies (3.56) with $\beta > \frac{1}{2}$ and $\mathcal{F}_j = \mathcal{F}_{j-1/2}$.

iii.) There exists a non-decreasing function g and a constant $\Theta > 0$ such that :

$$\mathbf{1}_{\{\mathbf{T}_\varepsilon > \mathbf{j}\}} \mathbb{E}\left[\Pi_{j+1} \mid \mathcal{F}_{j-1/2}\right] \leq \mathbf{1}_{\{\mathbf{T}_\varepsilon > \mathbf{j}\}} \left(\Pi_j - \Theta g\left(\frac{1}{M_j}\right)\right).$$

Let $M_\varepsilon = \zeta/\varepsilon$; assuming $\mu_0 = M_\varepsilon/\lambda^s$ and $\mu_{min} = M_\varepsilon/\lambda^t$ where $s, t > 0$ are integers, we have $M_j = M_\varepsilon/\lambda^k$ for some integer k. Hence, we can choose μ_{min} such that $\mu_{min} = \mu_0 \lambda^{s-t}$ where s is the smallest integer where ζ satisfies (3.36). Therefore, let $j_{max} = t - s \in \mathbb{Z}$ and,

$$\begin{aligned} & M_{min} = M_0 \lambda^{-j_{max}} \\ \iff \; & M_{min} = M_0 e^{\log(\lambda)(-j_{max})}. \end{aligned}$$

As $\lambda > 1$, let $\hat{\gamma} = \log(\lambda) > 0$. So there exists a constant $\hat{\gamma} > 0$ such that $M_{min} = M_0 e^{-\hat{\gamma} j_{max}}$ where $j_{max} \in \mathbb{Z}$ and $M_j \geq M_{min}$ for all $j \geq 0$. So, (i) is satisfied.

Considering the event S_j: "Iteration j is successful", we have the following lemma.

Lemma 8. *Let Assumptions 1 to 7 hold and choosing p and q such that $pq > \frac{1}{2}$. For all $j < T_\varepsilon$, assuming also $M_j \geq M_\varepsilon$, we have*

$$M_{j+1} = M_j e^{-\hat{\gamma}\Lambda_j}, \tag{3.58}$$

where $\hat{\gamma} = \log(\lambda)$, $\mathbf{1}_{S_j}$ equals 1 if iteration j is successful and 0 otherwise and $\Lambda_j = 2\mathbf{1}_{S_j} - 1$ is a birth-and-death process satisfying

$$\beta = \mathbf{P}(\Lambda_j = 1 \mid \mathcal{F}_{j-1/2}, M_j \geq M_\varepsilon) = 1 - \mathbf{P}(\Lambda_j = -1 \mid \mathcal{F}_{j-1/2}, M_j \geq M_\varepsilon) \geq pq > \tfrac{1}{2}.$$

Proof. By the mechanism of Algorithm 2, we have

$$M_{j+1} = M_j \frac{1}{\lambda} \mathbf{1}_{S_j} + M_j \lambda (1 - \mathbf{1}_{S_j}). \tag{3.59}$$

Let $\hat{\gamma} = \log(\lambda)$ and $\Lambda_j = 2\mathbf{1}_{S_j} - 1$, then

$$\begin{aligned} e^{-\hat{\gamma}\Lambda_j} &= e^{-\log(\lambda)(2\mathbf{1}_{S_j} - 1)} \\ &= e^{-\log(\lambda)(\mathbf{1}_{S_j} - 1) - \log(\lambda)\mathbf{1}_{S_j}} \\ &= e^{\log(\lambda)(1 - \mathbf{1}_{S_j}) + \log(1/\lambda)\mathbf{1}_{S_j}}. \end{aligned}$$

Consequently,

$$\begin{cases} \text{If} \quad \mathbf{1}_{S_j} = 1, & e^{-\hat{\gamma}\Lambda_j} = e^{\log(1/\lambda)} = \frac{1}{\lambda}, \\ \text{Else}, & e^{-\hat{\gamma}\Lambda_j} = e^{\log(\lambda)} = \lambda. \end{cases}$$

Then, $e^{-\hat{\gamma}\Lambda_j} = \frac{1}{\lambda}\mathbf{1}_{S_j} + \lambda(1 - \mathbf{1}_{S_j})$. Thus,

$$M_{j+1} = M_j(\frac{1}{\lambda}\mathbf{1}_{S_j} + \lambda(1 - \mathbf{1}_{S_j})) = M_j e^{-\hat{\gamma}\Lambda_j}.$$

Moreover, if $\mu_j \geq M_\varepsilon$, as $j < T_\varepsilon$,

$$\nu_j^{-1/2}\xi^*_{\text{cp}}(x_j, \nu_j^{-1})^{1/2} > \varepsilon = \frac{\zeta}{M_\varepsilon} \geq \frac{\zeta}{\mu_j}.$$

Then, assuming both m and (f_j^0, f_j^+) are accurate, we have with the κ_ξ accuracy of ξ_j,

$$\xi_j > \nu_j^{-1/2}\xi^*_{\text{cp}}(x_j, \nu_j^{-1})^{1/2} - \frac{\kappa_\xi}{\mu_j} \geq \frac{1}{\mu_j}(\zeta - \kappa_\xi) \geq \frac{\kappa_4}{\mu_j}.$$

According to Lemma 5, iteration j is a success. Hence, we have

$$\mathbf{P}(\Lambda_j = 1 \mid \mathcal{F}_{j-1/2}, M_j \geq M_\varepsilon) \geq pq.$$

□

Lemma 9. *Let Assumptions 1 to 7 hold and choosing p and q such that $pq > \frac{1}{2}$. Then,* (ii) *is satisfied with $\Lambda_j = 2\mathbf{1}_{S_j} - 1$ and $\hat{\gamma} = \log(\lambda) > 0$.*

Proof. As (ii) is satisfied whenever $\mathbf{1}_{\{T_\varepsilon > j\}} = 0$, we assume $T_\varepsilon \leq j$.

- If $M_j < M_\varepsilon$, recalling that $M_j = M_\varepsilon/\lambda^k$, as $M_j < M_\varepsilon$, $k > 1$ and we have $M_j \leq M_\varepsilon/\lambda$. Hence $M_{j+1} \leq M_\varepsilon$ [9].
- If $M_j \geq M_\varepsilon$. As $T_\varepsilon > j$, we have from Lemma 8

$$M_{j+1} = M_j e^{-\hat{\gamma}\Lambda_j}, \tag{3.60}$$

  where $\hat{\gamma} = \log(\lambda) > 0$. Besides, $\mathbf{1}_{S_j}$ equals to 1 if iteration j is successful and $\Lambda_j = 2\mathbf{1}_{S_j} - 1$ is a birth-and-death process satisfying

$$\mathbf{P}(\Lambda_j = 1 \mid \mathcal{F}_{j-1/2}, M_j \geq M_\varepsilon) = 1 - \mathbf{P}(\Lambda_j = -1 \mid \mathcal{F}_{j-1/2}, M_j \geq M_\varepsilon) \geq pq > \tfrac{1}{2}.$$

Thus (ii) is satisfied. □

We finally have the following proposition to prove (iii) is satisfied for Algorithm 2.

Proposition 7. *Let Assumptions 1 to 7 hold. Assume as well we have p and q such that $pq > \frac{1}{2}$ satisfying*

$$\frac{pq - \frac{1}{2}}{(1-p)(1-q)} \geq \frac{C_3}{C_1},$$

and

$$(1-p)(1-q) \leq \frac{1}{4}\left(\frac{(1-\tau)(1-\frac{1}{\lambda^2})}{\tau C_3 \zeta + (1-\tau)(\lambda^2-1)}\right).$$

Then, for $g : x \mapsto x^2$ and $\Theta \coloneqq \gamma$ where $\gamma \coloneqq \frac{(1-\tau)\left(1-\frac{1}{\lambda^2}\right)}{4}$, (iii) is satisfied.

Proof. Let $g : x \mapsto x^2$ and $\Theta \coloneqq \gamma > 0$. As g is a non-decreasing function on $\mathbb{R}_+$ and all assumptions of Theorem 5 are respected, we have with Theorem 5

$$\mathbb{E}\left[\Pi_{j+1} \mid \mathcal{F}_{j-1/2}\right] \mathbf{1}_{T_\varepsilon > j} \leq \Pi_j \mathbf{1}_{T_\varepsilon > j} - \Theta g\left(\frac{1}{M_j}\right) \mathbf{1}_{T_\varepsilon > j}.$$

Hence (iii) is satisfied. □

Since (i), (ii) and (iii) are satisfied, as $\frac{1}{M_\varepsilon} = \frac{\varepsilon}{\zeta}$ and $\Delta_{\varepsilon'} = \frac{1}{M_\varepsilon}$, we have from Theorem 6

$$\begin{aligned} \mathbb{E}\left[T_\varepsilon\right] &\leq \frac{\beta}{2\beta - 1} \frac{\Pi_0}{\gamma g\left(\frac{1}{M_\varepsilon}\right)} + 1 \\ \iff \mathbb{E}\left[T_\varepsilon\right] &\leq \frac{\beta}{2\beta - 1} \frac{\Pi_0 \zeta^2}{\gamma} \varepsilon^{-2} + 1. \end{aligned} \tag{3.61}$$

Thus, from (3.61), the complexity rate of our algorithm is $O(\varepsilon^{-2})$, a common convergence rate in this framework [3, 9, 11, 25, 65].

3.4 Numerical experiments

We implemented Algorithm 2 in Julia 1.8.3 and named our method `PLM` for "Probabilistic Levenberg-Marquardt". As the acceptation condition $\xi_j \geq \frac{\eta_3}{\mu_j}$ of Algorithm 2 is necessary in the theory but too restrictive in practice, we adapted the implementation consider iteration j is a fail if $\rho_j < \eta_2$ and a very successful step if $\xi_j \geq \frac{\eta_3}{\mu_j}$ and $\rho_j \geq \eta_2$. We update

$\mu_{j+1} = \max\left\{\frac{\mu_j}{\lambda}, \mu_{min}\right\}$ only for very successful iterations. This means μ_j remains unchanged for successful iterations, similar to a standard LM step selection.

Besides, in the theory, we considered having $f_j \neq f_j^0$ possible. However, in this implementation, we use the same sample to compute both f_j and f_j^0. If not, we should compute an additional residual only for f_j^0 which spends more computing resources only for another estimate of $f(x_j)$. `STORM` algorithm [27] proposes variants stating that f_j^0 may be different to f_j, but the way f_j^0 is estimated is not explained. We then use the same sample for f_j and f_j^0 to save numerical cost, resulting in $f_j^0 = f_j$ like `TR-SAA` variant of `STORM`.

We set $\eta_1 = 10^{16}$ as we want to avoid as much as possible using the Cauchy point throughout the algorithmic process. It aims to decrease sufficiently the objective function by rejecting too large steps using the Cauchy step instead which satisfies condition (3.9).

We use a quadratic regularization method named `R2` from RegularizedOptimization.jl package to compute LM step at Line 8 [4]. We use in practice $s_{j,cp}$ as the initial guess of the proximal-based method used for computing s_j to satisfy (3.9). The solver steplength is defined by

$$\hat{\nu}_j \coloneqq \frac{\theta}{\|J_j^m\|^2 + \sigma_j}, \tag{3.62}$$

which can be calculated after σ_j update. It results in $s_{j,cp}$ not being exactly the first step of the proximal-based method used for s_j computation but still being an acceptable initial guess for `R2`. The nonsmooth subsolver stops whenever

$$\left(\hat{\xi}_{j,cp}(x_j + s_j; \hat{\nu}_j^{-1})\right)^{\frac{1}{2}} \leq \begin{cases} 10^{-1}, & \text{if } k = 0, \\ \max\left\{\varepsilon, \min\left(10^{-2}, \xi_{j,cp}(x_j, \nu_j^{-1})/10\right)\right\}, & \text{if } k > 0, \end{cases} \tag{3.63}$$

where $\hat{\nu}_j$ is defined by (3.62) and $\left(\hat{\xi}_{j,cp}(x_j + s_j; \hat{\nu}_j^{-1})\right)^{\frac{1}{2}}$ represents the first-order stationarity measure of the subsolver. Here $\psi(s; x) = h(x + s)$ and we compute it with tools from ShiftedProximalOperators.jl package [5].

We first devised a strategy using a constant sample rate $\hat{\tau} \in (0, 1]$, called $\hat{\tau}$`%`. We secondly designed methods using nondecreasing sample rate approaches (`Nondec-Batch`), taking $\hat{\tau}_0$ as the initial sample rate. The first scheme increases the sample rate with respect to the number of epochs (`Nondec-Batch (w/ Epoch)`): after two epochs at $\hat{\tau}_0 < 20\%$, we set the sample rate at 20% for one epoch, then at 50% for 3 epochs, then 5 epochs at 90% and ends at 100%. The second increases the sample rate with respect to the evolution of the stationarity metric (`Nondec-Batch (w/ Stationarity)`). It increases the sample rate whenever ξ_j is 10^{-1} lower

than a stored threshold ξ_{mem} initialized with ξ_0 and is decreased by 10^{-1} each time the sample rate changes. The goal of this approach is to increase the precision relative to the increasing accuracy throughout optimization.

We finally designed adaptive sample rate schemes (`Adapt-Batch`) based on the step selection and among sample rates used for `Nondec-Batch` schemes, i.e., among $\{\hat{\tau}_0, 20\%, 50\%, 90\%, 100\%\}$. When we encounter two consecutive very successful steps, we increase the sample rate; conversely, after two consecutive failures, we decrease it. However, using only this scheme may cause the sample rate to stay unchanged whenever we have too many consecutive successful iterations. We consequently devised an adaptive strategy using a buffer to bound below the sample rate (`Adapt-Batch (w/ Buffer)`). The sample rate is increased by λ for very successful steps and by λ^{-1} for each fail, while the buffer increases when too many successive iterations occur. The buffer takes successively the rates 20%, 50%, 90%, and 100% each time it has to change. This aims to prevent Algorithm 2 from accepting successful iterations that do not improve the value of f and to ensure we respect Assumption 6 asymptotically. Table 3.1 summarizes each sampling strategy.

In practice, we change the sample for each successful iteration or each time the sample rate changes for all the estimates to be based on the same sample at each iteration j. Otherwise, we leave it unchanged for unsuccessful ones where the sample rate does not change. It is a compromise we made between the numerical cost and the stochastic aspect of Algorithm 2 as each change of sample requires computing again r_j and J_j. Following this scheme, for all iteration j, each estimate is based on the same sample.

The stopping condition of Algorithm 2 depends on whether the sampling is constant or not, as described below:

- $\hat{\tau}\%$: stops when $\xi_j \leq \varepsilon_a + \varepsilon_r \xi_0$ three consecutive times,

- `Nondec-Batch` and `Adapt-Batch`: stops when $\hat{\tau} = 100\%$ and $\xi_j \leq \varepsilon_a + \varepsilon_r \xi_0$,

where ε_a and ε_r are the absolute and relative tolerances respectively. We set $\varepsilon_a = \varepsilon_r$ for simplicity. $\hat{\tau}\%$ is required to fulfill the stopping criterion $\xi_j \leq \varepsilon_a + \varepsilon_r \xi_0$ three consecutive times arbitrarily to be more confident for the current point to be stationary but without being too demanding for Algorithm 2 to stop. It is not required for `Nondec-Batch` and `Adapt-Batch` variants as they are designed to reach a 100% sample rate asymptotically.

We compare `PLM` strategies altogether and to `R2` from RegularizedOptimization.jl package [4]. Note that `100%` is a classical LM method with the only specificity that σ_j is subject to the

scaling formula (3.7). As numerical experiments have not been carried out so far, we also designed a smooth variant of `PLM` which is an implementation of the algorithm of Bergou et al. [9]. The smooth subproblem is approximately solved using LSMR from Krylov.jl package [51]. The stopping criterion of LSMR is

$$\|\hat{J}_j^\top \hat{r}_j\| \leq \min\left(10^{-1}, \varepsilon_a + \varepsilon_r \|J_j^\top r_j\|^{1.3}\right), \tag{3.64}$$

where $\hat{J}_j^\top \hat{r}_j$ is the estimated gradient of LSMR. Similar to `R2`, ε_a and ε_r are the absolute and the relative stopping tolerances of LSMR respectively, where $\varepsilon_a = \varepsilon_r$. When the Jacobian is sparse (like for Bundle Adjustment problems), we solve the LM subproblem using QRMumps.jl routines instead to exploit the sparse structure of the problem [22].

We run `PLM` 10 times, the plots are median results and tables show statistics of a median run. The number of evaluations of the Jacobian are real numbers since they are weighted relative to the current sample rate.

3.4.1 Nonlinear SVM Problems

We consider problems related to binary classification by nonlinear SVM of the form (3.1) where

$$r(x) = \mathbf{1} - \tanh\left(b \odot \langle A, x\rangle\right), \quad \text{where } \mathbf{1} = [1, 1, \ldots, 1]^\top.$$

We introduce the Mean Squared Error (MSE) defined as follows

$$l(\mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} (y_i - \hat{y}_i)^2,$$

where $\mathcal{B}$ is the sample (or mini-batch), $y_i \in \{-1, 1\}$ is a given label and $\hat{y}_i$ is a predicted label. In the least-squares context,

$$l(\mathcal{B}) = \frac{1}{|\mathcal{B}|} \|r_j^m\|^2 = \frac{2}{|\mathcal{B}|} f_j. \tag{3.65}$$

We run our tests ensuring $\sigma_0 = 1$.

IJCNN1

We tested our implementation on the IJCNN1 dataset containing 49990 samples and 22 features. The objective here is to train a neural network architecture for binary classification, starting with $x_0 = \mathbf{1}$ so approximately half of labels are misclassified.

Table 3.1: Summary of devised sampling strategies.

Method name	Batch size evolution
$\tau\%$	Constant of rate τ
Nondec-Batch (w/ Epoch)	Nondecreasing: Increase with respect to a specific number of epochs.
Nondec-Batch (w/ Stationarity)	Nondecreasing: Increase when ξ_j becomes lower than ξ_{mem}.
Adapt-Batch	Adaptive: Increase when 2 successive big success or decrease when 2 successive fails.
Adapt-Batch (w/ Buffer)	Adaptive: Increase when 2 successive big success or decrease when 2 successive fails. Increase lower bound of sample rate (buffer) if too long without changing.

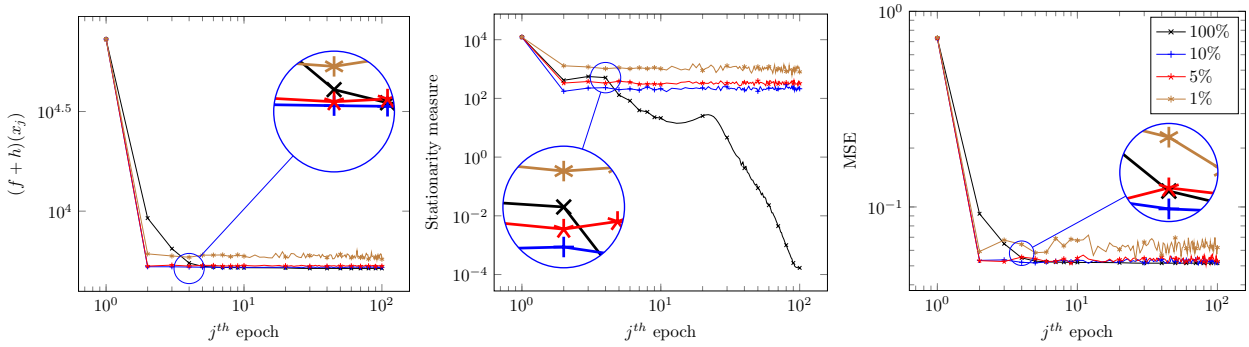
We use 100%, 10%, 5% and 1% batch size for $\hat{\tau}\%$ schemes and set the initial sample rate at 5% for `Nondec-Batch` and `Adapt-Batch` schemes. We set the precision $\varepsilon_a = \varepsilon_r = 10^{-8}$ and the maximum budget is of 100 epochs. We plot exact $f + h$ (i.e., $f + h$ with 100% batch-size), $\nu_j^{-1/2}\xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2}$ and MSE in terms of epochs for each strategy of `PLM` for three different regularizers: $\lambda\|\cdot\|_1$, $\lambda\|\cdot\|_{1/2}^{1/2}$ and $h = 0$ (i.e., the smooth variant) with $\lambda = 10^{-1}$. We choose those nonsmooth regularizers as they have an exact expression of their prox and they allow for having a sparse solution [3]. Note exact $f + h$ and $\nu_j^{-1/2}\xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2}$ are supposedly unavailable, but we compute them to draw unbiased comparisons between all the possible `PLM` sampling strategies.

Figs. 3.1 to 3.3 show that `5%` and `10%` are comparable relative to objective value and MSE descent while `1%` is outperformed by the others regarding the objective function value, the MSE, and the stationarity metric. Moreover, `5%` computes less expensive residuals and Jacobian products than `10%` and `100%`. However, constant sample rate strategies do not allow a decrease of the stationarity measure unlike `100%`.

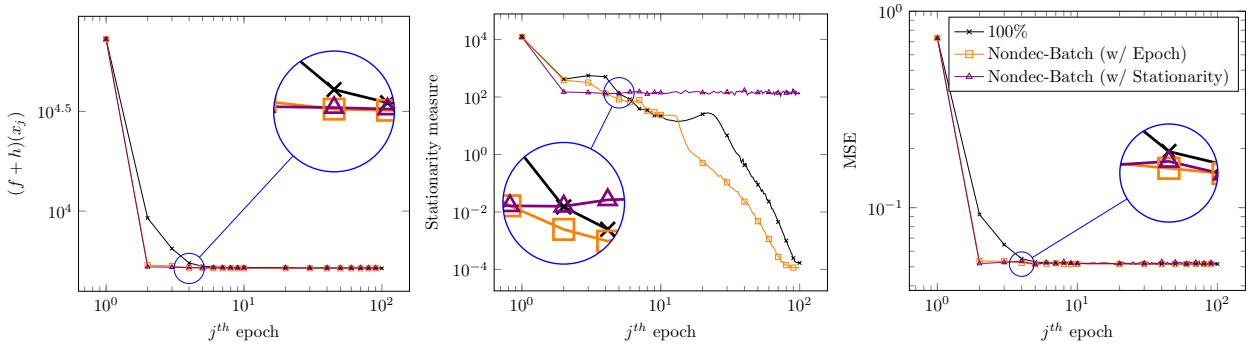
$\hat{\tau}\%$ for $\hat{\tau} < 1$ schemes are not able to decrease the stationarity measure below 10^2 asymptotically. However, `Nondec-Batch (w/ Epoch)` and `Adapt-Batch (w/ Buffer)` reach a 10^{-4} precision for both smooth and nonsmooth cases faster than `100%`.

Thus, for each h, the most promising approach among $\hat{\tau}\%$, `Nondec-Batch` and `Adapt-Batch` schemes are `5%`, `Nondec-Batch (w/ Epoch)` and `Adapt-Batch (w/ Buffer)` respectively. We denote in the sequel `Nondec-Batch (w/ Epoch)` and `Adapt-Batch (w/ Buffer)` by `Epoch` and `Success` respectively.

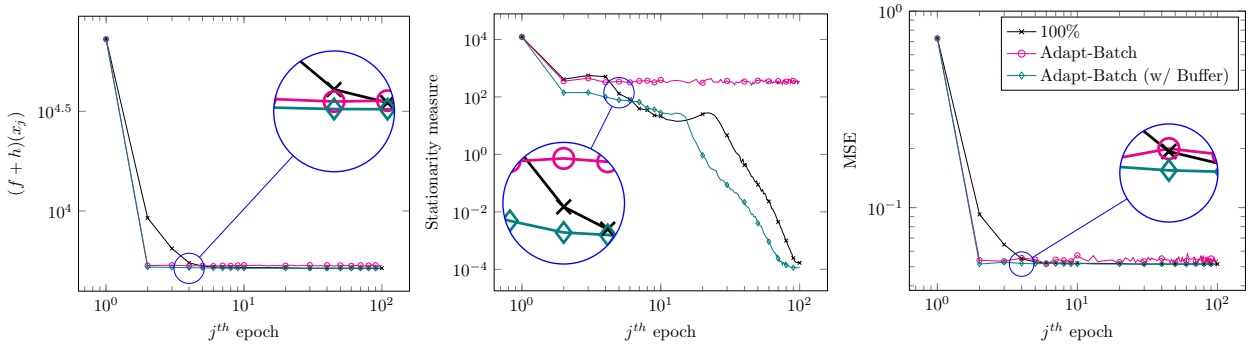
Fig. 3.4 shows the stochastic approaches decrease faster than `PLM 100%` to the objective function and the MSE, but the `PLM 5%` struggles to get high precision results unlike `PLM Epoch`



(a) Constant batch-size.

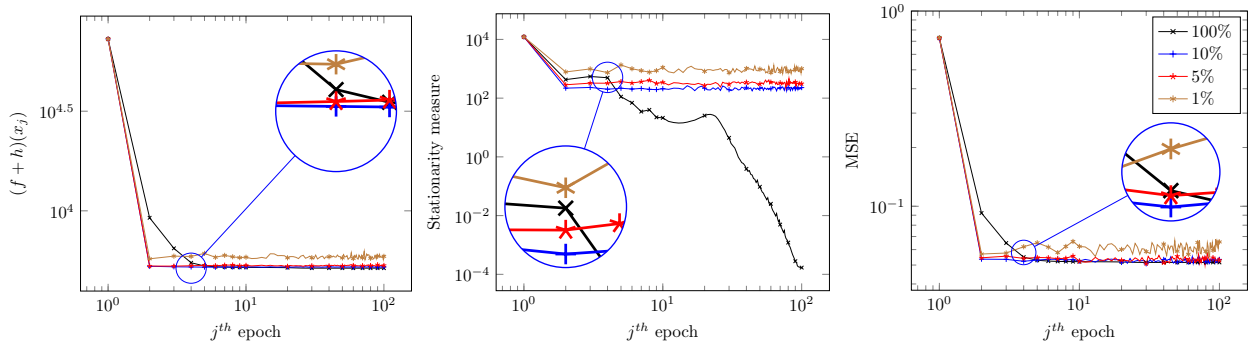


(b) Nondecreasing batch size.

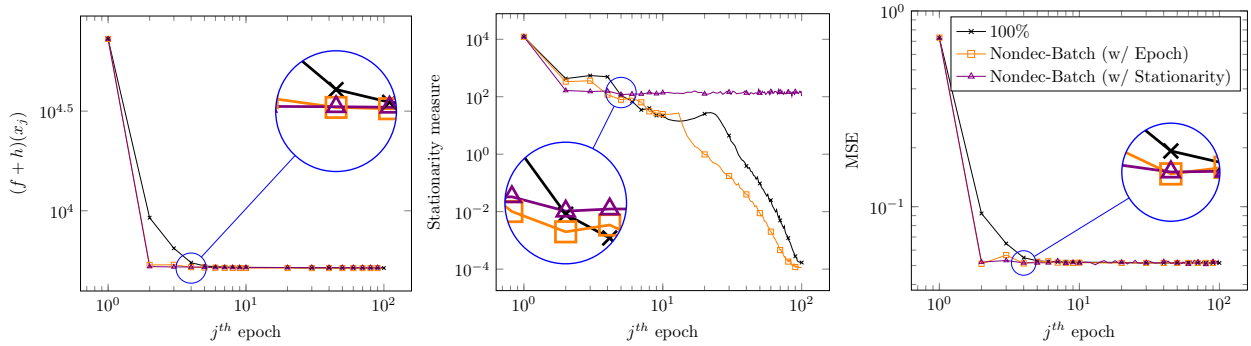


(c) Adaptive batch size.

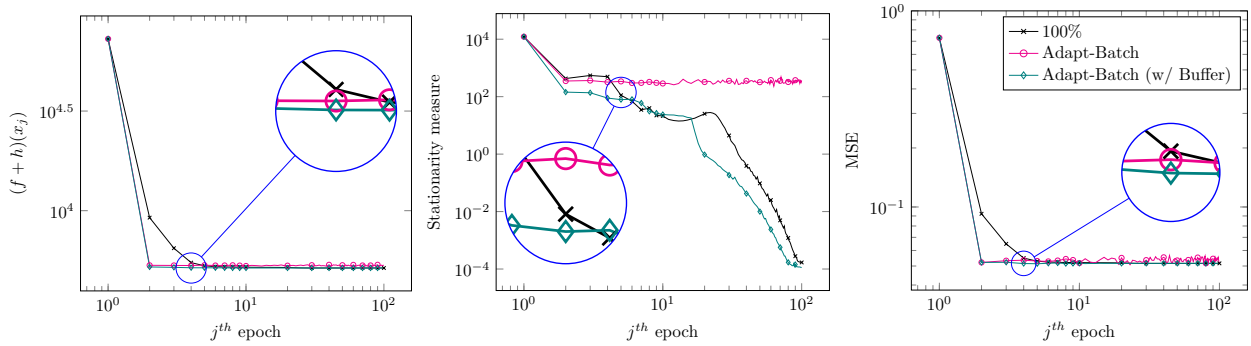
Figure 3.1 PLM method on IJCNN1 dataset with $h = \lambda\|\cdot\|_1$. Stationarity measure is $\nu_j^{-1/2}\xi^*_{\text{cp}}(x_j, \nu_j^{-1})^{1/2}$.



(a) Constant batch size.

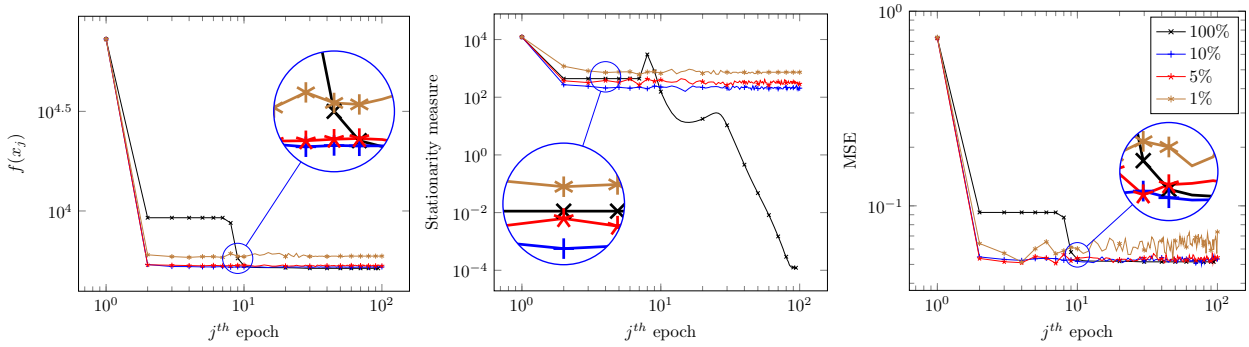


(b) Nondecreasing batch size.

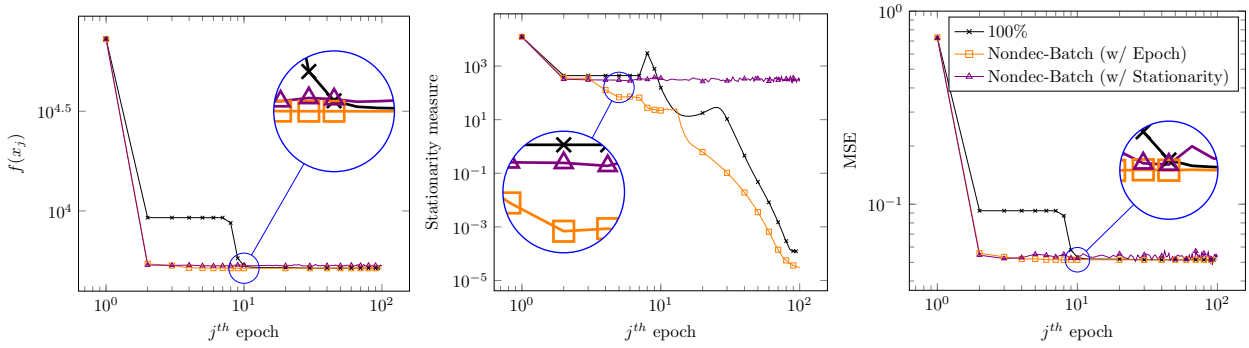


(c) Adaptive batch size.

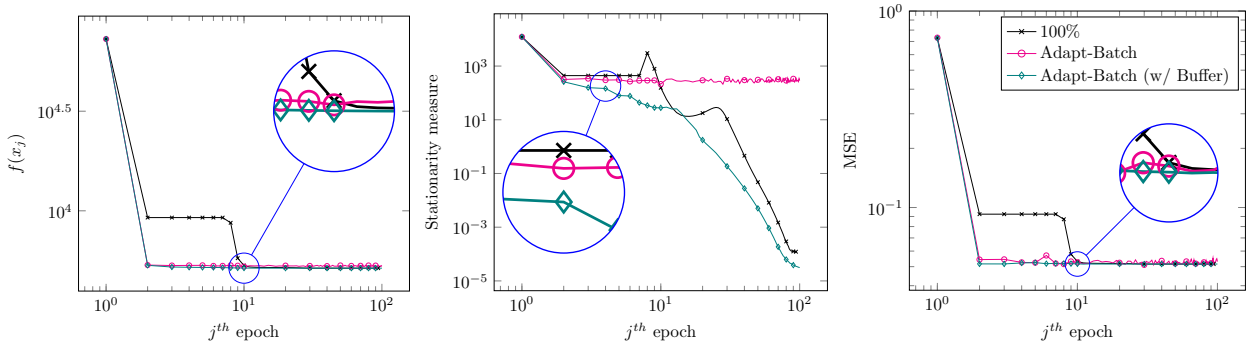
Figure 3.2 PLM method on IJCNN1 dataset with $h = \lambda\|\cdot\|_{1/2}^{1/2}$. Stationarity measure is $\nu_j^{-1/2}\xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2}$.



(a) Constant batch size.



(b) Nondecreasing batch size.



(c) Adaptive batch size.

Figure 3.3 PLM method on IJCNN1 dataset with $h = 0$. Stationarity measure is $\|J(x_j)^\top r(x_j)\|$.

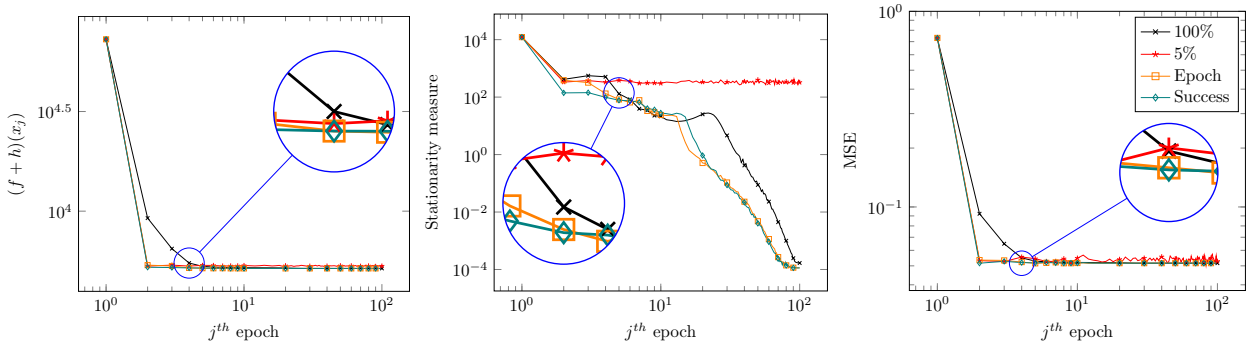


Figure 3.4 PLM methods on IJCNN1 dataset with $h = \lambda\|\cdot\|_1$. Stationarity measure is $\nu_j^{-1/2}\xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2}$.

and `PLM Success`. With asymptotically increasing batch sizes, `PLM` respects Assumption 6 and gets high precision results. Besides, `PLM Epoch` and `PLM Success` decrease the exact stationarity metric $\nu_j^{-1/2}\xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2}$ faster than `PLM 100%`. `PLM Epoch` and `PLM Success` achieve very similar performances on IJCNN1 according to Fig. 3.4.

Thus, stochastic variants are advantageous for the IJCNN1 problem, enabling faster convergence to higher-quality solutions.

MNIST

Consider an image recognition problem using the MNIST dataset, composed of 13,007 training and 2,163 testing vectorized images, each with a dimension of 784. For binary classification, we retain only the digits 1 and 7 and initialize x_0 as a vector of ones, resulting in approximately 50% misclassified data, similar to IJCNN1. As we want the support to be sparse, we choose $h = \lambda\|\cdot\|_{1/2}^{1/2}$ where $\lambda = 10^{-1}$, but we also run experiments with $h = 0$. The relative stopping condition here uses ξ_0 from `R2` and applies to all `PLM` methods to ensure consistency in stopping criteria. We give a budget of maximum 500 epochs (i.e., 500 iterations for deterministic methods) and set $\varepsilon = 10^{-4}$. We denote by $\#E$ the number of epochs, $\#f$ the number of residual computations, $\#\nabla f$ the number of Jacobian and transposed Jacobian products, and $\#inner$ the total number of subsolver iterations, representing the number of prox calls for nonsmooth methods and LSMR iterations for smooth ones. We also compare those methods with a reduced budget of 20 epochs.

Additionally, Tables 3.2 and 3.3 show statistics from the run with the median training set score. Note we display the value of $h = \lambda\|\cdot\|_{1/2}^{1/2}$ for the smooth variants as an indicator of the sparsity of the returned solution. As it is purely indicative information, we display it in italic font and we do not show the value of $f + h$ for smooth schemes since the problem nature is

different.

Table 3.2: Table of execution statistics on MNIST for 1 and 7 digits classification with a 500 epochs budget.

Alg	f	h	$f+h$	(Train, Test)	# E	$\#f$	$\#\nabla f$	# inner
Nonsmooth								
R2	64.23	81.58	145.81	(99.77, 99.31)	324	324.00	255.00	324
PLM ND-Epoch	91.11	57.57	148.67	(99.65, 99.17)	19	32.35	1596.65	614
PLM AD-Buffer	79.61	65.44	145.04	(99.65, 99.21)	12	18.05	1449.75	546
PLM 100%	66.44	76.19	142.63	(99.75, 99.31)	10	9.00	1643.00	575
PLM 5%	269.01	15.72	284.73	(98.90, 98.57)	231	462.05	9040.70	10805
Smooth								
R2	33.45	*310.17*	-	(99.89, 99.45)	501	501.00	402.00	501
PLM ND-Epoch	22.53	*370.22*	-	(99.94, 99.40)	21	35.35	529.10	239
PLM AD-Buffer	20.45	*375.67*	-	(99.95, 99.40)	13	18.85	374.15	127
PLM 100%	10.90	*411.05*	-	(99.99, 99.49)	15	14.00	426.00	108
PLM 5%	32.98	*323.48*	-	(99.88, 99.40)	23	68.95	698.95	1589

Table 3.3: Table of execution statistics on MNIST for 1 and 7 digits classification with a 20 epochs budget.

Alg	f	h	$f+h$	(Train, Test)	# E	$\#f$	$\#\nabla f$	# inner
Nonsmooth								
R2	190.93	93.92	284.85	(99.30, 99.12)	21	21.00	19.00	21
PLM ND-Epoch	90.32	60.50	150.82	(99.65, 99.08)	20	33.35	1492.20	575
PLM AD-Buffer	79.55	64.06	143.60	(99.67, 99.31)	14	20.85	1435.40	498
PLM 100%	66.44	76.19	142.63	(99.75, 99.31)	10	9.00	1643.00	575
PLM 5%	275.69	17.62	293.31	(98.93, 98.47)	20	39.85	847.30	1378
Smooth								
R2	170.77	*159.62*	-	(99.34, 99.17)	21	21.00	19.00	21
PLM ND-Epoch	20.79	*374.17*	-	(99.95, 99.40)	20	34.35	536.50	237
PLM AD-Buffer	20.55	*376.39*	-	(99.95, 99.40)	14	21.05	419.85	139
PLM 100%	10.90	*411.05*	-	(99.99, 99.49)	15	14.00	426.00	108
PLM 5%	58.91	*258.37*	-	(99.75, 99.45)	20	39.85	409.90	928

As for IJCNN1, Figs. 3.5 and 3.6 show nondeterministic `PLM` schemes enable a fast descent of the objective function and the MSE, but here `PLM 100%` becomes better in terms of $f+h$ value and MSE than the other methods after a few epochs. Besides, `PLM 5%` achieves a sharp decrease on the first epochs but is not able to converge ultimately for nonsmooth schemes.

In Tables 3.2 and 3.3, the h indicator for smooth variants is higher than nonsmooth ones, hence this latter is sparser and so cheaper to store. Fig. 3.7 shows a gray map representation of such solutions returned by methods using a nonsmooth regularization for a 500 epochs budget. These solution maps can be interpreted as how important are the pixels to help the SVM determine if the picture is a 1 or a 7.

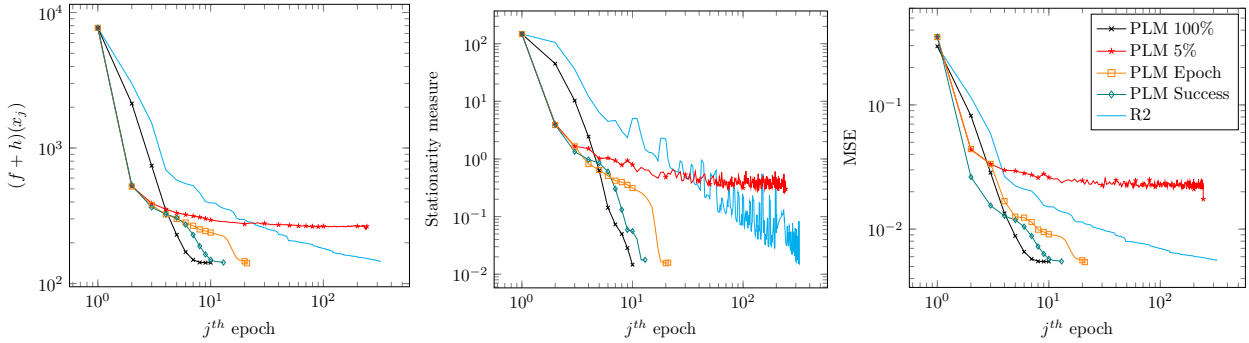


Figure 3.5 R2 and four PLM variants on MNIST with $h = \lambda\|\cdot\|_{1/2}^{1/2}$. Stationarity measure is $\nu_j^{-1/2}\xi_{\text{cp}}^*(x_j, \nu_j^{-1})^{1/2}$.

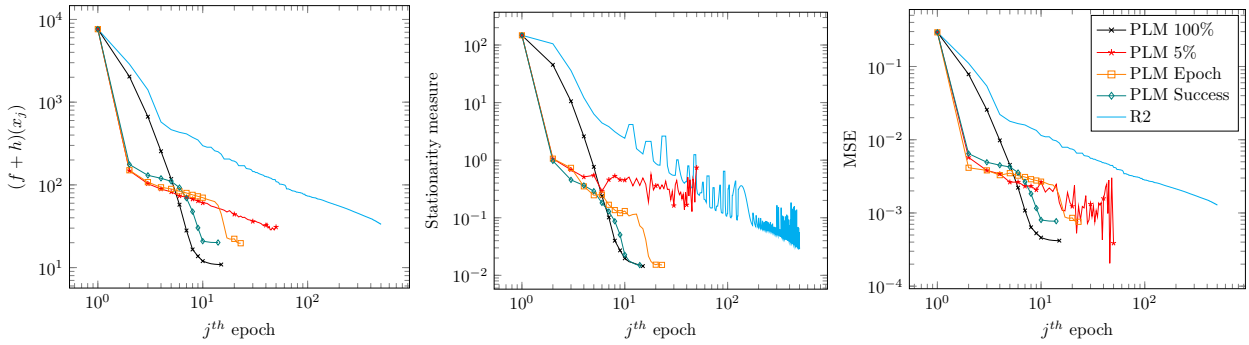
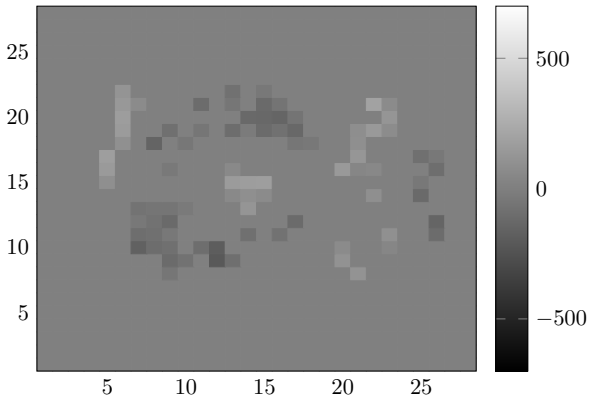
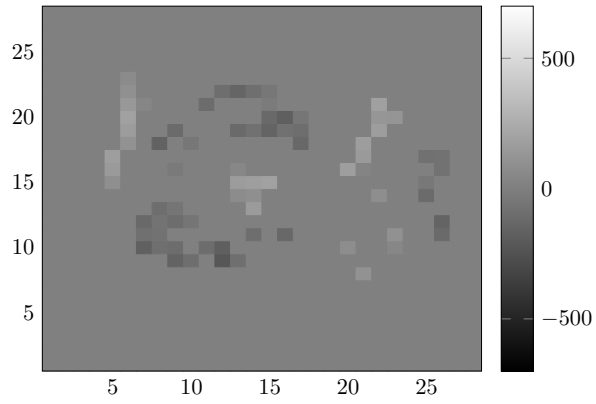


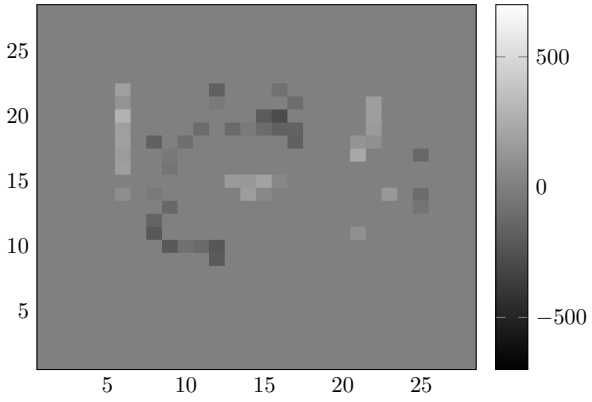
Figure 3.6 R2 and four PLM schemes on MNIST with $h = 0$. Stationarity measure is $\|(J_j^m)^\top r_j^m\|$.



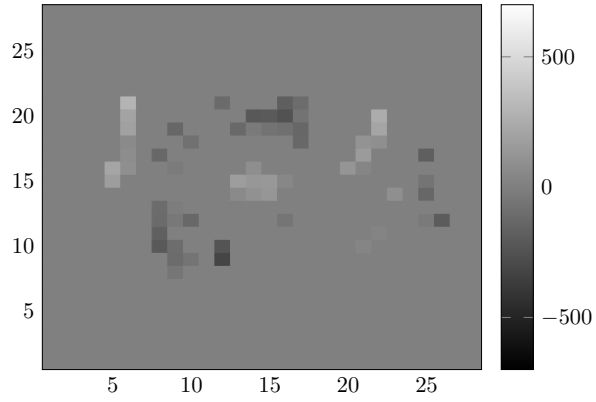
(a) Signal: R2



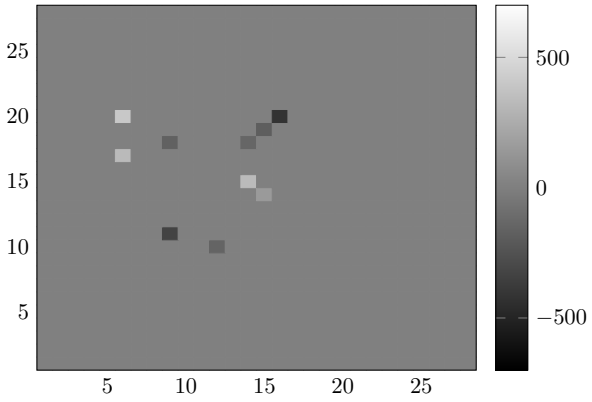
(b) Signal: PLM 100%



(c) Signal: PLM Epoch



(d) Signal: PLM Success



(e) Signal: PLM 5%

Figure 3.7 Representation of solutions for MNIST dataset with $h = \|\cdot\|_{1/2}^{1/2}$ and a budget of 500 epochs.

R2 needs more iterations to converge than PLM, so it is outperformed for a budget of 20 epochs (see Table 3.3). Among PLM schemes, PLM 5% returns the worst classifier while all the others are comparable in terms of $f + h$ final value and Train and Test accuracy score.

For nonsmooth variants, Tables 3.2 and 3.3 show that PLM Success and PLM Epoch carry out less J and $J^\top$ products than PLM 100% and fewer prox calls for PLM Success. However, they need more residual evaluations than PLM 100%. Indeed, we must compute additional residuals and Jacobian products each time we have a successful iteration or a sample rate change.

For smooth variants, Tables 3.2 and 3.3 show PLM 100% returns the best final f value while scores on Train and Test databases are comparable among the other solvers. PLM Success achieves to converge the fastest with the lowest Jacobian and transposed Jacobian products. However, PLM 100% needs the least number of LSMR iterations and the lowest number of evaluations of r.

Thus, a well-chosen sampling strategy achieves sharper objective descent with lower computational cost and comparable final statistics across smooth and nonsmooth variants. Nonsmooth variants return sparser solutions than smooth ones and are consequently cheaper to store.

3.4.2 Bundle Adjustment problems

Bundle Adjustment problems in computer vision aim to construct a 3D environment from 2D photographs [1]. Problem models for these tasks are available in the BundleAdjustmentModels.jl package [42].

We run `PLM` on the first problem from `dubrovnik` dataset composed of 16 images for 22106 points aiming to represent Onofrio's fountain (see on Figure 3.9 from the old city of Dubrovnik (Croatia). We illustrate the 3D scatter of the returned solution in Fig. 3.10 and the evolution of f and ∇f are in Fig. 3.8. We set $\varepsilon_a = \varepsilon_r = 10^{-8}$ and Table 3.4 presents the values of f and ∇f at both x_0 and the returned solution. It shows $\#E$ and $\#\nabla f$ as well which are the number of epochs and the number of Jacobian and transposed Jacobian products respectively. We conducted five test runs, each starting with $\sigma_0 = 10^{-1}$.

Table 3.4: Table of solution statistics of problem from `dubrovnik` dataset with $h = 0$.

Alg	$f(x_0)$	$f(x)$	$\|\nabla f(x_0)\|$	$\|\nabla f(x)\|$	# E	# ∇f
PLM 100.0%	4.19e+06	1.80e+04	1.71e+08	1.52e-01	8	13.00
PLM Epoch	4.19e+06	1.80e+04	1.71e+08	6.83e-03	18	41.30
PLM Success	4.19e+06	1.80e+04	1.71e+08	2.85e-03	10	29.35

Unlike nonlinear SVM, Fig. 3.8 shows stochastic approaches are not able to decrease the objective function sharper than a deterministic scheme, even if they are slightly better relative to the stationarity metric descent on the first epochs. Even if all the methods reach comparable solutions in terms of f and visualization (see on Fig. 3.10), the most competitive variant against `PLM 100%` remains `PLM Success`.

The results in Table 3.4 suggest that alternative sampling strategies may be needed for this type of problem to achieve lower computational costs. A possible explanation of why sampling fails on Bundle Adjustment is due to the very specific coherence of the overall problem. Stochastic estimates may mislead too much at the beginning of the optimization. To address this, we could provide more data to stochastic methods initially by starting with a lower sample rate and then increasing it more aggressively.

Besides, the current behavior of our methods tends to have a nondecreasing sample rate even if they can decrease it. It would be interesting to examine the results of a sample rate strategy that first increases, then decreases, and finally increases again. The advantage of such a scheme would be to allow `PLM` to reach a lower local minimum than the one found by deterministic approaches.

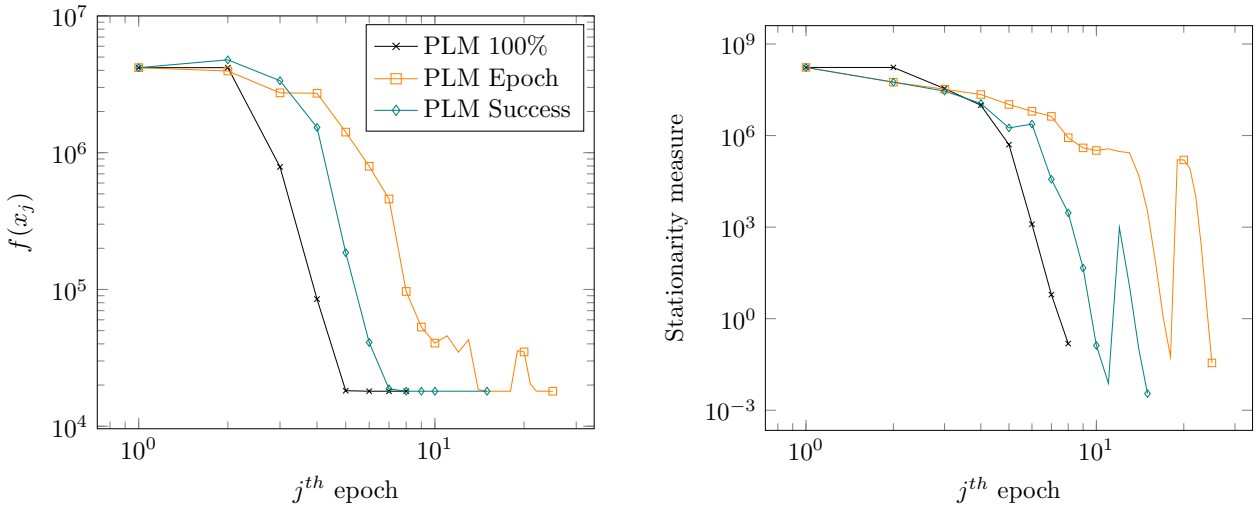
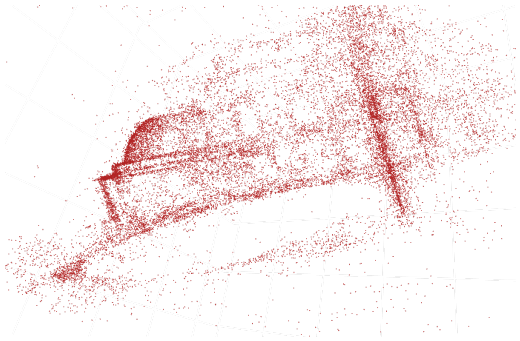
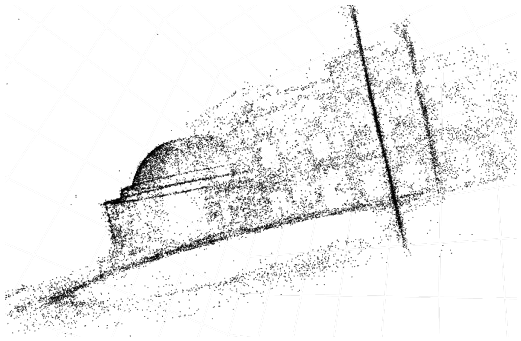


Figure 3.8 PLM methods on Bundle Adjustment Dubrovnik problem.

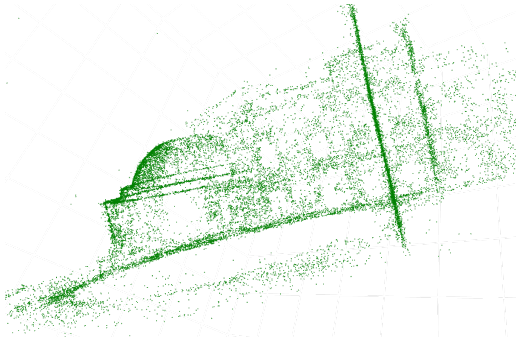
Figure 3.9 Onofrio's fountain © Dennis Jarvis, CC BY-SA 2.0



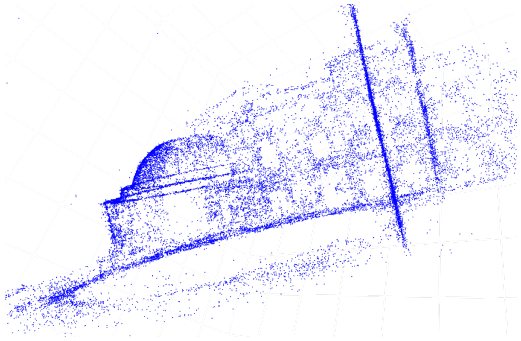
(a) Initial guess



(b) 100%



(c) Nondec-Batch



(d) Adapt-Batch

Figure 3.10 3D representation of Bundle Adjustment Dubrovnik problem using 3 `PLM` variants.

CHAPTER 4 CONCLUSION

4.1 Summary of Work

We proposed a Levenberg-Marquardt (LM) algorithm to solve nonsmooth regularized nonlinear least-squares problems, leveraging estimates of f and ∇f for large-scale optimization. We provided a convergence proof and complexity bound for the method using a scaling formula for the regularization parameter, allowing comparisons to trust-region schemes. The complexity bound in expectation aligns with that of other LM methods.

Our results demonstrate that utilizing estimates facilitates a sharp descent of the objective function, resulting in efficient schemes for low-budget or large-scale optimization. `PLM` is very efficient on IJCNN1 and reaches quickly solutions of very high quality. On the MNIST dataset, the stochastic schemes initially exhibit rapid descent, while the deterministic schemes ultimately converge faster. `PLM` may converge faster than `R2` and to a similar quality solution as `PLM 100%` with a lower computational cost on Jacobian products and prox evaluations.

Additionally, nonsmooth regularization enables `PLM` to return sparse solutions, unlike the dense solutions produced by smooth schemes, leading to significant savings in computational storage.

4.2 Limitations

The primary limitations of `PLM` stem from its sampling strategy, which can significantly impact results if poorly configured. First, `PLM` $\hat{\tau}\%$ methods struggle to produce accurate results because the accuracy assumptions are not met.

Results on MNIST highlight performance limits for `PLM Epoch`, especially concerning objective descent and computational cost. Computational savings are not always achieved, as stochastic schemes may require more outer or subsolver iterations, particularly in the smooth case.

Bundle Adjustment problems further highlight these limitations. In particular, the supposedly optimal nondeterministic `PLM` methods fail to outperform `PLM 100%` in terms of both descent speed and computational cost.

For stochastic estimates, we sampled only the smooth part of the problem, leaving the impact of estimates on the nonsmooth part untested. Additionally, our implementation does not exploit the flexibility of allowing $f_j^0 \neq f_j$, despite the theory supporting it.

4.3 Future Research

The performance of `PLM` depends significantly on the quality of the subsolver employed. Thus, developing a more efficient subsolver or using an improved one, like `R2N` [31], is crucial for enhancing the method's performance and should be explored.

Another relevant direction is to explore the inexact computation of h and the prox, in addition to f. Although sampling the variables is debatable, exploring alternative ways to estimate nonsmooth quantities could be valuable.

Since the current implementation does not leverage the flexibility of $f_j^0 \neq f_j$, developing an approach that employs distinct stochastic estimates for the model and objective function, without added computational cost, would be worthwhile.

Eventually, `PLM` can be improved through its sampling strategies. We tested basic ones based on simple heuristics, but more refined approaches may yield improved results. As there are potentially countless ways of sampling (and each problem may require its own one), further research may help to develop more generic and adaptive sampling schemes.

REFERENCES

[1] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building Rome in a day. *Commun. ACM*, 54(10):105–112, 2011.

[2] A. Y. Aravkin, R. Baraldi, and D. Orban. A proximal quasi-Newton trust-region method for nonsmooth regularized optimization. *SIAM J. Optim.*, 32(2):900–929, 2022.

[3] A. Y. Aravkin, R. Baraldi, and D. Orban. A Levenberg-Marquardt method for nonsmooth regularized least squares. *SIAM J. Sci. Comput.*, 46(4):A2557–A2581, 2024.

[4] R. Baraldi and D. Orban. RegularizedOptimization.jl: Algorithms for regularized optimization. `https://github.com/JuliaSmoothOptimizers/RegularizedOptimization.jl`, February 2022.

[5] R. Baraldi and D. Orban. ShiftedProximalOperators.jl: Proximal operators for regularized optimization. `https://github.com/JuliaSmoothOptimizers/ShiftedProximalOperators.jl`, February 2022.

[6] S. Bellavia, S. Gratton, and E. Riccietti. A Levenberg–Marquardt method for large nonlinear least-squares problems with dynamic accuracy in functions and gradients. *Num. Math.*, 140(3):791–825, Nov. 2018.

[7] E. Bergou, S. Gratton, and L. N. Vicente. Levenberg-Marquardt methods based on probabilistic gradient models and inexact subproblem solution, with application to data assimilation. *SIAM/ASA J. Uncertain. Quantif.*, 4(1):924–951, 2016.

[8] E. Bergou, Y. Diouane, V. Kunc, V. Kungurtsev, and C. Royer. A subsampling line-search method with second-order results. *INFORMS J. Opt.*, 4(4):403–425, 2022.

[9] E. Bergou, Y. Diouane, V. Kungurtsev, and C. W. Royer. A stochastic Levenberg-Marquardt method using random models with complexity results. *SIAM/ASA J. Uncertain. Quantif.*, 2022.

[10] E. H. Bergou, N. K. Chada, and Y. Diouane. A stochastic iteratively regularized Gauss-Newton method, 2024.

[11] J. Blanchet, C. Cartis, M. Menickelly, and K. Scheinberg. Convergence rate analysis of a stochastic trust region method for nonconvex optimization. *arXiv preprint arXiv:1609.07428*, 5, 2016.

[12] J. Blanchet, C. Cartis, M. Menickelly, and K. Scheinberg. Convergence rate analysis of a stochastic trust-region method via supermartingales. *INFORMS J. Opt.*, 1(2):92–119, 2019.

[13] J. Bolte, A. Daniilidis, O. Ley, and L. Mazet. Characterizations of Lojasiewicz inequalities and applications, 2008.

[14] J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Math. Program.*, 146(1):459–494, 2014.

[15] A. A. Borovkov. *An Arbitrary Space of Elementary Events*, pages 13–29. Springer London, London, 2013. ISBN 978-1-4471-5201-9.

[16] A. A. Borovkov. *Numerical Characteristics of Random Variables*, pages 65–105. Springer London, London, 2013. ISBN 978-1-4471-5201-9. `https://doi.org/10.1007/978-1-4471-5201-9_4`.

[17] A. A. Borovkov. *Random Variables and Distribution Functions*, pages 31–63. Springer London, London, 2013. ISBN 978-1-4471-5201-9.

[18] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Rev.*, 60(2):223–311, 2018.

[19] R. I. Boţ, E. R. Csetnek, and S. C. László. An inertial forward–backward algorithm for the minimization of the sum of two nonconvex functions. *EURO Journal on Computational Optimization*, 4(1):3–25, 2016. ISSN 2192-4406.

[20] J. Burke and S.-P. Han. A Gauss-Newton approach to solving generalized inequalities. *Math. Oper. Res.*, 11(4):632–643, 1986.

[21] J. V. Burke. Descent methods for composite nondifferentiable optimization problems. *Math. Program.*, 33:260–279, 1985.

[22] A. Buttari and A. Montoison. QRMumps.jl: A Julia interface to qr_mumps. `https://github.com/JuliaSmoothOptimizers/QRMumps.jl`, April 2021.

[23] H. Cao, P. Tao, H. Li, and J. Shi. Bundle adjustment of satellite images based on an equivalent geometric sensor model with digital elevation model. *ISPRS Journal of Photogrammetry and Remote Sensing*, 156:169–183, 2019.

[24] C. Cartis and L. Roberts. A derivative-free Gauss-Newton method. *Math. Program. Comp.*, 11:631–674, 2019.

[25] C. Cartis, N. I. Gould, and Ph. L. Toint. On the evaluation complexity of composite function minimization with applications to nonconvex nonlinear programming. *SIAM J. Optim.*, 21(4):1721–1739, 2011.

[26] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[27] R. Chen, M. Menickelly, and K. Scheinberg. Stochastic optimization using a trust-region method and random models. *Math. Program.*, 169:447–487, 2018.

[28] A. R. Conn, N. I. Gould, and P. L. Toint. *Trust region methods*. SIAM, 2000.

[29] P. Courtier, J.-N. Thépaut, and A. Hollingsworth. A strategy for operational implementation of 4D-Var, using an incremental approach. *Quarterly Journal of the Royal Meteorological Society*, 120(519):1367–1387, 1994.

[30] Q. Dai, B. Zhao, S. Wang, D. Huang, and C. Jin. Numerical simulation scheme of jointed rock masses using uav photogrammetry and a disk-based discontinuous deformation analysis model. *Electronic Research Archive*, 31:3381–3399, 04 2023.

[31] Y. Diouane, M. L. Habiboullah, and D. Orban. A proximal modified quasi-Newton method for nonsmooth regularized optimization. Cahier G-2024-64, GERAD, Montréal QC, Canada, 2024.

[32] D. Drusvyatskiy and A. S. Lewis. Error bounds, quadratic growth, and linear convergence of proximal methods. *Math. Oper. Res.*, 43(3):919–948, 2018.

[33] D. Drusvyatskiy and C. Paquette. Efficiency of minimizing compositions of convex functions and smooth maps. *Math. Program.*, 178:503–558, 2019.

[34] D. Drusvyatskiy, A. D. Ioffe, and A. S. Lewis. Nonsmooth optimization using Taylor-like models: error bounds, convergence, and termination criteria. *Math. Program.*, 185: 357–383, 2021.

[35] D. C.-L. Fong and M. Saunders. Lsmr: An iterative algorithm for sparse least-squares problems. *SIAM Journal on Scientific Computing*, 33(5):2950–2971, 2011.

[36] M. Fukushima and H. Mine. A generalized proximal point algorithm for certain non-convex minimization problems. *Int. J. Syst. Sci.*, 12(8):989–1000, 1981.

[37] W. Givens. Computation of plain unitary rotations transforming a general matrix to triangular form. *SIAM*, 6(1):26–50, 1958.

[38] G. N. Grapiglia, J. Yuan, and Y.-X. Yuan. Nonlinear stepsize control algorithms: complexity bounds for first-and second-order optimality. *J. Optimiz. Theory App.*, 171: 980–997, 2016.

[39] M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 1225–1234. PMLR, 2016.

[40] F. J. Herrmann, X. Li, A. Y. Aravkin, and T. van Leeuwen. A modified, sparsity-promoting, Gauss-Newton algorithm for seismic waveform inversion. In *Wavelets and Sparsity XIV*, volume 8138, pages 210–223. SPIE, 2011.

[41] A. S. Householder. Unitary triangularization of a nonsymmetric matrix. *J. Assoc. Comput. Mach.*, 5(4):339–342, Oct. 1958. ISSN 0004-5411.

[42] A. Kenens and D. Orban. BundleAdjustmentModels.jl: Julia repository of bundle adjustment problems. `https://github.com/JuliaSmoothOptimizers/BundleAdjustmentModels.jl`, March 2023.

[43] K. Kurdyka. On gradients of functions definable in o-minimal structures. In *Annales de l'institut Fourier*, volume 48, pages 769–783, 1998.

[44] G. Leconte and D. Orban. Complexity of trust-region methods with unbounded hessian approximations for smooth and nonsmooth optimization. Cahier G-2023-65, GERAD, Montréal QC, Canada, Mar. 2024.

[45] G. Leconte and D. Orban. The indefinite proximal gradient method. *Comput. Optim. Appl.*, Oct. 2024.

[46] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[47] K. Levenberg. A method for the solution of certain problems in least squares. *Q. Appl. Math.*, 2:164–168, 1944.

[48] A. S. Lewis and S. J. Wright. A proximal method for composite minimization. *Math. Program.*, 158(1):501–546, July 2016.

[49] H. Li and Z. Lin. Accelerated proximal gradient methods for nonconvex programming. *Advances in neural information processing systems*, 28, 2015.

[50] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 11:431–441, 1963.

[51] A. Montoison and D. Orban. Krylov.jl: A Julia basket of hand-picked Krylov methods. *J. Open Res. Softw.*, 8(89):5187, 2023.

[52] Y. Nesterov. Modified Gauss-Newton scheme with worst case guarantees for global performance. *Optim. Method Softw.*, 22(3):469–483, 2007.

[53] W. S. Noble. What is a support vector machine? *Nat. Biotech.*, 24(12):1565–1567, Dec. 2006.

[54] J. Nocedal and S. J. Wright, editors. *Fundamentals of Unconstrained Optimization*, pages 10–32. Springer New York, New York, NY, 1999.

[55] J. Nocedal and S. J. Wright. *Nonlinear Least-Squares Problems*, pages 250–275. Springer New York, New York, NY, 1999.

[56] C. Paquette and K. Scheinberg. A stochastic line search method with expected complexity analysis. *SIAM J. Optim.*, 30(1):349–376, 2020.

[57] M. J. Powell. On the global convergence of trust region algorithms for unconstrained minimization. *Math. Program.*, 29:297–303, 1984.

[58] R. T. Rockafellar and R. J.-B. Wets. *Variational Analysis*. Springer Verlag, Heidelberg, Berlin, New York, 1998.

[59] J. O. Royset. Consistent approximations in composite optimization. *Math. Program.*, 201(1):339–372, 2023.

[60] L. Stella, A. Themelis, P. Sopasakis, and P. Patrinos. A simple and efficient algorithm for nonlinear model predictive control. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1939–1944. IEEE, 2017.

[61] A. Tarantola. *Inverse problem theory and methods for model parameter estimation*. SIAM, 2005.

[62] A. Themelis, L. Stella, and P. Patrinos. Forward-backward envelope for the sum of two nonconvex functions: Further properties and nonmonotone linesearch algorithms. *SIAM Journal on Optimization*, 28(3):2274–2303, 2018.

[63] Y. Trémolet. Model-error estimation in 4D-Var. *Quarterly Journal of the Royal Meteorological Society*, 133(626):1267–1280, 2007.

[64] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings*, pages 298–372. Springer, 2000.

[65] K. Ueda and N. Yamashita. On a global complexity bound of the Levenberg-Marquardt method. *J. Optimiz. Theory App.*, 147:443–453, 2010.

[66] S. J. Wright and J. N. Holt. An inexact Levenberg-Marquardt method for large sparse nonlinear least squres. *J. Austral. Math. Soc. Ser. B*, 26(4):387–403, 1985.

[67] Z. Zhou, Z. Liu, C. Liu, and L. Luo. Incremental Gauss-Newton methods with superlinear convergence rates. *arXiv preprint arXiv:2407.03195*, 2024.