Second Order and High Order Approaches for Nonconvex Optimization with Objective Function-Free Algorithms

Sadok Jerad

RÉPUBLIQUE FRANÇAISE
Liberté Égalité Fraternité

Université de Toulouse

Doctorat de l'Université de Toulouse

préparé à Toulouse INP

Approches du second ordre de d'ordre élevées pour l'optimisation nonconvex avec variantes sans évaluation de la fonction objective

---

Thèse présentée et soutenue, le 19 janvier 2024 par

Sadok JERAD

École doctorale
EDMITT - École Doctorale Mathématiques, Informatique et Télécommunications de Toulouse

Spécialité
Mathématiques et Applications

Unité de recherche
IRIT : Institut de Recherche en Informatique de Toulouse

Thèse dirigée par
Serge GRATTON

Composition du jury
Mme Stefania BELLAVIA, Présidente, Università di Firenze
Mme Coralia CARTIS, Rapporteuse, University of Oxford
M. Rolf KRAUSE, Rapporteur, Università della Svizzera italiana
M. Phillipe Louis TOINT, Examinateur, Université de Namur
M. Jérôme BOLTE, Examinateur, Université de Toulouse 1 - Capitole
M. Frank E. CURTIS, Examinateur, University of Leigh
M. Eouharn SIMON, Examinateur, Toulouse INP
M. Serge GRATTON, Directeur de thèse, Toulouse INP

À mes deux parents,
À mes deux soeurs,
À mes grand-parents

Resumé

Même si l'optimisation non linéaire semble (a priori) être un domaine mature, de nouveaux schémas de minimisation sont proposés ou redécouverts pour les problèmes modernes à grande échelle. A titre d'exemple et en rétrospective de la dernière décennie, nous avons vu une vague de méthodes du premier ordre avec différentes analyses, malgré le fait que les limitations théoriques bien connues de ces méthodes ont été discutées en profondeur auparavant.

Cette thèse explore deux lignes principales de recherche dans le domaine de l'optimisation non-convexe avec un accent particulier sur les méthodes de second ordre et d'ordre supérieur. Dans la première série [114, 116], nous nous concentrons sur les algorithmes qui ne calculent pas les valeurs des fonctions et opèrent sans connaissance d'aucun paramètre, car les méthodes du premier ordre les plus adaptées pour les problèmes modernes appartiennent à cette dernière catégorie. Dans [114], nous commençons par redéfinir l'algorithme bien connu d'Adagrad dans un cadre de région de confiance et utilisons ce dernier paradigme pour étudier deux classes d'algorithmes OFFO (Objective-Free Function Optimization) déterministes du premier ordre. Pour permettre des algorithmes OFFO exacts plus rapides, nous proposons ensuite une méthode de régularisation adaptative déterministe d'ordre p qui évite le calcul des valeurs de la fonction [116]. Cette approche permet de retrouver la vitesse de convergence bien connu du cadre standard lors de la recherche de points stationnaires, tout en utilisant beaucoup moins d'informations.

Dans la deuxième série d'articles [118, 117], nous analysons les algorithmes adaptatifs dans le cadre plus classique où les valeurs des fonctions sont utilisées pour adapter les paramètres. Dans [118], nous étendons les méthodes de régularisation adaptatives à une classe spécifique d'espaces de Banach en développant un algorithme de descente du gradient de Hölder. En plus, nous étudions un algorithme de second ordre qui alterne entre la courbure négative et les étapes de Newton avec un taux de convergence quasi optimal [117]. Pour traiter les problèmes de grande taille, nous proposons des versions sous-espace de l'algorithme qui montrent des performances numériques prometteuses.

Dans l'ensemble, cette recherche couvre un large éventail de techniques d'optimisation et fournit des informations et des contributions précieuses aux algorithmes d'optimisation adaptatifs et sans paramètres pour les fonctions non convexes. Elle ouvre également la voie à des développements théoriques ultérieurs et à l'introduction d'algorithmes numériques plus rapides.

Thesis Abstract

Even though nonlinear optimization seems (a priori) to be a mature field, new minimization schemes are proposed or rediscovered for modern large-scale problems. As an example and in retrospect of the last decade, we have seen a surge of first-order methods with different analysis, despite the fact that well-known theoretical limitations of the previous methods have been thoroughly discussed.

This thesis explores two main lines of research in the field of nonconvex optimization with a narrow focus on second and higher order methods. In the first series [114, 116], we focus on algorithms that do not compute function values and operate without knowledge of any parameters, as the most popular currently used first-order methods fall into the latter category. In [114], we start by redefining the well-known Adagrad algorithm in a trust-region framework and use the latter paradigm to study two first-order deterministic OFFO (Objective-Free Function Optimization) classes. To enable faster exact OFFO algorithms, we then propose a pth-order deterministic adaptive regularization method that avoids the computation of function values [116]. This approach recovers the well-known convergence rate of the standard framework when searching for stationary points, while using significantly less information.

In the second set of papers [118, 117], we analyze adaptive algorithms in the more classical framework where function values are used to adapt parameters. In [118], we extend adaptive regularization methods to a specific class of Banach spaces by developing a Hölder gradient descent algorithm. In addition, we investigate a second-order algorithm that alternates between negative curvature and Newton steps with a near-optimal convergence rate [117]. To handle large problems, we propose subspace versions of the algorithm that show promising numerical performance.

Overall, this research covers a wide range of optimization techniques and provides valuable insights and contributions to both parameter-free and adaptive optimization algorithms for nonconvex functions. It also opens the door for subsequent theoretical developments and the introduction of faster numerical algorithms.

Remerciements

Avant de plonger dans les arcanes de ce "kteb"[1], je souhaite avant tout adresser des mots de remerciements à toutes les personnes sans qui ce travail, fruit de trois années de labeur, n'aurait pas été réalisable ni concevable.

Mes premiers chaleureux remerciements s'adressent en particulier à mes deux directeurs de thèse, Serge Gratton et Philippe L. Toint. Ce fut un grand honneur pour moi d'être sous votre tutelle pour accomplir mes premiers pas dans la recherche scientifique en général et l'optimisation plus particulièrement. Philippe m'a marqué par son remarquable dévouement, son exigence afin de mener des travaux de qualité et surtout son style d'écriture affiné. J'ose espérer qu'à ton contact j'en ai pris de la graine. Serge, ta culture scientifique, ta rapidité à embrayer sur les nouvelles thématiques, tes remarques perspicaces et ton sens relationnel ont permis d'accélérer le développement de cette thèse. Je vous suis infiniment reconnaissant pour tous vos conseils, remarques et soutiens fournis durant ces trois années.

Je tiens à exprimer mes chaleureux remerciements aux trois rapporteurs Stefania Bellavia, Coralia Cartis, Raulf Krause d'avoir consacré une partie de leur temps afin de fournir des rapports de qualité contribuant à améliorer le présent manuscrit. Merci aussi à Stefania d'avoir réalisé le déplacement à Toulouse afin d'assister à la défense. Je tiens aussi à remercier les examinateurs Jérôme Bolte, Frank E. Curtis, Ehouarn Simon pour les échanges enrichissants et les nouvelles thématiques qu'ils ont proposé en lien avec mon travail.

Je pense aussi à tous les membres de l'équipe APO qui ont été présents durant ces trois années de thèse. Antoine Bernigaud, Antoine Jégo, Bastien, Olivier, Rémy, Jérémy, Mathis, Valentin, Boris, Sophie qui ont tous contribué à l'atmosphère détendue qui régnait au troisième étage du bâtiment F de l'ENSEEIHT. Pensée spéciale à Théo, mon collègue de bureau au F318 avec qui on a formé le duo inamovible RU 11h30.

Comme 'le travail c'est la santé et ne rien faire c'est la préserver', je tiens à remercier mes deux amis insulaires Jean-Baptiste et Nicolas pour les moments de convivialité devant le tournoi des Six Nations et les restos tunisiens tactiques du week-end. Une pensée à mon ami Wilson que j'ai connu depuis mon arrivée à Polytechnique et qui vient de temps en temps à Toulouse et à Skander, mon ami depuis le lycée, et compagnon de soirée lors de mes escapades parisiennes.

La thèse étant l'aboutissement d'un long cheminement. Il va de soi que ce manuscrit comporte la contribution même infime des enseignants, professeurs et camarades que j'ai connus auparavant à divers endroits que ce soit à Tunis comme à esprit-prépa ou au lycée Bourguiba, à Paris à l'école polytechnique et encore plus en amont de mon parcours scolaire.

[1] livre en tunisien

Évidemment que la famille élargie a aussi eu son rôle et je tiens à les remercier affectueusement pour le soutien et l'affection fournis. Je remercie ainsi les tantes et oncles, paternels et maternels, et les cousins pour tout ce qu'ils m'ont donné. Une pensée particulière à mes oncles Adel, Jalel, Jridi, Kamel et à ma tante Alia d'être venus de Tunis et d'avoir été présents à la soutenance de thèse. Je remercie aussi ma cousine Yasmine pour l'aide apportée lors de mon installation à Toulouse. Comme mon parcours académique en France a commencé par les épreuves orales des grandes écoles, je remercie mon oncle Sahbi pour les conditions de séjour optimales qu'il m'a offertes durant l'été 2016.

Il va de soi que ma gratitude et ma reconnaissance sont incommensurables envers ma famille proche. À mon père Habib qui m'a tenu la main pour m'apprendre à lire et à écrire et à ma mère Samira qui m'a tant entouré d'affection et de bienveillance, à mes deux sœurs Fatma et Khadija pour avoir tracé le sentier de l'excellence académique, et à mes grands-parents paternels et maternels qui ont su apporter des havres de joie que ce soit à Kairouan, à Bounouma aux Îles Kerkennah et à Tunis. Je leur dédie à tous ce livre avec une pensée particulière à mon défunt grand-père paternel Sadok.

Sur ce, je vous souhaite une bonne lecture.

Notation Index

$\mathbb{R}^n$	n-tuples of real numbers
I_n	Idendity matrix of dimension n
$\lambda_{\min}(M)$, $\lambda_{\max}(M)$	Minimum and maximum eigenvalues of a symmetric matrix M
$\odot$	Hadmard product
e_i	ith column of I_n for $1 \leq i \leq n$
$[x]_+$	$\max(x, 0)$ for $x \in \mathbb{R}$
$x^\intercal y$	Scalar product between two vectors of $\mathbb{R}^n$
$\mathcal{V}$	Infinite dimensionnal Banach space
$\mathcal{V}'$	Dual of $\mathcal{V}$
$\langle ., . \rangle$	Dual pairing between $\mathcal{V}'$ and $\mathcal{V}$
$\|.\|_{\mathcal{V}}$	Primal norm in $\mathcal{V}$
$\|.\|_{\mathcal{V}'}$	Dual norm in $\mathcal{V}'$
$f(x)$	Objective function
$\nabla_x^j f$	Derivative tensor of f of order j
$\mathcal{O}(.)$	No-larger-than orders of its argument
k	Iteration counter; as a subscript indicates a quantity at iteration k
x_k	kth iterated version of vector k
s_k	Step at iteration k
$T_{f,p}$	pth order Taylor serie approximation
Δ_k	Trust-region radii
σ_k	Regularization parameter at iteration k
m_k	Model at $x = x_k$
g_k, H_k	Gradient and Hessian at the iterate x_k respectively

$\mathcal{C}^{p,\beta}(\mathcal{V};\mathbb{R})$	The class of p times Fréchet differentiable function with β-Hölder pth derivative
α, α_k	Stepsize along a descent direction
$\mathbb{E}_{.\sim\mathcal{D}}[.]$	Excpectation of . under a distribution $\mathcal{D}$
$\overline{f}$	Inexact approximation of a specfic quantity
$\mathcal{L}^{\ell}(\mathcal{V}^{\otimes\ell})$	space of multilinear continuous functionals from $\mathcal{V}\times\mathcal{V}\times\cdots\times\mathcal{V}$ to $\mathbb{R}$
$\mathcal{L}^{\ell}_{sym}(\mathcal{V}^{\otimes\ell})$	subspace of $\mathcal{L}^{\ell}(\mathcal{V}^{\otimes\ell})$ that is m-linear symmetric

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In computational mathematics, engineering, or finance, many problems can be reformulated as optimization problems. The variable to be optimized might be restricted to a vector space or a discrete set. Additionally, the objective function can have specific structures, such as being convex, nonconvex, linear, or quadratic. If there is no prior knowledge about the structures the problem possesses, and the optimization space is continuous, such problems are categorized under Non-Linear Programming (NLP) terminology.

However, to achieve efficiency in devising minimization schemes, it's crucial to exploit the inherent structure of the given problem. For example, when the objective function is convex and the decision variable is unconstrained, efficient numerical methods can be applied. This is because every local minimum in such scenarios is also a global minimum [165, 168]. In the following, we will primarily focus on the unconstrained problem

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

where f is both differentiable and nonconvex. Unlike the convex scenario, achieving global optimization for the aforementioned problem is usually unattainable. Moreover, verifying that a feasible point is a local minimum of (P) becomes an NP-hard challenge, as demonstrated in [161]. Consequently, for the unconstrained nonconvex NLP category, we might primarily aim for local improvements from a given starting point.

Algorithms for solving (P) have a lengthy history predating modern computers. As early as 1830, Cauchy introduced the steepest descent method, which relies solely on the gradient [139]. Such optimization schemes, utilizing only gradient-related information, are termed *first-order optimization methods*. However, these methods often fall short in effectively navigating the loss landscape, leading to slow convergence, even for toy problems. The slow pace becomes evident when using the gradient of an ill-conditioned quadratic function.

To accelerate the optimization process, one could consider employing Newton's method. Originally, Newton applied this method to solve polynomial equations, but it was later adapted to address broader issues. For a historical perspective on the Newton-Raphson algorithm, [217] offers comprehensive insights. Building on these methods, another pivotal development was the Gauss-Newton method, tailored specifically for a certain class of (P) problems. Since these schemes utilize the Hessian or its efficient approximation, they fall under the category of *second-order methods*. Variations of Newton's method typically achieve rapid convergence as they near the solution. However, if they start far from the optimal

point, these algorithms might converge slowly or, in the worst-case scenario, even diverge. These pitfalls can be mitigated using step-length control techniques, commonly referred to as ''globalization" techniques.

First, we will provide an overview of past and recent globalization techniques and how their theoretical analysis are performed. To begin, we shall propose a comprehensive survey encompassing historical and contemporary globalization techniques. Our review of recent methodologies will underscore issues necessitating attention. Moving forward, we shall embark upon an examination of the incompatibility between traditional globalization approaches and the requirements posed by modern large-scale optimization problems. A new globalization paradigm, developed specifically for first-order methods and distinct from conventional strategies, has emerged to address these issues. We will examine a representative algorithm from this class, followed by a discussion on the potential of extending this paradigm to second-order optimization techniques.

1.1 Standard Globalization Strategies

An algorithm that proceeds along the direction of the gradient can be globalized under appropriate assumptions. The most straightforward method hinges on knowing the Lipschitz constant of the gradient for step-length control, which may not be easy to compute. More effective globalization techniques have been proposed and developed in the literature.

1.1.1 Line Search

Line search methods involve iteratively exploring different step sizes along a selected search direction and evaluating the objective function at those points. The goal of the algorithm is to find the step size that leads to the most significant reduction in the objective function value by comparing function values at various step sizes. This step size is then used to update the current solution. The algorithm's general framework is presented below.

Algorithm 1.1.1: Generic Line Search Algorithm

Step 0: Initialization An initial point $x_0 \in \mathbb{R}^n$, an initial stepsize $\alpha_0 > 0$ as well as line search hyperparameters.

Step 1: Compute descent direction and initial stepsize Compute a descent direction d_k.

Step 2: Step length computation Compute step length α_k satisfying specific requirements.

Step 3:Iterate update Set $x_{k+1} = x_k + \alpha_k d_k$.

Increment k by 1 and go to Step 1.

For the sake of comprehensiveness, we present here a more elaborate account of Algorithm 1.1.1. The algorithm's framework demonstrates the importance of Steps 1 and 2 for

the algorithm to be successful. The selection of d_k involves choosing between a pure gradient descent step ($d_k = -\nabla_x^1 f(x_k)$) or a Newton-type direction ($d_k = B_k^{-1}\nabla_x^1 f(x_k)$), where B_k is a definite positive approximation of the Hessian.

The conditions in Step 2 are necessary to ensure the algorithm convergence because the simple requirement ($f(x_k + \alpha_k d_k) \leq f(x_k)$) is insufficient. To address this issue, we introduce the Armijo condition for the step, which is expressed as follows

$$f(x_k) - f(x_k + \alpha_k d_k) \geq -c_1\, \alpha_k\, \nabla_x^1 f(x_k)^\intercal d_k, \tag{1.1.1}$$

where $c_1 \in (0, 1)$. It should be noted that as d_k represents a descent direction, $\nabla_x^1 f(x_k)^\intercal d_k$ is negative, resulting in a decrease in function values in accordance with (1.1.1). Since f is differentiable, values of α_k that are small enough will meet the inequality. However, the algorithm's convergence may be slow. To address this issue, another requirement is added and expressed as follows

$$d_k^\intercal \nabla_x^1 f(x_k + \alpha_k d_k) \geq c_2\, d_k^\intercal \nabla_x^1 f(x_k), \tag{1.1.2}$$

with c_2 in the interval of $(c_1, 1)$. The Wolfe conditions, described in [205] and [206], comprise of two conditions denoted by (1.1.1) and (1.1.2). The second one (1.1.2) limits the step length to a minimum value. If the gradient is Lipschitz, this condition forces a lower-bound on α_k scaling in $\frac{1}{L_1}$. By implementing these conditions, the need for precise derivative information to achieve convergence is eliminated.

To uphold the aforementioned conditions, a widely used approach is to employ a backtracking line search, which is briefly described here. Initially, a large step length is taken, which is then iteratively decreased until condition (1.1.1) is satisfied.

To elaborate, the step size is reduced using the formula

$$\alpha_k = \tau^m \alpha_{k,0},$$

where $\tau \in (0, 1)$ is a constant and m is the smallest integer that satisfies (1.1.1). The scalar $\alpha_{k,0}$ represents the initial step length at iteration k. The advantage of employing this method lies in its ability to implicitely guarantee a step size that is large enough through the mechanism for updating α_k, thereby removing the necessity of explicitly considering the curvature condition (1.1.2).

Note that while our primary focus is restricted to the unconstrained problem (P), the line search approach is extensively employed in constrained optimization scenarios. In these settings, the line search technique plays a pivotal role in identifying appropriate steps while maintaining the constraints. An excellent example of this is the Interior-Point Optimizer (IPOPT) [201], which implements the line search strategy for computing valid steps. For further details on executing the line search, such as choosing hyperparameters and conducting theoretical analysis, additional information can be found in Chapter 4 of [172]. We now present an alternative paradigm that utilizes more extensively second-order information, which we detail below.

1.1.2 Trust-region and related methods

In this subsection, we give a brief overview of trust-region techniques, starting with a description of their paradigm. At each iteration x, trust-region methods construct a model for the

objective function. This model can be either linear or quadratic. The model is minimized within a trust-region of a certain radius, which is updated according to the acceptance rate. The basic framework of the algorithm is presented below.

Algorithm 1.1.2: Basic Trust-Region Algorithm (BTR)

Step 0: Initialization An initial point $x_0 \in \mathbb{R}^n$, an initial stepsize $\Delta_0 > 0$ are given, as well as the parameters $0 < \eta_1 \leq \eta_2 < 1$ and $0 < \gamma_1 \leq \gamma_2 < 1$. Compute $f(x_0)$ and set $k = 0$.

Step 1: Compute current model Build a first or second-order model $m_k(s)$.

Step 2: Step computation Compute s_k such that

$$s_k \simeq \arg\min_{\|s\| \leq \Delta_k} m_k(s) \tag{1.1.3}$$

with m_k defined in (1.1.6).

Step 3: Acceptance of the trial point Compute $f(x_k + s_k)$ and define

$$\rho_k \stackrel{\text{def}}{=} \frac{f(x_k) - f(x_k + s_k)}{(m_k(0) - m_k(s_k))}. \tag{1.1.4}$$

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

Step 4: Trust-region radius update Set

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, +\infty) & \text{if } \rho_k \geq \eta_2, \\ [\gamma_2 \Delta_k, \Delta_k] & \text{if } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k] & \text{if } \rho_k < \eta_1. \end{cases} \tag{1.1.5}$$

Increment k by 1 and go to Step 2.

We will now discuss the procedure for the aforementioned algorithm. In Step 1, the used model is usually a quadratic function, as shown below

$$m_k(s) = f(x_k) + s^\intercal \nabla_x^1 f(x_k) + \frac{1}{2} s^\intercal B_k s. \tag{1.1.6}$$

Note that $\nabla_x^1 f(x_k)$ represents the exact gradient of the function f while B_k is either an efficient approximation or exact curvature information ($\nabla_x^2 f(x_k)$).

To compute the trial step s_k, we must approximately minimize (1.1.6) subject to $\|s_k\| \leq \Delta_k$. Indeed, from the structure of problem (1.1.3), the condition on the exact minimize can be written as

$$(B_k + \lambda_k I_n) s_k = -\nabla_x^1 f(x_k), \quad \lambda_k(\Delta_k - \|s_k\|) = 0. \tag{1.1.7}$$

with λ_k a positive constant. Using the last inequality, we can develop methods that employ the factorization of matrix B_k to calculate the trial step s_k or use iterative conjugate gradient methods to compute the step. For further information on step computation, please refer to [68, Chapter 7].

We remark also from the characterization (1.1.7) that trust-region can be related to the Levenberg-Marquardt method [140, 150] that were initially proposed to globalize Gauss-Newton where the regularization is applied directly to the local quadratic approximation model

$$m_k(s) = f(x_k) + s^\intercal \nabla_x^1 f(x_k) + \frac{1}{2} s^\intercal B_k s + \frac{\lambda_k}{2}\|s\|^2. \tag{1.1.8}$$

The λ_k parameter is updated in a comparable manner to Δ_k in (1.1.5), but it is incremented when the model does not result in a decrease in the objective function value. We conclude this short digression on Levenberg-Marquardt methods and resume the examination of Algorithm 1.1.2.

After computing the trial step, we compare the reduction achieved in the objective function to that derived from the model. If the step satisfies the condition $\rho_k \geq \eta_1$, it is deemed sufficiently large and the iteration is considered successful. If not, the step is unsuccessful and the trust region is contracted. In order to ensure algorithm convergence, it is common practice to enforce the trial step decrease s_k to be greater than that obtained from the steepest descent direction. Mathematically, it writes as

$$m_k(s_k) \leq \underset{t \in \mathbb{R}^+}{\arg\min}\, m_k(-t\nabla_x^1 f(x_k)). \tag{1.1.9}$$

Trust-region methods enjoy computationally efficient variant suited for large scale optimization instances. They can be easily combined with quasi-Newton approximation of the Hessian, where first-order information is used to provide efficient second-order approximations. A commonly used approach is the L-BFGS method [39], which, when combined with a trust-region method, can tackle large-scale problems. For further discussion of quasi-Newton methods, we refer the reader to [172, Chapter 6] and the references therein. Trust-region methods have also been successful in a variety of numerical optimization domains, including Derivative-Free Optimization (DFO) [69] and Augmented Lagrangian methods [25]. For a comprehensive analysis of trust-region methods for both constrained and unconstrained optimization problems, see [68]. Recent advances in the field are reviewed in [208].

From the two previously mentioned frameworks of line search and trust-region methods, convergence of the iterate sequence can be ensured under mild conditions. Either by performing a line-search along a gradient-related direction or by upholding requirement (1.1.9) for trust-regions. Local analyses that retrieve the behavior of Newton's method were investigated [68, 172]. Of course, these algorithms have also been tested on standard nonlinear optimization benchmarks. One of the most popular is CUTEst with its various developments over the years [34, 106].

Various numerical simulations have been devised for both algorithms and different parameters may affect the numerical capability of the implemented variance. For instance, the problem's structure can be utilized to create a sparse estimate of the curvature matrix. Various preconditioning matrices can speed up the resolution of (1.1.3). Practitioners prioritize computational CPU time, which depends on the hardware and its configuration. Therefore, extensive experiments should be conducted to guarantee a just comparison of different non-

convex optimization techniques. Below, we focus on the evaluation complexity of various algorithms. Complexity analysis is a central theme throughout this thesis. In the subsequent sections, we outline its definition for problem (P) and explore this notion in the context of standard nonlinear optimization techniques.

1.2 Evaluation complexity

The evaluation complexity is determined for a given algorithm, like steepest descent or trust-region, within a general problem class with a specific termination condition. Based on the required criticality level and other function characteristics, a bound is established on the number of iterations, objective function and derivative evaluations. The evaluation complexity comes also under different names. It is also recognized as the *worst-case complexity* of an algorithm for a given class of problem or its *global convergence rate*. We now give detailed examples and convergences rates of standard methods.

1.2.1 Evaluation complexity for convex problems

We take a brief detour and shift focus to the convex problem (P) for which we provide an overview of the major results. Assume f is convex and problem (P) is well-defined, thus, an optimal value $f_\star$ exists. The optimization routines introduced in Section 1.1 may be used to address the problem. Given $\epsilon > 0$, the algorithm would have completed if

$$f(x_k) - f_\star \leq \epsilon. \tag{1.2.1}$$

If a steepest descent method with a backtracking line search mechanism as described in Algorithm 1.1.1 is employed for a function with Lipschitz gradient, it will take $\mathcal{O}\left(\epsilon^{-1}\right)$ iterations to reach (1.2.1). The dependence on the Lipschitz constant and the distance to the optimal point is concealed in $\mathcal{O}$. However, faster optimization methods that utilize previous gradient information can yield a faster convergence rate at $\mathcal{O}(\epsilon^{-1/2})$ for convex problems. In the case of strongly convex problems, where the Hessian satisfies

$$\mu\|s\|^2 \leq s^\intercal \nabla_x^2 f(x) s \leq L_1\|s\|^2 \tag{1.2.2}$$

and μ is a strictly positive constant, the criterion for convergence is typically the distance to the optimal point $\|x_k - x_\star\| \leq \epsilon$. In this scenario, the rate of convergence is $\mathcal{O}\left(\frac{L_1}{\mu}\log(\epsilon)\right)$. Employing rapid convex optimization algorithms allows us to enhance the constants to $\mathcal{O}\left(\sqrt{\frac{L_1}{\mu}}\log(\epsilon)\right)$. Therefore, in addition to the dependence on ϵ, one has to take also into account dependencies on other problem constants. The detailed findings and extensive analysis of convex optimization algorithms are available in [165]. Hereafter, we will focus solely on non-convex settings for the rest of the present thesis.

1.2.2 Evaluation complexity for nonconvex problems

For nonconvex optimization (P), global optima are *generally* not achievable: only approximate stationary points can be reached in practice. In the case of the unconstrained nonconvex

problem, the associated first-order optimality condition is expressed as

$$\|\nabla_x^1 f(x_\epsilon)\| \leq \epsilon \tag{1.2.3}$$

where $\epsilon \in (0, 1]$ is the requested gradient threshold. Note that this criterion, that is an approximate necessary condition for optimality, has also been examined for the convex case and can be applied to stochastic optimization problems where (1.2.3) holds in expectation. For second-order approximate necessary optimality condition for problem (P) is given by

$$\|\nabla_x^1 f(x_\epsilon)\| \leq \epsilon_1, \quad \max\left(0, -\lambda_{\min}(\nabla_x^2 f(x_\epsilon))\right) \geq -\epsilon_2, \tag{1.2.4}$$

where ϵ_1 and ϵ_2 are in $(0, 1]$. The reader may be more familiar with (1.2.3) as it is related to the optimality condition of convex optimization, and gradient descent schemes algorithms generally reach an iterate satisfying only (1.2.3). However, it may be worthwhile to enforce more stringent requirements on the final iterate. Indeed, several important problems such as phase retrieval [41], matrix completion [97] or specific regression tasks [144], a local minima satisfying (1.2.4) with ($\epsilon_1 = 0$ and $\epsilon_2 = 0$) yields a global minima.

We will now restate previously established convergence rate of standard optimization methods with globalization strategies previously exposed in Section 1.1. For the backtracking line search steepest descent method applied to a Lipschitz gradient function, achieving a point satisfying (1.2.3) requires $\mathcal{O}\left(\epsilon^{-2}\right)$ evaluations of the function and its derivative. The $\mathcal{O}()$ term conceals dependencies with respect to both the Lipschitz gradient constant L_1, and the value of $f(x_0)$. The final bound is sharp, as there exists a function, f, within the class of interest that attains the worst-case bound [57]. It is hoped that nonlinear optimization techniques which exploit curvature information will achieve rates faster than first-order rates, thereby providing an explanation for their outstanding practical performance on a wide range of problems. Unfortunately, when *globalized*, the plain Newton's method has the same rate as the steepest descent method for Lipschitz Hessian nonconvex functions. Indeed, [57, Theorem 3.1.1] proposes a function in $\mathbb{R}^2$ that reaches the same rate. Similarly, other efficient globalization techniques encounter the same issue, whether the generalization strategy is line search or trust-region, efficient approximation of the Hessian, or exact information. The same rate $\mathcal{O}\left(\epsilon^{-2}\right)$ occurs. The book [57] offers a detailed and comprehensive presentation of the results just mentioned. When seeking a second-order stationary point, it is necessary to utilize information pertaining to the curvature and perform potentially costly numerical computations with the Hessian. Specifically, a direction of *negative curvature* may be employed. For a wide range of second-order algorithms, the convergence rate required to ensure the second part of (1.2.4) is $\mathcal{O}\left(\epsilon_2^{-3}\right)$. See [57] for more discussions on second-order algorithms that can reach (1.2.4).

While the second-order optimization schemes introduced earlier have demonstrated remarkable efficacy in terms of rapid local convergence, as exemplified in past comprehensive reviews [172, 68], their global convergence rates have remained comparable to those achieved by first-order methods. However, recently, a notable breakthrough has emerged in the form of an efficient second-order optimization technique known as 'cubic regularization'. It is to this method that we now turn our attention, aiming to provide a comprehensive discourse on its principles and applications.

1.3 Optimal Second-Order methods

1.3.1 Cubic Regularization

First, we briefly outline the main mechanism behind the convergence of first-order optimization schemes of the Lipschitz gradient function, which depends on the following local quadratic bound

$$f(x+s) \leq f(x) + s^{\intercal}\nabla_x^1 f(x) + \frac{L_1}{2}\|s\|^2, \tag{1.3.1}$$

where the last bound is derived from standard Taylor bounds. For example, for steepest descent, the decrease of the function value is ensured whenever $\alpha \leq \frac{1}{2L}$. The cubic regularization method extends this idea to the second-order, using a quadratic model augmented by a cubic regularization term to obtain a local upper bound. For function with Lipschitz Hessian, the latter bound becomes cubic and is written as

$$f(x+s) \leq m_k(s) = f(x) + s^{\intercal}\nabla_x^1 f(x) + \frac{1}{2}s^{\intercal}\nabla_x^2 f(x)s + \frac{L_2}{6}\|s\|^3. \tag{1.3.2}$$

Thus, at each iteration x_k, [169] proposes to minimize exactly the local cubic upper bound $s \mapsto m_k(s)$ and thus assumes knowledge of Lipschitz smoothness. For this new optimization scheme, the numerical routine proposed in [169] requires $\mathcal{O}\left(\epsilon^{-3/2}\right)$ iterations to reach a point where (1.2.3) holds. We also get a guarantee of convergence to the second-order stationary point. The number of iterations required is written as $\mathcal{O}\left(\max\left(\epsilon_1^{-3/2}, \epsilon_2^{-3}\right)\right)$. As a matter of historical fact, the use of (1.3.2) was first proposed by [120] in a technical report and later independently rediscovered by [169]. The latter routine, however, requires precise information about the Lipschitz function. [51, 50] propose an adaptive cubic regularization method that adapts to the local Lipschitz smoothness. We give a detailed outline of the proposed algorithm on the next page.

It is natural that the condition imposed in Step 2 of Algorithm 1.3.1 will play a crucial on the final convergence rate. We detail below some conditions and their associated rates. Similar to trust-region methods, we can impose a *Cauchy condition* on the computed step, which is expressed as

$$m_k(s_k) \leq \underset{t \in \mathbb{R}^+}{\arg\min}\, m_k(-t\nabla_x^1 f(x_k)). \tag{1.3.7}$$

Unfortunately, under this condition, the convergence rate of ARC is the same as that of standard gradient descent to approximate the first-order critical point, i.e: $\mathcal{O}\left(\epsilon^{-2}\right)$. To obtain the improved convergence rate proved by [169], the authors impose the following condition on the step

$$\|\nabla_s^1 m_k(s_k)\| \leq \kappa_\theta \min\left(1, \|s_k\|\right)\|g_k\| \tag{1.3.8}$$

with $\kappa_\theta > 0$. Note that the previous condition only requires computing an approximate minimum of the cubic model (1.3.4). We leave the discussion of subroutines that compute a trial step for later in the manuscript.

Algorithm 1.3.1: Adaptative Regularization Algorithm (ARC)

Step 0: Initialization An initial point $x_0 \in \mathbb{R}^n$, a regularization parameter σ_0 is given. The constants η_1, η_2, γ_1, γ_2, γ_3, and $\sigma_{\min}$ are also given such that

$$\sigma_{\min} \in (0, \sigma_0], 0 < \eta_1 \leq \eta_2 < 1 \quad \text{and} \quad 0 < \gamma_1 < 1 < \gamma_2 < \gamma_3. \tag{1.3.3}$$

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

Step 1: Check for termination Evaluate $g_k = \nabla_x^1 f(x_k)$ and $H_k = \nabla_x^2 f(x_k)$. Terminate with $x_\epsilon = x_k$ if either (1.2.3) or (1.2.4) is satisfied.

Step 2: Step calculation Compute a step s_k which sufficiently reduces the model m_k defined below

$$m_k(s) = f(x_k) + g_k^\intercal s + \frac{1}{2} s^\intercal H_k s + \frac{\sigma_k}{6}\|s\|^3. \tag{1.3.4}$$

Step 3: Acceptance of the trial point Compute $f(x_k + s_k)$ and define

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(0) - m_k(s_k)}. \tag{1.3.5}$$

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

Step 4: Regularization parameter update Set

$$\sigma_{k+1} \in \begin{cases} [\max(\sigma_{\min}, \gamma_1\sigma_k), \sigma_k] & \text{if } \rho_k \geq \eta_2, \\ [\sigma_k, \gamma_2\sigma_k] & \text{if } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_2\sigma_k, \gamma_3\sigma_k] & \text{if } \rho_k < \eta_1. \end{cases} \tag{1.3.6}$$

Increment k by one and go to Step 1.

1.3.2 Other ARC Related Variants

Other subsequent modifications have been proposed for easier analysis [29] or to tackle more complicated problems [111]. For example, [29] proposes a second-order (among a larger class of optimization methods) algorithm that follows the layout of the ARC algorithm and only changes the gradient method requirement, writing it as

$$\|\nabla_s^1 m_k(s_k)\| \leq \kappa_\theta \frac{\sigma_k}{2}\|s_k\|^2. \tag{1.3.9}$$

In [111], another requirement was devised to extend cubic regularization for a boarder class of problems

$$\|g_k + H_k s_k\| \leq \kappa_\theta \frac{\sigma_k}{2}\|s_k\|^2 \tag{1.3.10}$$

where $\kappa_\theta \geq 1$. Note that we can consider a modification of the denominator in (1.3.5) and consider the local quadratic approximation instead ($g_k^\intercal s_k + \frac{1}{2} s_k^\intercal H_k s_k$).

Since the development of the algorithm 1.3.1, several extensions have been proposed, each

striving to achieve this increased convergence rate.

Some of these innovative approaches use line search with different steps at each iteration [184, 22]. Others exploit the underlying structure of linear conjugate methods [75, 146]. Meanwhile, a separate set of algorithms proposes trust-region techniques that use specific schemes, as shown in the work of [74] and [121]. These algorithms are well suited to address the challenges posed by large-scale problems.

In addition, a parallel research path consider potentially more computationally intensive methods. These algorithms require the factorization of the Hessian matrix, as shown in [26, 27, 90] and related references. However, all of these methods rely on finely tuned subroutines to enforce descent of the desired order [26, 27]. Even for the standard ARC presented in Algorithm 1.3.1, computing a step involves more sophisticated subroutines. Indeed, computing a trial step of (1.3.4) can not be reformulated as a linear system since the first-order condition writes as

$$(\nabla_x^2 f(x_k) + \frac{\sigma_k}{2}\|s_k\| I_n) s_k = -\nabla_x^1 f(x_k). \tag{1.3.11}$$

The additional $\frac{\sigma_k}{2}\|s_k\|$ introduces implicitness in the linear system. As noted by [156], the improvement in complexity has been achieved by trading the simple Newton step, which requires only the solution of a single single linear system for more complex or slower procedures, such as secular iterations, possibly using Lanczos preprocessing [50, 51] (see also [57, Chapters 8 to 10]) or (conjugate) gradient descent [43]. In contrast, for the Newton step (where we only need to solve a linear system), a wide range of numerical tools are available, including direct methods [76], iterative approaches [186], and parallel computing variants [180]. And so a first question arises

Question 1 *Can a second-order method that "mostly" employs a regularized Newton method for all iterations be developed for nonconvex optimization with convergence rate close to the one of ARC method?*

In addition to the development of new efficient second-order methods, there have been efforts to achieve the $\mathcal{O}\left(\epsilon^{-3/2}\right)$ rate under milder conditions. Such extensions have incorporated inexact derivatives [211, 214], though they necessitate an exact function value for adjusting the regularization parameter σ_k. Non-Euclidean geometries have been proposed in [111] and [1] devised an extension to Riemannian geometry. Probabilistic variants [15, 46] and stochastic ones [195] have also been studied. Other extensions of the cubic regularization methods have focused on adding well-established paradigms of nonlinear optimization to the ARC algorithm. To name a few, non-montone acceptance schemes of the trial iterates s_k [24, 170, 171], quasi-Newton approximation of curvature [19], the use of an iteration dependent norm [21, 152]. Extensions specific to the convex case only, such as acceleration schemes, have also been studied in [164] and [64].

To pursue faster algorithms, higher-order derivatives are employed to improve the quality of the local trial step. As higher-order algorithms represent an emerging area in nonlinear optimization, we briefly deviate from this introduction to outline definitions pertaining to higher-order derivatives and their associated approximation bounds.

1.3.3 Higher Order Models

1.3.3.1 High-Order Tensors and approximation bounds

$\mathcal{L}(\mathcal{V}^{\otimes m};\mathbb{R})$ denotes the space of multilinear continuous functionals from $\mathcal{V}\times\mathcal{V}\cdots\times\mathcal{V}$ to to $\mathbb{R}$ and $\mathcal{L}^m_{sym}(\mathcal{V}^{\otimes m};\mathbb{R})$ is the subspace of $\mathcal{L}^m(\mathcal{V}^{\otimes m};\mathbb{R})$, which is m-linearly symmetric. For a functional f defined from $\mathcal{V}$ to $\mathbb{R}$, that is p times Fréchet differentiable, $\nabla^k_x f(x)\in\mathcal{L}^k_{sym}(\mathcal{V}^{\otimes k};\mathbb{R})$. For $S\in\mathcal{L}^m_{sym}$, $S[v_1,v_2\ldots,v_m]\in\mathbb{R}$ denotes the result of applying S to $v_1,\ldots,v_m$. $S[v]^m$ is the result of applying result of applying S to m copies of v and $S[v]^l\in\mathcal{L}^{m-l}_{sym}(\mathcal{V}^{\otimes m-l};\mathbb{R})$ is the result of applying to l copies of v.
We define the norm in $\mathcal{L}^m_{sym}(\mathcal{V}^{\otimes m};\mathbb{R})$ as

$$\begin{aligned}\|S\| &\stackrel{\text{def}}{=} \sup_{\|v_1\|_{\mathcal{V}}=\cdots=\|v_m\|_{\mathcal{V}}=1} |S[v_1,\ldots,v_m]| \\ &= \sup_{\|v\|_{\mathcal{V}}=1} |S[v]^p|.\end{aligned} \tag{1.3.12}$$

The last inequality is true for any $S\in\mathcal{L}^m_{sym}(\mathcal{V}^{\otimes m};\mathbb{R})$, see [168, Appendix 1] for a proof.

The pth derivative tensor $\nabla^p_x f(x)\in\mathcal{L}(\mathcal{V}^p;\mathbb{R})$ is globally *Hölder continuous*, that is, there exist constants $L_p>0$ and $\beta\in(0,1]$ such that

$$\|\nabla^p_x f(x)-\nabla^p_x f(y)\|\le L_p\|x-y\|^\beta_{\mathcal{V}},\quad \text{for all}\quad x,y\in\mathcal{V}. \tag{1.3.13}$$

with some positive continuous L_p. For $p=1$ and $\beta=1$, we get the class of Lipschitz continuous gradient function, and for $p=2$ and $\beta=1$, the Lipschitz continuous Hessian one. We remind that the class of p times Fréchet differentiable function with β-Hölder pth derivative is denoted as $\mathcal{C}^{p,\beta}(\mathcal{V};\mathbb{R})$.

The pth order Taylor series at x writes as

$$T_{f,p}(x,s)\stackrel{\text{def}}{=} f(x)+\sum_{l=1}^{p}\frac{1}{l!}\nabla^l_x f(x)[s]^l. \tag{1.3.14}$$

We now restate error bound between the function f and its polynomial approximation, the latter being standard tools of Numerical Analysis.

Lemma 1.3.1 *Suppose that $f\in\mathcal{C}^{p,\beta}(\mathcal{V};\mathbb{R})$ holds. Then*

$$|f(x+s)-T_{f,p}(x,s)|\le\frac{L_p}{(p+\beta)!}\|s\|^{p+\beta}_{\mathcal{V}}, \tag{1.3.15}$$

$$\|\nabla^1_x f(x+s)-\nabla^1_s T_{f,p}(x,s)\|_{\mathcal{V}'}\le\frac{L}{(p+\beta-1)!}\|s_k\|^{p+\beta-1}_{\mathcal{V}}, \tag{1.3.16}$$

and

$$\|\nabla^2_x f(x+s)-\nabla^2_s T_{f,p}(x,s)\|\le\frac{L}{(p+\beta-2)!}\|s_k\|^{p+\beta-2}_{\mathcal{V}}. \tag{1.3.17}$$

Proof. By Taylor's theorem applied to $\psi : t \in [0,1] \mapsto f(x+ts)$ of order p, we derive that

$$f(x+s) = f(x) + \sum_{i=1}^{p-1} \frac{1}{i!} \nabla_x^i f(x)[s]^i + \int_0^1 \frac{\nabla_x^p f(x+ts)[s]^p}{(p-1)!}(1-t)^{p-1}\,\mathrm{dt}$$

Using the expression of $T_{f,p}$ in (1.3.14) and the fact that $\int_0^1 (1-t)^{p-1}\,dt = \frac{1}{p}$, we derive that

$$\begin{aligned}
|f(x+s) - T_{f,p}(x,s)| &= \left| \int_0^1 \frac{(1-t)^{p-1}}{(p-1)!} \nabla_x^p f(x+ts)[s]^p\,dt - \frac{1}{p!}\nabla_x^p f(x)[s]^p \right| \\
&= \left| \int_0^1 \frac{(1-t)^{p-1}}{(p-1)!} \left(\nabla_x^p f(x+ts)[s]^p - \nabla_x^p f(x)[s]^p\right)\,dt \right| \\
&\leq L_p \|s\|^{p+\beta} \int_0^1 \frac{(1-t)^{p-1} t^\beta}{(p-1)!}\,dt = \frac{L_p}{(p+\beta)!}\|s\|^{p+\beta}.
\end{aligned}$$

Where we used (1.3.14) for the inequality thus obtaining (1.3.15). For the last idendity $\int_0^1 \frac{(1-t)^{p-1}t^\beta}{(p-1)!}\,dt = \frac{1}{(p+\beta)!}$, we refer the reader to [55, Equation (A.1)]. As for the identities (1.3.16) and (1.3.17), we apply the previous reasoning to functions $\langle \nabla_x f(.), v\rangle$ and $\nabla_x^2 f(.)[v]^2$ with the direction v fixed. □

1.3.3.2 High Order methods

In pursuit of an enhanced convergence rate, one can turn to the usage of higher-order derivatives also denominated as tensors. These techniques, commonly known as tensor methods in the convex optimization literature [166], or as pth adaptive regularization methods [29] for nonconvex optimization. We now briefly present the mechanism of the adaptive regularization algorithms denominated ARp.

At iteration k, we construct a surrogate model of the objective function by adding a regularization term to the pth order Taylor approximation. For example, when dealing with $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n;\mathbb{R})$, we apply regularization to the Taylor series as follows

$$m_k(s) \overset{\text{def}}{=} T_{f,p}(x_k, s) + \frac{\sigma_k}{(p+\beta)!}\|s\|^{p+\beta}. \tag{1.3.18}$$

$T_{f,p}(x,s)$ is the pth order Taylor expansion of functional f at x truncated at order p defined at (1.3.14). The σ_k term guarantees that $m_k(s)$ is bounded below and thus makes the procedure of finding a step s_k by (approximately) minimizing $m_k(s)$ well-defined. The σ_k constant aims at matching the Hölder parameter of the tensor derivative $\nabla_x^p f$. When $p = 1$ and $\beta = 1$, we obtain a line search algorithm with a step size equal to $1/\sigma_k$. The following figure illustrates the need for regularization when $p = 3$ and $\beta = 1$. In this case, the local third order is a loose approximation of the function plotted in blue, while the quartic model plotted in green gives a tighter approximation.

When computing a step s_k, it is necessary to ensure that the computed step provides local decrease, namely that

$$m_k(s_k) < m_k(0) = f(x_k). \tag{1.3.19}$$

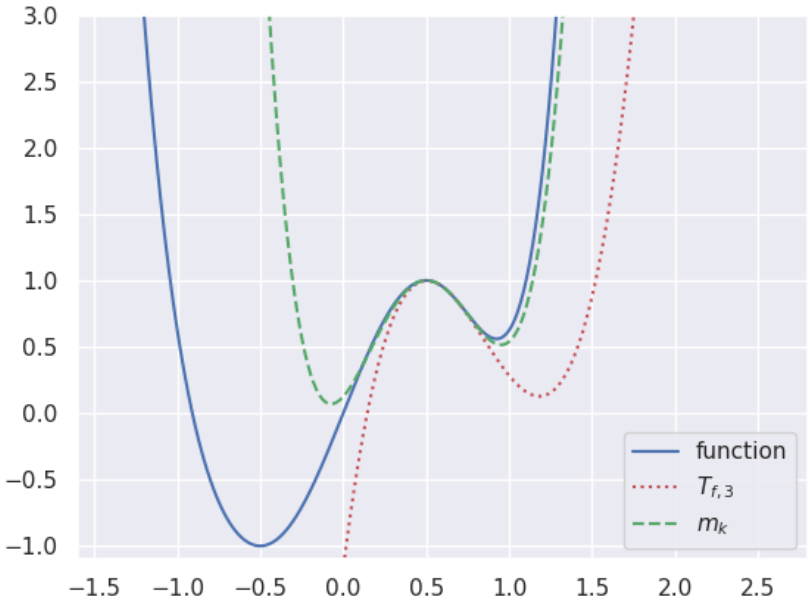

Figure 1.1: Plot of m_k and $T_{f,2}$ for the function $f(x) = \sin(\pi x) + 10(x-0.5)^4\mathbb{1}_{(0.5,+\infty)} + 10(x+0.5)^4\mathbb{1}_{(-\infty,0.5)}$ with $\sigma_k = 240$.

To ensure convergence to an approximate first-order stationary point, it is also required that

$$\|\nabla_s^1 m_k(s_k)\| \leq \frac{\theta\|s_k\|^{p+\beta-1}}{(p+\beta-1)!}, \tag{1.3.20}$$

with θ strictly positive. Note that this condition only imposes an approximate minima of m_k. After computing the trial step, we compute an acceptance ratio as in the trust-region Algorithm 1.1.2 which write as

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(0) - m_k(s_k)}. \tag{1.3.21}$$

If the ratio is positive, the step produces a decrease in the function value, it is accepted and we move on by forming a new model. Otherwise, we have not sufficiently regularized the problem and we increase σ_k in a similar fashion to (1.3.6).

Under the previous conditions on the step, computation of (1.3.21) and a suitable update rule of the regularization parameter σ_k, convergence rate to reach an iterate that verifies (1.2.3) is $\mathcal{O}\left(\epsilon^{-(p+\beta)/(p+\beta-1)}\right)$ for $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n;\mathbb{R})$ [29]. Therefore, when designing new pth adaptive regularization methods, we aim to recover the aforementioned rate, as its optimality was shown in [45]. Several extensions of adaptive regularization have been developed. For example, when convergence to a second order stationary point is required, another condition

is imposed on the curvature model, which is written as

$$\max\left(0, -\lambda_{\min}(\nabla_s^2 m_k(s_k))\right) \leq \frac{\theta\|s_k\|^{p+\beta-2}}{(p+\beta-2)!} \tag{1.3.22}$$

with $\theta > 0$. This new condition gives a $\mathcal{O}\left(\epsilon_1^{-(p+\beta)/(p+\beta-1)}, \epsilon_2^{-(p+\beta)/(p+\beta-2)}\right)$ in the worst case to reach an iterate that satisfies (1.2.4) [54]. In the context of convex optimization, subsequent efforts have focused on the introduction of inexact variants [80, 167], acceleration schemes adapted to uniformly convex functions [96, 37], extension to the case of composite objective functions [128, 127, 108] and further to "general composite" optimization problems [162, 81]. Even "super-universal" schemes that are able to adapt to both the smoothness of the function and the Hölder exponent associated with it [84, 107]. The usage of high-order tensors have also been to tackle the boarder class of variational inequalities problems, see [38, 142, 129] and the references therein.

In the nonconvex case, the paradigm has been extended to tackle a larger class of problems including multiobjective minimization [40], constrained optimization under various assumptions [151, 63] and probabilistic or inexact derivatives as done in [12, 15]. Practical implementations of regularized third-order method for nonconvex function have been proposed in [31, 47] with promising initial numerical results. Coordinate variant where only a subset of variables are updated at each iteration have been considered in [28, 3].

For all the proposed NLP contributions, the algorithm's analysis is performed under the usage of the Euclidean norms in the primal space. However, we know that standard Euclidean norm may be ill-suited for a certain class of problems. As an example, in convex optimization, Bregman divergence [36] can be used to measure the smoothness of the gradient of the function. One popular algorithm proposed in that framework is Mirror Descent, see [10]. Moreover, choosing another norm than the usual Euclidean one may enhance the dependency of smoothness constant w.r.t the input dimension n. For high-order methods, this issue has been tackled by [111] where adaptive regularization algorithms are studied with $\mathcal{V} = (\mathbb{R}^n, \|.\|_p)$. The proposed analysis hinges crucially on a new acceptance condition instead of the usual (1.3.20). It therefore seems a natural extension to extend the high-order tensor method to other types of $\mathcal{V}$. Of course, the following questions arise

Question 2 *Can a high-order tensor method be developed for Banach space? In particular, How to ensure the existence of a suitable step at a given iteration?*

After this broad overview of past and recent nonlinear optimization techniques and some plausible extensions that may be proposed, we now expose some pitfalls of the optimization techniques that have been presented for problems arising in modern applications.

1.4 On the necessity of new approaches

1.4.1 An Overview of Modern Optimization Challenges

Context For modern optimization problems, especially Machine Learning [32], (P) becomes

$$\min_{x\in\mathbb{R}^n} \frac{1}{p}\sum_{i=1}^{p} f_i(x, y_i, a_i) \tag{P-ML}$$

where both p and n may exceed to millions and f_i may be nonconvex. The pairs (a_j, y_j) are independent and identically distributed random variables coming from an a priori unknown distribution $\mathcal{D}$. The formulation of (P-ML) encompasses *supervised learning*. A particular subclass of interest are Deep Learning *DL* problems where f_i simulates the output of an artificial neural network. Deep Learning has achieved great empirical success in various fields such as computer vision [138], linguistics [199], physics simulation [181] and biology [131]. For a deeper understanding of neural networks, we direct readers to the comprehensive monograph [101] and the references therein. We give below some examples arising in (P-ML).

- **Mean Least Square Regression** For $j \in \{1, \dots, p\}$, we define f_j as

  $$f_j(x, y_j, a_j) = (y_j - g(x, a_j))^2$$

  where g takes both the parameter the and feature as input and outputs a scalar. g can be either the scalar product in $\mathbb{R}^n$ yielding the least-square problem or the output of a multilayer perceptron.

- **Binary Regression** For $j \in \{1, \dots, p\}$, define

  $$f_j(x, y_j, a_j) = -y_j \log\left(\frac{1}{1+\exp(g(x, a_j))}\right) - (1-y_j)\log\left(\frac{1}{1+\exp(g(x, a_j))}\right).$$

  In the previous case, $y_j \in \{0.1\}$ encodes the class to be predicted. Again, g returns a scalar. It can be either the dot product in $\mathbb{R}^n$, which gives binary logistic regression, or the output of an artificial neural network.

We now discuss some of the challenges of using classical nonlinear optimization techniques to solve problems of the form (P-ML).

The first point is that access to the exact gradient and Hessian is impossible due to the sheer amount of memory required. In fact, one of the first successful neural network *Alexnet* [138] has p in (P) scaling up to 12 million input images. In addition, the variable x to be optimized has up to 60 million components. The approximate Hessian would require thousands of terabytes just for storage. To tackle this problem, we can sample a gradient $\{\nabla_x^1 f_j(x, y_j, a_j)\}$ for $j \in \{1, \dots, p\}$, and since (a_j, y_j) are assumed to be i.i.d. samples from the same distribution, we get that

$$\mathbb{E}_{(a_j,y_j)\sim\mathcal{D}}\left[\nabla_x^1 f_j(x, a_j, y_j)\right] = \nabla_x^1 f(x). \tag{1.4.1}$$

From the previous characterization, we can use the SGD method developed in [183] which writes as

$$x_{k+1} = x_k - \alpha_k \nabla_x^1 f_j(x_k, y_j, a_j) \quad \text{and} \quad \sum_{i=0}^{\infty} \alpha_i = \infty, \ \sum_{i=0}^{\infty} \alpha_i^2 < \infty. \tag{1.4.2}$$

Under appropriate conditions, this algorithm can converge to a first-order stationary point. However, its rate of convergence is than the first-order deterministic method. To obtain an iteration that verifies (1.2.3) in expectation, $\mathcal{O}\left(\epsilon^{-4}\right)$ iterations are required in the worst case. We refer the reader to the two papers [134, 98] that obtained the aforementioned rate for stochastic gradient descent under various conditions on the variance of the sampled first-order derivatives. Even though (1.4.2) is slower than Algorithm 1.1.1 (with $d_k = -\nabla_x^1 f(x)$) in the worst case, the trade-off offers significant advantages due to the substantial cost savings per iteration and the fact that low accuracy solutions are often sufficient. To reduce the variance of the algorithm and to obtain a more accurate estimate of the current gradient, we consider a *minibatch* in the update rule of (1.4.2). Minibatch training involves partitioning the data set into smaller subsets or batches. During each optimization iteration, a batch is randomly selected and its approximate loss and gradients are computed. Let $i_1, \cdots, i_m$ denote a batch of size m. The approximate gradient is then given by

$$\nabla_x^1 f(x) \simeq \frac{1}{m} \sum_{i \in \{i_1, \cdots, i_m\}} \nabla_x^1 f_i(x, a_j, y_j),$$

where the last inequality would hold in expectation as (1.4.1). Even if modern processors allow the use of larger batches or the use of multiple communicating devices [136, 202], there is still the problem of computing α_k. In practice, several runs are performed under different α_k to ensure good final performance with the use of other heuristics. A popular one is the cosine schedule [148] of the parameter α_k. One may wonder if it is possible to use classical nonlinear optimization schemes for problems in the class (P-ML) to adjust the stepsize α_k. Unfortunately, we expose some failures of this strategy in the next paragraph.

1.4.2 On the Usage of Function Value in Inexact Optimization

First, from the presentation of the BTR algorithm 1.1.2, it is clear that Step 3 plays a pivotal role in determining whether to accept a trial step, thereby ensuring progress, or to reject it, thereby reducing the trust region step. One critical assumption that has been thoroughly examined is the use of a fully quadratic model, which we formally state below.

Definition 1.4.1 *Given a function and constants κ_{ef}, $\kappa_{eg} > 0$, an iteration k of the* BTR*, a model function m_k is a* (κ_{ef}, κ_{eg})− fully linear model*, if for all $\|s\| \leq \Delta_k$,*

$$|m_k(s) - f(x_k + s)| \leq \kappa_{ef} \Delta_k^2, \tag{1.4.3}$$

$$\|\nabla_s^1 m_k(s) - \nabla_x^1 f(x_k + s)\| \leq \kappa_{eg} \Delta_k. \tag{1.4.4}$$

The latter condition was first proposed in the work of [7] to prove the convergence of probabilistic trust-region methods to a first-order stationary point. The aforementioned analysis was subsequently extended and refined in a number of following works [113, 33, 62, 42]

to derive the complexity rates. For example, to ensure second-order convergence, the inequalities (1.4.4) and (1.4.3) are strengthened to Δ_k^3 and Δ_k^2, respectively. Furthermore, a condition is imposed on the error bound between the model and function Hessians, which should be of the order of Δ_k [113]. Of course, when these conditions are extended to the stochastic settings, we require that (1.4.3) and (1.4.4) hold with some probability [33]. The requirements of Definition 1.4.1 have shown their versatility and have been incorporated into other nonlinear optimization paradigms. As an example, the series of papers [14, 16, 17] used conditions (1.4.3) and (1.4.4) with inexact restoration ideas [30] to analyze numerical routines taylormade for problem (P-ML).

The conditions proposed in Definition 1.4.1 have also been adapted to the case where a gradient descent is performed, namely that $\Delta_k = \alpha_k \|\nabla_x^1 f(x_k)\|$ where α_k is the current step size. The two conditions are called *stochastic gradient norm conditions* [42] write for the approximate gradient $\overline{\nabla_x^1 f}(x_k)$ and function $\overline{f}(x_k)$ values as

$$|\overline{f}(x_k) - f(x_k)| \leq \kappa_{\text{ef}} \alpha_k^2 \|\nabla_x^1 f(x_k)\|^2, \quad \|\overline{\nabla_x^1 f}(x_k) - \nabla_x^1 f(x_k)\| \leq \kappa_{\text{eg}} \alpha_k \|\nabla_x^1 f(x_k)\|. \quad (1.4.5)$$

From Definition 1.4.1 or the bounds of (1.4.5), we see some of the challenges associated with using standard trust-region or line search methods for modern problems. It becomes clear that as Δ_k decreases to match the local smoothness of the function, the error required on the function, as given in (1.4.3), must be an order of magnitude higher than that in the gradient (1.4.4). The same reasoning applies to α_k as shown in (1.4.5). In addition, in certain analyses, the value of κ_{ef} in (1.4.3) must be adjusted to fit within a certain range of values, as shown in the analyses in [62, 175].

We now present a figure that visually illustrates the challenges posed by the two requirements outlined in Lemma 1.4.1. We have considered a simple line search method that satisfies the conditions for all iterations (1.4.5) for a one-dimensional real function. We set κ_{ef} and κ_{eg} to 0.05 and 10 respectively.

As can be seen in Figure 1.2, the required error on the function quickly decreases to a small value. This phenomenon is a consequence of both the choice of κ_{ef} and the role played by α_k^2 in the bound given in (1.4.3). The error on the gradient, on the other hand, remains approximately of the same order as the true gradient value. This property proves to be advantageous in practical applications, as it makes sense that the error on the gradient should match the desired level of accuracy.

Even within a fully stochastic framework, where we only impose inequalities that hold in expectation for the function value, the requirements are still constraining. This is because it necessitates the consideration of all values of $\{1, \ldots, p\}$ when computing the function value of (P-ML). An illustrative example can be found in [175].

It's worth noting that a new line of work has recently proposed to analyze probabilistic line search under less restrictive assumptions on the function value approximation. For example, [130] studied a probabilistic gradient descent line search with a "light-tail" error bound on the function proxy and a bounded expectation error. The analysis by [200] examines an Armijo line search under the assumption that each f_i has a Lipschitz gradient, which is a significantly stronger assumption. Another approach that uses the function value to compute α_k is the *Polyak* stepsize [178], which assumes that knowledge of the optimal function value is available. This approach has recently gained attention, resulting in a number of papers [147, 187, 174] that adapt the *Polyak* stepsize to problems of the form (P-ML). It's important to note, however, that this analysis is limited to convex functions.

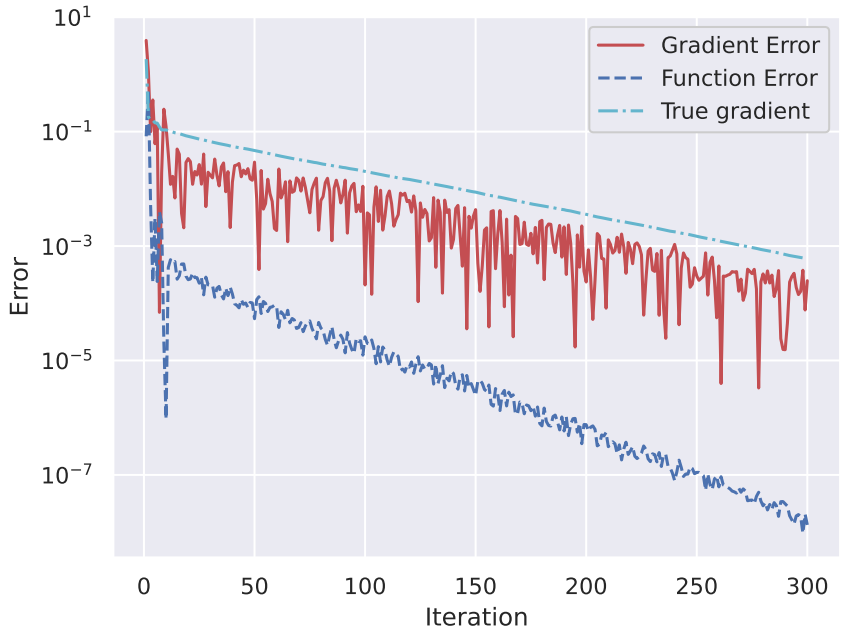

Figure 1.2: an optimization run that satisfies both conditions (1.4.3) and (1.4.4).

As we can see from this overview, vanilla optimization schemes are not well suited for problems of the form (P-ML), and modern attempts often leave a significant gap between theory and practice, or focus only on the restrictive convex settings. Therefore, it became necessary to develop optimization schemes capable of making progress even without access to function values. The class of *adaptive gradient methods* satisfies this requirement, which we outline below.

1.4.3 A New Analysis of Adaptive Gradient Methods

To tackle problems of form (P-ML), the algorithm Adagrad [88, 154] was first proposed in online learning. The latter topic is not a focus of this thesis, for more details we refer the reader to the recent survey of [125]. In the following, we present the Adagrad algorithm when used to solve (P-ML) and give some reasons for its success, perform a survey of other adaptive gradient methods, and discuss our contributions.

Notice from the above algorithm that, unlike Algorithm 1.1.2, there is no calculation of an acceptance ratio, and the step is always accepted. Therefore, all the required conditions on the function values (such as (1.4.3) for trust-region methods) are unnecessary. Thus, only assumptions about the inexact gradient are required. Based on this observation, we coin the term **OFFO** (Objective Free Function Optimization) for the class of algorithms that do not evaluate the objective function and achieve convergence. These algorithms don't require strict assumptions about the function values used. They also differ from standard nonlinear optimization [172, 68] in that they are inherently nonmonotone. It turns out that

Algorithm 1.4.1: AdaGrad

Step 0: Initialization An initial point x_0, a parameter $\varsigma \in (0, 1)$, a step size α. Set $k = 0$ and $w_k \in \mathbb{R}^n$ to zero.

Step 1: Sample gradient Sample stochastic gradient $\overline{\nabla_x^1 f}(x_k)$ from (P-ML).

Step 2: Update weights Set

$$w_k = \sqrt{\varsigma + \sum_{j=0}^{k} \overline{\nabla_x^1 f}(x_j) \odot \overline{\nabla_x^1 f}(x_j)}. \tag{1.4.6}$$

Step 3: Iterate update Set

$$x_{k+1} = x_k - \alpha(\mathrm{diag}(w_k))^{-1} \overline{\nabla_x^1 f}(x_k). \tag{1.4.7}$$

Increment k by one and go to Step 1.

first-order OFFO methods (i.e., OFFO methods that use only gradients) have been around for some time and have proven to be popular and useful in fields such as machine learning or sparse optimization, and all adaptive gradient methods, including Adagrad presented in Algorithm 1.4.1, belong to this class.

The shown component-wise weighting of the gradient descent direction has proven remarkably successful in practice and has led to the to the proposal of many variants. Among them, RMSprop [192], ADADELTA [219], Adam [135] uses an exponentially decreasing moving average when updating the weights instead of the non-decreasing technique proposed in [88]. In particular, Adam [88] has shown excellent performance in deep learning applications, as shown in a recent numerical study by [189]. Closely related variants have also been shown to be practically competitive in [61]. However, its potential divergence was shown in [182] where AMSGrad, a non-decreasing convergent scheme, was proposed as an alternative. The original theoretical analysis of Adagrad [88, 154] was subsequently refined for strongly convex functions [160] with a slight modification of the update rule, extended for minmax problems [213], further adapted for online use [173], modified to ensure privacy [4] enhanced with a line search for specific machine learning problems [200], or with an accelerated gradient-like update [133]. Adagard norm where a global (rather than component-wise, as in algorithm 1.4.1) weight is used for all components. As a final word for this review, we emphasize that we have not provided here a comprehensive and thorough review of all adaptive gradient methods and their different settings of application. In fact, for deep learning alone, [189] has found up to 50 papers, each developing their own variants each one with their specific subtleties.

However, and despite its excellent performance, an analysis of Adagrad in the nonconvex setting has appeared only relatively recently in [141] and [204]. [141] requires *a-priori* knowledge of the Lipschitz constant while [204] is parameter agnostic. Both assume boundedness of the gradients for stochastic problems and conclude that Adagrad's global convergence rate is comparable to that of well-tuned stochastic gradient methods, with a complexity of $\mathcal{O}\left(\epsilon^{-4}\right)$ to

achieve (1.2.3). Other proofs, still requiring a uniform bound on the sampled gradient, were subsequently developed using simpler arguments and better dependence for Adam [77], giving improved convergence rates under 'gradient sparsity' [222]. The assumption of bounded stochastic gradient has finally been removed, see [93, 94, 5] and the references therein. A complexity analysis of the Adagard norm for *deterministic* nonconvex optimization has been established in [204]. This analysis demonstrates a global convergence rate of $\mathcal{O}\left(\epsilon^{-2}\right)$, achieved without the need to assume bounded gradients. Building on this foundation, [194] introduced a new analysis for (component-wise) Adagrad in the convex scenario. Their analysis exhibits an explicit[1] dependence on the problem size, a feature not present in the study by [204]. Yet, as observed in algorithm 1.4.1, there's no utilization of curvature information. Furthermore, no theoretical guidelines for scaling alpha have been provided in the literature, especially for the nonconvex setting.

Question 3 *Can Adagrad be reformulated to allow the use of Hessian information? How to devise a theoretical scaling rule for the stepsize α in this case?*

To obtain faster OFFO algorithms than gradient-based ones, it is necessary to use other derivatives be it Hessian or higher-order tensors. In the next section, we review some conditions that have been proposed in the literature for the analysis of stochastic cubic and high-order methods.

1.5 OFFO High-Order Adaptive Regularization

Although the proposed framework of Definition 1.4.1 is no longer valid as it depends crucially on the trust-region paradigm, the conclusions drawn in Subsection 1.4.1 are still valid in this case. Indeed, even in deterministic inexact settings [214, 211], errors are only imposed on the derivatives and access to the exact function value is required. We will now give the conditions that have been proposed in a probabilistic adaptive cubic algorithm [11] as an illustration. In the above reference, the inaccurately sampled gradient and the Hessian satisfy

$$\begin{aligned} \|\overline{\nabla_x^1 f}(x_k) - \nabla_x^1 f(x_k)\| &\leq \kappa(1-\beta)^2 \left(\frac{\|\overline{\nabla_x^1 f}(x_k)\|}{\sigma_k}\right)^2, \\ \|\overline{\nabla_x^2 f}(x_k) - \nabla_x^2 f(x_k)\| &\leq c_k, \\ \|\overline{\nabla_x^1 f}(x_k)\| \leq \kappa_g, \quad \|\overline{\nabla_x^2 f}(x_k)\| &\leq \kappa_B. \end{aligned} \tag{1.5.1}$$

κ, κ_g, κ_B and β are fixed constants, while c_k is a constant that depends on $\overline{\nabla_x^1 f}(x_k)$.

Despite that approximation procedures to generate imprecise derivatives that satisfy (1.5.1) are provided in [211, 11] and that the theoretical analysis of [11] allows (1.5.1) to hold in probability, the algorithm proposed in [11] still uses exact function values to adapt its regularization parameter. The latter conclusion also hold for tensors methods.

[1] Here, 'explicit' implies disregarding the potential dependence of the problem's Lipschitz constant on its dimension.

Indeed, the probabilistic high-order minimization algorithm proposed in [15] allows only stochasticity on the derivatives and error bounds on $\overline{f}(x_k)$ and $\overline{f}(x_k+s_k)$ hold for all iterations. For the sake of brevity, we will not detail the imposed conditions but we refer the reader to the aforementioned references for more details.

In other developments, the assumption that the noise on the function values must be controlled at a level lower than that allowed for the derivatives is a common thread in various works. This requirement is emphasized in studies such as [46, 62, 33, 91, 20, 12, 13, 11]. More recently, a "fully" probabilistic cubic regularization has been proposed by [188]. The function oracles verify conditions similar to those of [130] for stochastic first-order gradient line search with a probabilistic "light-tail" error. Other fully stochastic variants [221, 195, 83, 203, 60] require knowledge of Lipschitz constants and thus do not fall within our thesis analysis framework.

Since the success of first-order methods in Machine Learning is due to the existence of OFFO algorithms, it is natural to develop higher-order OFFO methods to tackle large-scale noisy problems. Regarding second-order OFFO methods, [109] proposed an exact OFFO trust-region. However, as a vanilla trust-region method, the proposed variant suffers from a $\mathcal{O}\left(\epsilon^{-2}\right)$ convergence rate as a first-order method. Therefore, there is a significant gap between the best known convergence rate of second-order methods [169, 50, 48] and the already developed second-order OFFO method. Thus, if we are interested in using the Hessian to solve (P-ML) while retaining the improved rate of standard adaptive regularization methods, devising an OFFO variant of the latter seems a natural extension to obtain a robust high-order algorithm that is not sensitive to the value of the function.

Question 4 *Can an exact second-order OFFO algorithm be developed? Can the paradigm be extended to higher order methods?*

We now conclude our brief review of past optimization methods and summarize the main questions that naturally arise from this review, and that will be central to the present manuscript.

1.6 Main contributions of subsequent Chapters

We provide answers to the fours above question, using material that has been or is being published in various continuous optimization journals. Each chapter will summarize elements presented in the following series of papers [117, 114, 118, 116]. When needed, we implement modifications to the original material for the sake of consistency with the rest of the thesis as we describe now.

1.6.1 A New Analysis of Deterministic Adagrad

Regarding Question 3, we take a new look at the Adagrad algorithm and consider it in a weighted trust-region framework. The Algorithm 1.4.1 is then interpreted as a box trust-region scaled with certain weights. In this new framework, one can naturally use curvature information to estimate the step size α in (1.4.7). Within this new approach, we propose

two classes of methods; A first class containing "Adagrad"-like variants and a second class containing a "divergent" stepsize. For both classes, we also propose a further normalization of the weights to improve the dependencies on the Lipschitz constant compared to the first analysis of nonconvex Adagrad developed in [204]. These developments are detailed in Chapter 2 of the thesis and are based on [114], to which we added convincing large-scale experiments.

Novelty The main contribution of this chapter lies mainly in the proof technique. Whereas vanilla convergence rate analysis relies on a local decrease property. In this case, we directly bound $\sum_{j=0}^{k} \|g_j\|^2$, by proving that

$$\sum_{j=0}^{k} \|g_j\|^2 \leq \kappa \tag{1.6.1}$$

where κ is a problem dependent constant. The last inequality also implies that

$$\min_{j\in\{0,\ldots,k\}} \|g_j\|^2 = \mathcal{O}\left(\frac{1}{(k+1)}\right). \tag{1.6.2}$$

which yields the $\mathcal{O}\left(\epsilon^{-2}\right)$ rate of convergence to a point satisfying (1.2.3).

1.6.2 On High-order Objective Free Methods

In Chapter 3 of the thesis, we provide an answer to the Question 4 by developing an exact adaptive regularization algorithm that does not utilize function values. For $p = 1$ and $\beta = 1$, our proposed algorithm recovers a previous adaptive gradient method (WNGrad) developed in [207]. When exact function values, are used standard ARC methods [51, 29] outperform our OFFO variants as it adapts more easily to the local smoothness of the function. However, even for small noise level on function evaluation, the performance of standard ARC deteriorates dramatically whereas our OFFO variant is more robust. The content is based on the journal paper [116] with extension to Hölder smooth tensors functions whereas the original dealt only with Lipschitz ones. We also develop a more sophisticated analysis for a particular algorithmic variant.

Novelty For the update rule of σ_k, we don't use the ratio (1.3.21) and, for $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n;\mathbb{R})$, we update the regularization parameter as

$$\sigma_{k+1} = \sigma_k + \sigma_k \|s_k\|^{p+\beta},$$

where the step verifies the local decrease condition (1.3.19) and a first-order condition related to (1.3.20). At first glance, it may seem that σ_k would be divergent. Below is an illustration of the convergence mechanism of the algorithm.

- Before reaching the first iteration k_1 where $\sigma_{k_1} \leq 2L_p$, no decrease is guaranteed and the objective function may increase. We just prove that the required number of iterations to reach $2L_p$ is of the required order. Namely, for $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n;\mathbb{R})$, it is $\mathcal{O}\left(\epsilon^{-(p+\beta)/(p+\beta-1)}\right)$ with ϵ the tolerance on the gradient norm (1.2.3).

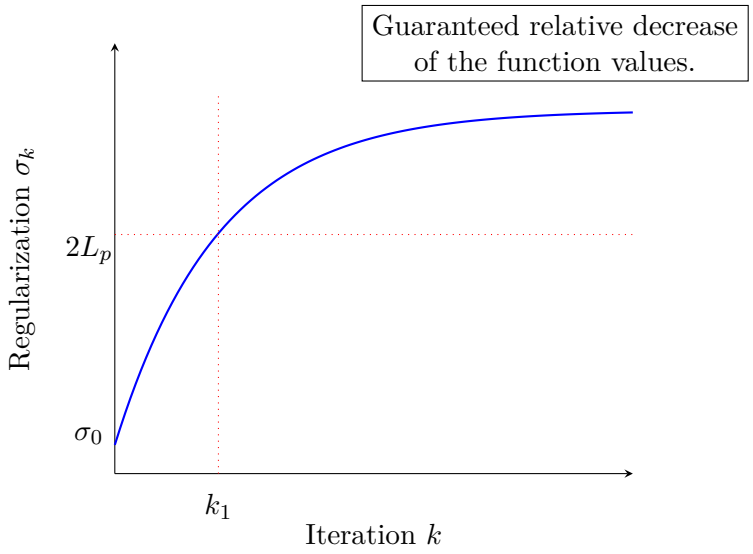

Figure 1.3: Sketch of the evolution of the regularization parameter σ_k

- At iteration k_1, some specific constants need to be bounded. The proof depends crucially on a bound on the step $\|s_k\|$. We therefore proof a bound on the step in adaptive regularization methods. We believe that this bound might be useful in other settings.
- For the numerical experiments, we propose to use past-information in order to better estimate the local Lipschitz smoothness and incorporate it within our new update rule of σ_k.

1.6.3 Fast Newton Method

Recently, both [82] and [156], independently proposed to change the implicit term $\frac{\sigma_k}{2}\|s_k\|$ by the explicit $\sqrt{\sigma_k\|\nabla_x^1 f(x_k)\|}$, resulting in a formulation that is both efficient and effective. This substitution simplifies the computation and leads to a more straightforward approach for tackling the optimization problem, which can be expressed as

$$(\nabla_x^2 f(x_k) + \sqrt{\sigma_k\|\nabla_x^1 f(x_k)\|}I_n)s = -\nabla_x^1 f(x_k). \tag{1.6.3}$$

The two papers, [82] and [156], also proved that the update rule of (1.6.3) lead to known optimal complexity rates for convex optimization methods. This new globalization trick spurred work on super-universal schemes [84], algorithms in which the Hessian is updated only for specific iterates [85], and adaptations for a class of self-concordant functions [79]. Unfortunately, all these works focus on the convex case.

In order to answer Question 1, we develop an adaptive algorithm that employs a step of type (1.6.3) in a convex region. In the proposed algorithm, a term accounting for negative curvature is added to $\sqrt{\sigma_k\|\nabla_x^1 f(x_k)\|}$. To avoid computing precise Hessian information, we also propose subspace implementations and trial step where only gradient regularization is considered. This new method enjoys a complexity rate that differs only by a logarithmic term from the standard ARC [50, 51]. At variance with previously mentioned optimal second-order methods [27, 26, 184, 74], our proposed algorithm does not involve complex inner iterations,

and initial numerical results show the merits of our approach. The details and developments are based on elements present [117] and are found in Chapter 4.

Novelty Our proposed algorithm combines a Newton step with a negative curvature direction in a sophisticated way. The complexity analysis proof is also more involved than of standard second-order methods [29, 50]. Indeed, some steps do not provide the required decrease. We circumvent this difficulty by using a well-chosen division of the sequence of successful steps.

1.6.4 Adaptive Regularization in Banach Spaces

Regarding Question 2 and the development of an adaptive high-order methods in infinite dimensional Banach space, the main difficulty lies in the fact that a minimizer of (1.3.18) is no longer guaranteed to exist. To tackle this problem, we develop a Hölder gradient descent method that can reach an approximate first-order point of a specific class of minimization problems This class naturally includes functions of the form (1.3.18). The Hölder gradient descent is endowed with a backtracking line search mechanism and can be easily implemented in practice. Afterwards, the theory subsequently follows the lines developed in [29] with only slight modifications to (1.3.20). This line of work is presented in the published paper [118].

Novelty The contributions in this chapter lie in the development of a backtracking line search for a particular class of functions. The gradient of a function ϕ in this class verifies a two terms bound which write as

$$\|\nabla_x^1\phi(x) - \nabla_x^1\phi(y)\|_{\mathcal{V}'} \leq L_f\|x-y\|^{\beta_1} + L_s\|x-y\|^{\beta_2} \tag{1.6.4}$$

for $x, y \in \mathcal{V}$ and $\beta_1, \beta_2 \in (0, 1]$. We also provide some characterization on the derivative of a power of $\|.\|_{\mathcal{V}}$ for a wide choice of Banach spaces. Further arguments regarding the necessity of studying nonlinear optimization algorithms in Banach spaces are given in Chapter 5.

Chapter 2

Complexity of First-Order OFFO Algorithms

Chapter Abstract

A parametric class of trust-region algorithms for unconstrained nonconvex optimization is considered where the value of the objective function is never computed. The class contains a deterministic version of the first-order Adagrad method typically used for minimization of noisy function, but also allows the use of (possibly approximate) second-order information when available. The rate of convergence of methods in the class is analyzed and is shown to be identical to that known for first-order optimization methods using both function and gradients values, recovering existing results for purely-first order variants and improving the explicit dependence on problem dimension. This rate is shown to be essentially sharp. A new class of methods is also presented, for which a slightly worse and essentially sharp complexity result holds. Limited numerical experiments show that the new methods' performance may be comparable to that of standard steepest descent, despite using significantly less information, and that this performance is relatively insensitive to noise.

Reference: The theory in the chapter and some experiments are taken from [114]. Additional large-scale numeric on image recognition are also provided.

2.1 Introduction

In the following, we consider the deterministic Adagrad algorithm [88] using a trust-region point [68] of view. We remind the reader that a review of Adagrad and the class of Adaptive gradient methods was provided at Subsection 1.4.3. Although this approach has not been used much in the machine learning context, trust region techniques have been investigated in the framework of noisy optimization problems. The method proposed in [92] uses function values but updates the radius bounding the steplength as a function of the gradient norm, which is also the case of [73]. The algorithm of [33] and [18] also uses the (noisy) objective function's value while also allowing noise in the derivatives. In contrast, the algorithms described in [109] and [91] do not evaluate the objective function. They differ from the proposal we are about to describe both in the technique of proof and the fact that they do not subsume Adagrad or many of its variants. Moreover, the analysis of [91] requires the explicit knowledge of the problem's Lipschitz constant in the algorithm for obtaining the best complexity estimate.

The purpose of the present chapter is to bridge the gap between standard first-order OFFO methods such as Adagrad and OFFO trust-region algorithms by considering a unified framework. More specifically,

1. we re-interpret the deterministic Adagrad as a particular member of a fairly general parametric class of trust-region methods (Sections 2.2 and 2.3). This class not only contains purely first-order algorithms such as Adagrad, but also allows the use of (possibly approximate) second-order information, should it be available using a Barzilai-Borwein approach [8], a limited-memory BFGS technique [143] or even exact second derivatives.

2. We then provide, for our proposed class, an essentially sharp global[1] bound on the gradient's norm as a function of the iteration counter, which is identical to that known for first-order optimization methods using both function and gradients values. This complexity result does not assume bounded gradients and extends that of [204] only valid for Adagrad-Norm to the complete class. It also uses one of the available parameters of the class to mitigate the explicit dependence of that bound on the problem's dimension.

3. We next exploit the proposed OFFO trust-region framework of Section 2.2 to propose (in Section 2.4) a new class class of such methods, for which an essentially sharp complexity result is also provided.

4. We finally illustrate our proposals by discussing some numerical experiments in Section 2.5, suggesting that the considered OFFO methods may indeed be competitive with steepest descent in efficiency and reliability while being much less sensitive to noise.

Additional notation. $w_{i,k}$ denotes the ith component of a vector $w_k \in \mathbb{R}^n$.

2.2 A class of first-order minimization methods

We consider the problem (P) where f is a smooth function from $\mathbb{R}^n$ to $\mathbb{R}$. In particular, we will assume in what follows that

AS.1: the objective function $f(x)$ is continuously differentiable and its gradient is Lipschitz continuous with Lipschitz constant L_1. i.e: $f \in \mathcal{C}^{1,1}(\mathbb{R}^n;\mathbb{R})$.

AS.2: there exists a constant f_{low} such that, for all x, $f(x) \geq f_{\text{low}}$.

AS.1 and AS.2 are standard for the complexity analysis of optimization methods seeking first-order critical points. We stress once more that we do not assume that the gradient are uniformly bounded, at variance with [207, 77, 222, 114].

The class of methods of interest here are iterative and generate a sequence of iterates $\{x_k\}_{k\geq 0}$. The move from an iterate to the next directly depends on the gradient at x_k and algorithm-dependent *scaling factors* $\{w_k = w(x_0, \ldots, x_k)\}$ whose main purpose is to control the move's magnitude. In our analysis, we will assume that

AS.3: for each $i \in \{1, \ldots, n\}$ there exists a constant $\varsigma_i \in (0, 1]$ such that, $w_{i,k} \geq \varsigma_i$ for all $k \geq 0$.

[1] I.e., valid at every iteration.

Since scaling factors are designed to control the length of the step, they are strongly reminiscent of the standard mechanism of the much studied trust-region optimization methods (see [68] for an extensive coverage and [208] for a more recent survey). In trust-region algorithms, a model of the objective function at an iterate x_k is built, typically using a truncated Taylor series, and a step s_k is chosen that minimizes this model with a *trust-region*, that is a region where the model is assumed to represent the true objective function sufficiently well. This region is a ball around the current iterate, whose radius is updated adaptively from iteration to iteration, based on the quality of the prediction of the objective function value at the trial point $x_k + s_k$. For methods using gradient only, the model is then chosen as the first two terms of the Taylor's expansion of f at the iterate x_k. Although, we are interested here in methods where the objective function's value is not evaluated, and therefore cannot be used to accept/reject iterates and update the trust-region radius, a similar mechanism may be designed, this time involving the weights $\{w_k\}$, the choice of which will detailed in the following two sections for two algorithmic subclasses of interest. The resulting algorithm, which we call ASTR1 (for Adaptively Scaled Trust Region using 1rst order information) is stated on the following page.

The algorithm description calls for some comments.

1. Observe that we allow the use of second-order information by effectively defining a quadratic model
$$g_k^T s + \tfrac{1}{2} s^T B_k s \tag{2.2.9}$$
where B_k can of course be chosen as the true second-derivative matrix of f at x_k (provided it remains bounded to satisfy (2.2.2)) or any approximation thereof. Choosing $B_k = 0$ results in a purely first-order algorithm.

   The condition (2.2.2) on the Hessian approximations is quite weak, and allows in particular for a variety of quasi-Newton approaches, limited-memory or otherwise. In a finite-sum context, sampling bounded Hessians is also possible.

2. Conditions (2.2.4)–(2.2.7) define a "generalized Cauchy point" (GCP), much in the spirit of standard trust-region methodology (see [68, Section 6.3] for instance), where the quadratic model (2.2.9) is minimized in (2.2.7) along a good first-order direction (s_k^L) to obtain a "Cauchy step" s_k^Q. Any step s_k can then be accepted provided it remains in the trust region (see (2.2.3)) and enforces a decrease in the quadratic model which is a least a fraction τ of that achieved at the Cauchy step (see (2.2.4)).

3. At variance with many existing trust-region algorithms, the radius Δ_k of the trust-region (2.2.1) is not recurred adaptively from iteration to iteration depending on how well the quadratic model predicts function values, but is directly defined as a scaled version of the local gradient. This is not without similarities with the trust-region methods proposed by [92], which corresponds to a scaling factor equal to $\|g_k\|^{-1}$, or [91] where the trust-region radius depends on $\|g_k\|$.

4. As stated, the ASTR1 algorithm does not include a termination rule, but such a rule can easily be introduced by terminating the algorithm in Step 1 if $\|g_k\| \leq \epsilon$, where $\epsilon > 0$ is a user-defined first-order accuracy threshold.

Algorithm 2.2.1: ASTR1

Step 0: Initialization. A starting point x_0 is given. Constants $\kappa_{\rm B} \geq 1$ and $\tau \in (0, 1]$ are also given. Set $k = 0$.

Step 1: Define the TR. Compute $g_k = g(x_k)$ and define

$$\Delta_{i,k} = \frac{|g_{i,k}|}{w_{i,k}} \tag{2.2.1}$$

where $w_k = w(x_0, \ldots, x_k)$.

Step 2: Hessian approximation. Select a symmetric Hessian approximation B_k such that

$$\|B_k\| \leq \kappa_{\rm B}. \tag{2.2.2}$$

Step 3: GCP. Compute a step s_k such that

$$|s_{i,k}| \leq \Delta_{i,k} \quad (i \in \{1, \ldots, n\}), \tag{2.2.3}$$

and

$$g_k^{\mathsf{T}} s_k + \tfrac{1}{2} s_k^{\mathsf{T}} B_k s_k \leq \tau \left(g_k^{\mathsf{T}} s_k^Q + \tfrac{1}{2} (s_k^Q)^{\mathsf{T}} B_k s_k^Q \right), \tag{2.2.4}$$

where

$$s_{i,k}^L = -\text{sign}(g_{i,k}) \Delta_{i,k}, \tag{2.2.5}$$

$$s_k^Q = \alpha_k s_k^L, \tag{2.2.6}$$

with

$$\alpha_k = \begin{cases} \min \left[1, \dfrac{|g_k^{\mathsf{T}} s_k^L|}{(s_k^L)^{\mathsf{T}} B_k s_k^L} \right] & \text{if} \quad (s_k^L)^{\mathsf{T}} B_k s_k^L > 0, \\ 1 & \text{otherwise.} \end{cases} \tag{2.2.7}$$

Step 4: New iterate. Define

$$x_{k+1} = x_k + s_k, \tag{2.2.8}$$

increment k by one and return to Step 1.

5. It may seem to the reader that we have introduced two algorithmic parameters typically not present in existing OFFO methods. As it turns out, this is standard practice for trust-region methods and it is widely acknowledged that the behaviour of the algorithm is relatively insensitive to the choice made. Typically values are

$$\tau = \tfrac{1}{10} \quad \text{and} \quad \kappa_{\rm B} = 10^5,$$

the last one being possibly adapted to reflect the problem scaling. Note that these values are constant throughout the execution of the algorithm. At variance, α_k is the iteration dependent stepsize, a quantity present in every first-order minimization method. Observe that we do not impose restrictions of the stepsize (beyond being positive), thereby covering most standard choices. Note that $\alpha_k = 1$ and $s_k^Q = s_k^L$ whenever $B_k = 0$.

The algorithm being defined, the first step of our analysis is to derive a fundamental property of objective-function decrease, valid for all choices of the scaling factors satisfying AS.3.

Lemma 2.2.1 *Suppose that AS.1, AS.2 and AS.3 hold. Then we have that, for all* $k \geq 0$,

$$f(x_{j+1}) \leq f(x_j) - \sum_{i=1}^{n} \frac{\tau \varsigma_{\min} g_{i,j}^2}{2\kappa_B w_{i,j}} + \tfrac{1}{2}(\kappa_B + L_1) \sum_{i=1}^{n} \frac{g_{i,j}^2}{w_{i,j}^2} \tag{2.2.10}$$

and

$$f(x_0) - f(x_{k+1}) \geq \sum_{j=0}^{k} \sum_{i=1}^{n} \frac{g_{i,j}^2}{2\kappa_B w_{i,j}} \left[\tau \varsigma_{\min} - \frac{\kappa_{BBL}}{w_{i,j}} \right] \tag{2.2.11}$$

where $\varsigma_{\min} \stackrel{def}{=} \min_{i \in \{1,\ldots,n\}} \varsigma_i$ *and* $\kappa_{BBL} \stackrel{def}{=} \kappa_B(\kappa_B + L)$.

Proof. Using (2.2.5) and AS.3, we deduce that, for every $j \geq 0$,

$$|g_j^T s_j^L| = \sum_{i=1}^{n} \frac{g_{i,j}^2}{w_{i,j}} = \sum_{i=1}^{n} \frac{w_{i,j} g_{i,j}^2}{w_{i,j}^2} \geq \sum_{i=1}^{n} \frac{\varsigma_i g_{i,j}^2}{w_{i,j}^2} \geq \varsigma_{\min} \|s_j^L\|^2. \tag{2.2.12}$$

Suppose first that $(s_j^L)^T B_j s_j^L > 0$ and $\alpha_j < 1$. Then, in view of (2.2.6), (2.2.7), (2.2.12) and (2.2.2),

$$g_j^T s_j^Q + \tfrac{1}{2}(s_j^Q)^T B_j s_j^Q = \alpha_j g_j^T s_j^L + \tfrac{1}{2}\alpha_j^2 (s_j^L)^T B_j s_j^L = -\frac{(g_j^T s_j^L)^2}{2(s_j^L)^T B_j s_j^L} \leq -\frac{\varsigma_{\min} |g_j^T s_j^L|}{2\kappa_{\rm B}}.$$

Combining this inequality with the first equality in (2.2.12) then gives that

$$g_j^T s_j^Q + \tfrac{1}{2}(s_j^Q)^T B_j s_j^Q \leq -\frac{\varsigma_{\min}}{2\kappa_{\rm B}} \sum_{i=1}^{n} \frac{g_{i,j}^2}{w_{i,j}}. \tag{2.2.13}$$

Suppose now that $(s_j^L)^\intercal B_j s_j^L \leq 0$ or $\alpha_j = 1$. Then, using (2.2.6), (2.2.13) and (2.2.5),

$$g_j^\intercal s_j^Q + \tfrac{1}{2}(s_j^Q)^\intercal B_j s_j^Q = g_j^\intercal s_j^L + \tfrac{1}{2}(s_j^L)^\intercal B_j s_j^L \leq \tfrac{1}{2} g_j^\intercal s_j^L < 0$$

and (2.2.13) then again follows from the bound $\kappa_{\rm B} \geq 1$. Successively using AS.1 and gradient Lipschitz bound, (2.2.4), (2.2.13), (2.2.2) and (2.2.1) then gives that, for $j \geq 0$,

$$\begin{aligned} f(x_{j+1}) &\leq f(x_j) + g_j^\intercal s_j + \tfrac{1}{2} s_j^\intercal B_j s_j - \tfrac{1}{2} s_j^\intercal B_j s_j + \tfrac{1}{2} L \|s_j\|^2 \\ &\leq f(x_j) + \tau \left(g_j^\intercal s_j^Q + \tfrac{1}{2}(s_j^Q)^\intercal B_j s_j^Q \right) + \tfrac{1}{2}(\kappa_{\rm B} + L)\|s_j\|^2 \\ &\leq f(x_j) - \sum_{i=1}^n \frac{\tau \varsigma_{\min} g_{i,j}^2}{2\kappa_{\rm B} w_{i,j}} + \tfrac{1}{2}(\kappa_{\rm B} + L) \sum_{i=1}^n \Delta_{i,j}^2 \\ &\leq f(x_j) - \sum_{i=1}^n \frac{\tau \varsigma_{\min} g_{i,j}^2}{2\kappa_{\rm B} w_{i,j}} + \tfrac{1}{2}(\kappa_{\rm B} + L) \sum_{i=1}^n \frac{g_{i,j}^2}{w_{i,j}^2}. \end{aligned}$$

This is (2.2.10). Summing up this inequality for $j \in \{0, \ldots, k\}$ then yields (2.2.11). □

Armed with Lemma 2.2.1, we are now in position to specify particular choices of the scaling factors $w_{i,k}$ and derive the convergence properties of the resulting variants of ASTR1.

2.3 An Adagrad-like variant of ASTR1 using second-order models

We first consider a choice of scaling factors directly derived from the definition of the Adagrad algorithm [88]. For given $\varsigma \in (0, 1]$, $\vartheta \in (0, 1]$, $\theta > 0$ and $\mu \in (0, 1)$, define, for all $i \in \{1, \ldots, n\}$ and for all $k \geq 0$,

$$w_{i,k} \in \left[\sqrt{\vartheta}\, v_{i,k}, v_{i,k} \right] \quad \text{where} \quad v_{i,k} \stackrel{\rm def}{=} \theta \left(\varsigma + \sum_{\ell=0}^k g_{i,\ell}^2 \right)^\mu. \tag{2.3.1}$$

The Adagrad scaling factors are recovered by $\mu = \frac{1}{2}$, $\theta = 1$ and $\vartheta = 1$, and ASTR1 with (2.3.1) and $B_k = 0$ is then the standard (deterministic) Adagrad method. The formulation (2.3.1) allows a parametric analysis of methods "in the neighbourhood" of Adagrad, using not only first-order but also second-order information. The ϑ parameter is introduced for flexibility, in particular allowing non-monotone scaling factors[2] The additional scaling parameter[3] θ is introduced as a technique to improve the convergence rate of the resulting algorithm and so devising an approach can give the appropriate scale to the stepsize α_k and hence give an answer to the second part of Question 3. Such a scaling has been considered and may be useful when the gradient is sparse [89] or when designing a private Adagrad version [4] in order to improve the complexity bound with respect to the problem's parameters. The above parametrization with $\theta = 1$ has also been considered in [58] in the context of a continuous Ordinary Differential Equations analysis and, with θ depending on problem's constants, in [141] where the sum on ℓ in (2.3.1) is terminated at $\ell = k-1$ and μ is restricted to the interval

[2] Typical values are $\varsigma_i = \frac{1}{100}$ and $\vartheta = \frac{1}{1000}$.

[3] Which can be viewed as a stepsize/learning rate parameter when $B_k = 0$.

$[\frac{1}{2}, 1)$ in the stochastic regime. In what follows, we consider the discrete deterministic case for the interval $(0, 1)$. For modern deep Learning problems, practitioners often use an exponential moving average in order to estimate $v_{i,k}$ defined at (2.3.1) as to put more emphasis on recent information, this however lead to an algorithm that is divergent even for the simple convex case as shown by [182]. Therefore, we restricted ourselves to the aforementioned case (2.3.1)

Before stating the global rate of convergence of the variant of ASTR1 using (2.3.1), we first prove a lemma, partly inspired by [204, 77].

Lemma 2.3.1 *Let $\{a_k\}_{k\geq 0}$ be a non-negative sequence, $\mu > 0$, $\xi > 0$ and define, for each $k \geq 0$, $b_k = \sum_{j=0}^{k} a_j$. Then if $\mu \neq 1$,*

$$\sum_{j=0}^{k} \frac{a_j}{(\xi + b_j)^{\mu}} \leq \frac{1}{(1-\mu)}((\xi + b_k)^{1-\mu} - \xi^{1-\mu}). \tag{2.3.2}$$

Otherwise (i.e. if $\mu = 1$),

$$\sum_{j=0}^{k} \frac{a_j}{(\xi + b_j)} \leq \log\left(\frac{\xi + b_k}{\xi}\right). \tag{2.3.3}$$

Proof. Consider first the case where $\mu \neq 1$ and note that $\frac{1}{(1-\mu)}x^{1-\mu}$ is then a non-decreasing and concave function on $(0, +\infty)$. Setting $b_{-1} = 0$ and using these properties, we obtain that, for $j \geq 0$,

$$\begin{aligned}\frac{a_j}{(\xi + b_j)^{\mu}} &\leq \frac{1}{1-\mu}\left((\xi + b_j)^{1-\mu} - (\xi + b_j - a_j)^{1-\mu}\right) \\ &= \frac{1}{1-\mu}\left((\xi + b_j)^{1-\mu} - (\xi + b_{j-1})^{1-\mu}\right).\end{aligned}$$

We then obtain (2.3.2) by summing this inequality for $j \in \{0, \ldots, k\}$.

Suppose now that $\mu = 1$, we then use the concavity and non-decreasing character of the logarithm to derive that

$$\frac{a_j}{(\xi + b_j)^{\mu}} = \frac{a_j}{(\xi + b_j)} \leq \log(\xi + b_j) - \log(\xi + b_j - a_j) = \log(\xi + b_j) - \log(\xi + b_{j-1}).$$

The inequality (2.3.3) then again follows by summing for $j \in \{0, \ldots, k\}$. □

From (2.3.2), we also obtain that, for $\mu < 1$,

$$\sum_{j=0}^{k} \frac{a_j}{(\xi + b_j)^{\mu}} \leq \frac{1}{(1-\mu)}(\xi + b_k)^{1-\mu} \tag{2.3.4}$$

while, for $\mu > 1$,

$$\sum_{j=0}^{k} \frac{a_j}{(\xi + b_j)^\mu} \leq \frac{\xi^{1-\mu}}{(\mu - 1)}. \tag{2.3.5}$$

Note that both the numerator and the denominator of the right-hand side of (2.3.2) tend to zero when μ tends to one. Applying l'Hospital rule, we then see that this right-hand side tends to the right-hand side of (2.3.3) and the bounds on $\sum_{j=0}^{k} a_j/(\xi + b_j)^\mu$ are therefore continuous at $\mu = 1$.

Lemma 2.3.1 is crucial in the proof of our main complexity result, which we now state. The proof hinges crucially on proving that $\sum_{j=0}^{k} \|g_j\|^2$ is bounded.

Theorem 2.3.2 *Suppose that AS.1 and AS.2 hold and that the* ASTR1 *algorithm is applied to problem (P) with its scaling given by (2.3.1). If we define*

$$\Gamma_0 \stackrel{def}{=} f(x_0) - f_{\rm low},$$

then,

(i) if $0 < \mu < \frac{1}{2}$,

$$\underset{j \in \{0,\ldots,k\}}{\text{average}} \|g_j\|^2 \leq \frac{\kappa_1}{k+1}, \tag{2.3.6}$$

with

$$\kappa_1 = \max\left\{\varsigma, \left[\frac{2^{2\mu}\vartheta(1-2\mu)\theta^2\Gamma_0}{n(\kappa_B + L)}\right]^{\frac{1}{1-2\mu}}, \left[\frac{4\,n\,\kappa_{BBL}}{(1-2\mu)\tau\theta\varsigma^\mu\vartheta^{\frac{3}{2}}}\right]^{\frac{1}{\mu}}\right\}; \tag{2.3.7}$$

(ii) if $\mu = \frac{1}{2}$,

$$\underset{j \in \{0,\ldots,k\}}{\text{average}} \|g_j\|^2 \leq \frac{\kappa_2}{k+1}, \tag{2.3.8}$$

with

$$\kappa_2 = \max\left\{\varsigma, \frac{1}{2}e^{\frac{2\Gamma_0\vartheta\theta^2}{n(\kappa_B+L)}}, \frac{1}{2\varsigma}\left(\frac{8n\kappa_B(\kappa_B+L)}{\tau\vartheta^{\frac{3}{2}}\theta}\right)^2 \left|W_{-1}\left(-\frac{\tau\varsigma\theta\vartheta^{\frac{3}{2}}}{8n\kappa_B(\kappa_B+L)}\right)\right|^2\right\}, \tag{2.3.9}$$

where W_{-1} is the second branch of the Lambert function [70];

(iii) if $\frac{1}{2} < \mu < 1$,

$$\underset{j \in \{0,\ldots,k\}}{\text{average}} \|g_j\|^2 \leq \frac{\kappa_3}{k+1} \tag{2.3.10}$$

with

$$\kappa_3 \stackrel{def}{=} \max\left\{\varsigma, \left[\frac{2^{1+\mu}\kappa_B}{\tau\varsigma^\mu\sqrt{\vartheta}}\left(\Gamma_0\theta + \frac{n(\kappa_B+L)\varsigma^{1-2\mu}}{2\vartheta\theta(2\mu-1)}\right)\right]^{\frac{1}{1-\mu}}\right\} \tag{2.3.11}$$

Proof. We see from (2.3.1) that $w_{i,k}$ verifies AS.3. We may thus use Lemma 2.2.1. Moreover, (2.3.1) also implies that

$$\varsigma^{\mu}\sqrt{\vartheta}\theta \leq w_{i,j} \leq \theta\left(\varsigma + \sum_{\ell=0}^{j} \|g_\ell\|^2\right)^{\mu} \tag{2.3.12}$$

for all $j \geq 0$ and all $i \in \{1,\ldots,n\}$. We now deduce from (2.2.1) and (2.2.11) that, for $k \geq 0$,

$$f(x_{k+1}) \;\leq\; f(x_0) - \sum_{j=0}^{k} \frac{\tau\varsigma^{\mu}\sqrt{\vartheta}\,\|g_j\|^2}{2\kappa_{\mathrm{B}} \max\limits_{i\in\{1,\ldots,n\}} w_{i,k}} + \tfrac{1}{2}(\kappa_{\mathrm{B}}+L)\sum_{i=1}^{n}\sum_{j=0}^{k}\Delta_{i,j}^2. \tag{2.3.13}$$

For each $i \in \{1,\ldots,n\}$, we then apply Lemma 2.3.1 with $a_\ell = g_{i,\ell}^2$, $\xi = \varsigma$ and $2\mu < 1$, and obtain from (2.2.1) and (2.3.1) that,

$$\sum_{j=0}^{k}\Delta_{i,j}^2 \leq \frac{1}{\theta^2\vartheta(1-2\mu)}\left[\left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2\right)^{1-2\mu} - \varsigma^{1-2\mu}\right] \leq \frac{1}{\theta^2\vartheta(1-2\mu)}\left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2\right)^{1-2\mu}. \tag{2.3.14}$$

Now

$$\sum_{i=1}^{n}\sum_{j=0}^{k}\Delta_{i,j}^2 \leq \sum_{i=1}^{n}\frac{1}{\theta^2\vartheta(1-2\mu)}\left(\varsigma + \sum_{i=1}^{n}\sum_{\ell=0}^{k} g_{i,\ell}^2\right)^{1-2\mu} \leq \frac{n}{\theta^2\vartheta(1-2\mu)}\left(\varsigma + \sum_{\ell=0}^{k}\|g_\ell\|^2\right)^{1-2\mu} \tag{2.3.15}$$

and thus substituting this bound in (2.3.13) and using AS.2 gives that

$$\sum_{j=0}^{k}\frac{\tau\varsigma^{\mu}\sqrt{\vartheta}\|g_j\|^2}{2\kappa_{\mathrm{B}}\max\limits_{i\in\{1,\ldots,n\}} w_{i,k}} \leq \Gamma_0 + \frac{n(\kappa_{\mathrm{B}}+L)}{2\theta^2\vartheta}(1-2\mu)\left(\varsigma + \sum_{j=0}^{k}\|g_j\|^2\right)^{1-2\mu}. \tag{2.3.16}$$

Suppose now that

$$\sum_{j=0}^{k}\|g_j\|^2 \geq \max\left\{\varsigma, \left[\frac{2^{2\mu}\theta^2\vartheta(1-2\mu)\Gamma_0}{n(\kappa_{\mathrm{B}}+L)}\right]^{\frac{1}{1-2\mu}}\right\}, \tag{2.3.17}$$

implying

$$\varsigma + \sum_{j=0}^{k}\|g_j\|^2 \leq 2\sum_{j=0}^{k}\|g_j\|^2 \;\text{ and }\; \Gamma_0 \leq \frac{n(\kappa_{\mathrm{B}}+L)}{2\theta^2\vartheta(1-2\mu)}\left(2\sum_{j=0}^{k}\|g_j\|^2\right)^{1-2\mu}.$$

Then, using (2.3.16) and (2.3.12),

$$\frac{\tau\varsigma^{\mu}\sqrt{\vartheta}}{2^{1+\mu}\,\kappa_{\mathrm{B}}\theta\left[\sum_{\ell=0}^{k}\|g_\ell\|^2\right]^{\mu}}\sum_{j=0}^{k}\|g_j\|^2 \leq \frac{2^{1-2\mu}\,n\,(\kappa_{\mathrm{B}}+L)}{\vartheta\theta^2(1-2\mu)}\left(\sum_{j=0}^{k}\|g_j\|^2\right)^{1-2\mu}.$$

Solving this inequality for $\sum_{j=0}^{k} \|g_j\|^2$ gives that

$$\sum_{j=0}^{k} \|g_j\|^2 \leq \left[\frac{4\, n\ \kappa_{\mathrm{BBL}}}{(1-2\mu)\theta\tau\varsigma^{\mu}\vartheta^{\frac{3}{2}}} \right]^{\frac{1}{\mu}}$$

and therefore

$$\underset{j\in\{0,\ldots,k\}}{\text{average}} \|g_j\|^2 \leq \left[\frac{4\, n\, \kappa_{\mathrm{BBL}}}{(1-2\mu)\theta\tau\varsigma^{\mu}\vartheta^{\frac{3}{2}}} \right]^{\frac{1}{\mu}} \cdot \frac{1}{k+1}. \tag{2.3.18}$$

Alternatively, if (2.3.17) fails, then

$$\underset{j\in\{0,\ldots,k\}}{\text{average}} \|g_j\|^2 < \max\left\{ \varsigma, \left[\frac{2^{2\mu}\vartheta(1-2\mu)\theta^2\Gamma_0}{2^{1-2\mu}n(\kappa_{\mathrm{B}}+L)} \right]^{\frac{1}{1-2\mu}} \right\} \cdot \frac{1}{k+1}. \tag{2.3.19}$$

Combining (2.3.18) and (2.3.19) gives (2.3.6).

Let us now consider the case where $\mu = \frac{1}{2}$. For each $i \in \{1, \ldots, n\}$, we apply Lemma 2.3.1 with $a_k = g_{i,k}^2$, $\xi = \varsigma$ and $2\mu = 1$ and obtain that,

$$\sum_{i=1}^{n}\sum_{j=0}^{k} \Delta_{i,j}^2 \leq \frac{1}{\vartheta\theta^2} \sum_{i=1}^{n} \log\left(\frac{1}{\varsigma}\left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2 \right) \right) \leq \frac{n}{\vartheta\theta^2} \log\left(1 + \frac{1}{\varsigma}\sum_{\ell=0}^{k} \|g_\ell\|^2 \right).$$

and substituting this bound in (2.3.13) then gives that

$$\sum_{j=0}^{k} \frac{\tau\sqrt{\varsigma\vartheta}\|g_j\|^2}{2\kappa_{\mathrm{B}} \max\limits_{i\in\{1,\ldots,n\}} w_{i,k}} \leq \Gamma_0 + \frac{n(\kappa_{\mathrm{B}}+L)}{2\vartheta\theta^2} \log\left(1 + \frac{1}{\varsigma}\sum_{j=0}^{k} \|g_j\|^2 \right).$$

Suppose now that

$$\sum_{j=0}^{k} \|g_j\|^2 \geq \max\left[\varsigma, \frac{1}{2} e^{\frac{2\vartheta\theta^2\Gamma_0}{n(\kappa_B+L)}} \right], \tag{2.3.20}$$

implying that

$$1 + \frac{1}{\varsigma}\sum_{j=0}^{k} \|g_j\|^2 \leq \frac{2}{\varsigma}\sum_{j=0}^{k} \|g_j\|^2 \ \text{ and } \ \Gamma_0 \leq \frac{n(\kappa_{\mathrm{B}}+L)}{2\vartheta\theta^2} \log\left(\frac{2}{\varsigma}\sum_{j=0}^{k} \|g_j\|^2 \right).$$

Using (2.3.12) for $\mu = \frac{1}{2}$, we obtain then that

$$\frac{\tau\sqrt{\varsigma\vartheta}}{2\sqrt{2}\,\theta\,\kappa_{\mathrm{B}}\sqrt{\sum_{\ell=0}^{k} \|g_\ell\|^2}} \sum_{j=0}^{k} \|g_j\|^2 \leq \frac{n(\kappa_{\mathrm{B}}+L)}{\vartheta\theta^2} \log\left(\frac{2}{\varsigma}\sum_{j=0}^{k} \|g_j\|^2 \right),$$

that is

$$\frac{\tau\sqrt{2\varsigma}\vartheta^{\frac{3}{2}}\theta}{4\kappa_{\mathrm{B}}} \sqrt{\sum_{j=0}^{k} \|g_j\|^2} \leq 2n(\kappa_{\mathrm{B}}+L) \log\left(\sqrt{\frac{2}{\varsigma}\sum_{j=0}^{k} \|g_j\|^2} \right). \tag{2.3.21}$$

Now define

$$\gamma_1 \stackrel{\text{def}}{=} \frac{\tau\varsigma\vartheta^{\frac{3}{2}}\theta}{4\kappa_{\mathrm{B}}}, \quad \gamma_2 \stackrel{\text{def}}{=} 2n(\kappa_{\mathrm{B}}+L) \quad \text{and} \quad u \stackrel{\text{def}}{=} \sqrt{\frac{2}{\varsigma}\sum_{j=0}^{k}\|g_j\|^2} \tag{2.3.22}$$

and observe that that $\gamma_2 > 3\gamma_1$ because $\tau\sqrt{\varsigma}\vartheta^{\frac{3}{2}} \leq 1$ and $\kappa_{\mathrm{B}} \geq 1$. The inequality (2.3.21) can then be rewritten as

$$\gamma_1 u \leq \gamma_2 \log(u). \tag{2.3.23}$$

Let us denote by $\psi(u) \stackrel{\text{def}}{=} \gamma_1 u - \gamma_2 \log(u)$. Since $\gamma_2 > 3\gamma_1$, the equation $\psi(u) = 0$ admits two roots $u_1 \leq u_2$ and (2.3.23) holds for $u \in [u_1, u_2]$. The definition of u_2 then gives that

$$\log(u_2) - \frac{\gamma_1}{\gamma_2}u_2 = 0$$

which is

$$u_2 e^{-\frac{\gamma_1}{\gamma_2}u_2} = 1.$$

Setting $z = -\frac{\gamma_1}{\gamma_2}u_2$, we obtain that

$$ze^z = -\frac{\gamma_1}{\gamma_2}$$

Thus $z = W_{-1}(-\frac{\gamma_1}{\gamma_2}) < 0$, where W_{-1} is the second branch of the Lambert function defined over $[-\frac{1}{e}, 0)$. As $-\frac{\gamma_1}{\gamma_2} \geq -\frac{1}{3}$, z is well defined and thus

$$u_2 = -\frac{\gamma_2}{\gamma_1}\, z = -\frac{\gamma_2}{\gamma_1}\, W_{-1}\left(-\frac{\gamma_1}{\gamma_2}\right) > 0.$$

As a consequence, we deduce from (2.3.23) and (2.3.22) that

$$\sum_{j=0}^{k}\|g_j\|^2 = \frac{\varsigma}{2}\,u_2^2 = \frac{1}{2\varsigma}\,\left(\frac{8n\kappa_{\mathrm{B}}(\kappa_{\mathrm{B}}+L)}{\tau\vartheta^{\frac{3}{2}}\theta}\right)^2 \left|W_{-1}\left(-\frac{\tau\varsigma\vartheta^{\frac{3}{2}}\theta}{8n\kappa_{\mathrm{B}}(\kappa_{\mathrm{B}}+L)}\right)\right|^2.$$

and

$$\operatorname*{average}_{j\in\{0,\ldots,k\}}\|g_j\|^2 \leq \frac{1}{2\varsigma}\,\left(\frac{8n\kappa_{\mathrm{B}}(\kappa_{\mathrm{B}}+L)}{\tau\vartheta^{\frac{3}{2}}\theta}\right)^2 \left|W_{-1}\left(-\frac{\tau\varsigma\vartheta^{\frac{3}{2}}\theta}{8n\kappa_{\mathrm{B}}(\kappa_{\mathrm{B}}+L)}\right)\right|^2 \cdot \frac{1}{k+1}. \tag{2.3.24}$$

If (2.3.20) does not hold, we have that

$$\operatorname*{average}_{j\in\{0,\ldots,k\}}\|g_j\|^2 < \max\left\{\varsigma, \frac{1}{2}\,e^{\frac{2\Gamma_0\vartheta\theta^2}{n(\kappa_B+L)}}\right\} \cdot \frac{1}{k+1}. \tag{2.3.25}$$

Combining (2.3.24) and (2.3.25) gives (2.3.8).

Finally, suppose that $\frac{1}{2} < \mu < 1$. Once more, we apply Lemma 2.3.1 for each $i \in \{1, \ldots, n\}$

with $a_\ell = g_{i,\ell}^2$, $\xi = \varsigma$ and $2\mu > 1$ and obtain that

$$\sum_{j=1}^{k} \Delta_{k,j}^2 \leq \frac{1}{\theta^2 \vartheta(1-2\mu)} \left(\left(\varsigma + \sum_{\ell=0}^{k} g_{i,\ell}^2 \right)^{1-2\mu} - \varsigma^{1-2\mu} \right) \leq \frac{\varsigma^{1-2\mu}}{\vartheta \theta^2 (2\mu - 1)}. \tag{2.3.26}$$

Substituting the bound (2.3.26) in (2.3.13) and using (2.3.12) and AS.2 gives that

$$\sum_{j=0}^{k} \frac{1}{(\varsigma + \sum_{j=0}^{k} \|g_j\|^2)^\mu} \frac{\tau \varsigma^\mu \sqrt{\vartheta} \|g_j\|^2}{2\kappa_{\mathrm{B}} \theta} \leq \Gamma_0 + \frac{n(\kappa_{\mathrm{B}} + L)\varsigma^{1-2\mu}}{2\theta^2 \vartheta (2\mu - 1)}.$$

If we now suppose that

$$\sum_{j=0}^{k} \|g_j\|^2 \geq \varsigma, \tag{2.3.27}$$

then

$$\underset{j \in \{0,\ldots,k\}}{\text{average}} \|g_j\|^2 \leq \left[\frac{2^{1+\mu} \kappa_{\mathrm{B}}}{\tau \varsigma^\mu \sqrt{\vartheta}} \left(\Gamma_0 \theta + \frac{n(\kappa_{\mathrm{B}} + L)\varsigma^{1-2\mu}}{2\vartheta(2\mu - 1)\theta} \right) \right]^{\frac{1}{1-\mu}} \cdot \frac{1}{k+1}. \tag{2.3.28}$$

If (2.3.27) does not hold, we derive that

$$\underset{j \in \{0,\ldots,k\}}{\text{average}} \|g_j\|^2 \leq \frac{\varsigma}{(k+1)}. \tag{2.3.29}$$

Thus, (2.3.28) and(2.3.29) finally imply (2.3.10). □

These result suggest additional remarks.

1. That the bounds given are not continuous as a function μ at $\mu = \frac{1}{2}$ is a result of our bounding process within the proof of Theorem 2.3.2 (for instance in the last inequality of (2.3.14)). Continuous bounds have been proved (see [115]) if one is ready to assume that the objective functions' gradients remain uniformly bounded.

2. If the algorithm is terminated as soon as $\|g_k\| \leq \epsilon$ (which is customary for deterministic algorithms searching for first-order points), it must stop at the latest at iteration

$$k = \kappa_\star^2 \epsilon^{-2}, \tag{2.3.30}$$

where $\kappa_\star = \kappa_1$ for $\mu \in (0, \frac{1}{2})$, $\kappa_\star = \kappa_2$ for $\mu = \frac{1}{2}$ and $\kappa_\star = \kappa_3$ for $\mu \in (\frac{1}{2}, 1)$. It is truly remarkable that there exist first-order OFFO methods whose global complexity order is identical to that of standard first-order methods using function evaluations (see [163, 112, 49] or [57, Chapter 2]), despite the fact that the latter exploit significantly more information. As mentionned in the introduction, this complexity rate was also derived for the analysis of Adagrad-Norm in [204].

3. It is possible to give a weaker but more explicit bound on κ_2 by finding an upper bound on the value of the involved Lambert function. This can be obtained by using [59,

Theorem 1] which states that, for $x > 0$,

$$\left|W_{-1}(-e^{-x-1})\right| \leq 1 + \sqrt{2x} + x.$$

Remembering that, for γ_1 and γ_2 given by (2.3.22), $\log\left(\frac{\gamma_2}{\gamma_1}\right) \geq \log(3) > 1$ and taking $x = \log\left(\frac{\gamma_2}{\gamma_1}\right) - 1 > 0$ in (2.3.31) then gives that

$$\left|W_{-1}\left(-\frac{\gamma_1}{\gamma_2}\right)\right| \leq \log\left(\frac{\gamma_2}{\gamma_1}\right) + \sqrt{2\left(\log\left(\frac{\gamma_2}{\gamma_1}\right) - 1\right)}. \tag{2.3.31}$$

4. It is also possible to extend the definition of s_k^L in (2.2.5) by premultiplying it by a stepsize $\alpha_k \in [\alpha_{\min}, 1]$ for some $\alpha_{\min} \in (0, 1]$. Our results again remain valid (with modified constants). Covering a deterministic momentum-less Adam would require extending the results to allow for (2.3.1) to be replaced by

$$w_{i,k} = \varsigma + \sum_{j=0}^{k} \beta_2^{k-j} g_{i,j}^2 \quad (i \in \{1, \ldots, n\}) \tag{2.3.32}$$

for some $\beta_2 < 1$. This can be done by following the argument of Theorem 2 in [77]. However, as in this reference, the final bound on the squared gradient norms does not tend to zero when k grows[4], illustrating the (known) lack of convergence of Adam. We therefore do not investigate this option in detail.

5. Focusing on the choice of the parameter μ independently of ϑ and θ, we verify that the choice $\mu = \frac{1}{2}$ is best, yielding an upper complexity bound for the deterministic Adagrad algorithm and recovering a similar result obtained in [204] for the Adagrad-Norm algorithm.

   The choice $\mu = 1$ is not covered by our theory but has been considered in [160], where a specific choice of the constant θ with $\mu = 1$ is used to derive a first order method with optimal regret for strongly convex problems. Unfortunately, the proposed SC(Strong Convex)-Adagrad algorithm requires the knowledge of the problem's Lipschitz constant and noise characteristics.

We now discuss the dependence of the bounds given by Theorem 2.3.2 as a function of the problem's constants L, n and Γ_0 and recall that if n is known *a priori*, this is generally not the case for the Lispchitz constant L, while the gap Γ_0 may be available for some classes of problems (such as nonlinear regressions where $\Gamma_0 \leq f(x_0)$). We are mostly interested in the explicit dependence of the bounds on n, taking into account that the unknown L often varies little with dimension –such as in problems arising from discretizations– but admittedly ignoring the fact it may also depend on n, sometimes severely [57, p. 14]. As we are allowed to do so, we would like to choose the scaling parameter θ in order to offset this dependence as much as possible. In view of (2.3.7), (2.3.9) and (2.3.11), and noting that (2.3.31) indicates that the $|W_{-1}|^2$ term in (2.3.9) can be summarized as $\mathcal{O}\left(\log(nL)^2\right)$, we attempt to balance the impact of the various terms involving both θ and n and, in the absence of additional

[4] A constant term in $-\log(\beta_2)$ refuses to vanish.

information on the problem, select

$$\theta_\star = \begin{cases} \sqrt{n/\Gamma_0} & \text{when } \Gamma_0 \text{ is known,} \\ \sqrt{n} & \text{otherwise.} \end{cases} \tag{2.3.33}$$

We may then compare the bounds obtained by reinjecting this value in (2.3.9) ($\mu = \frac{1}{2}$) with the results for deterministic Adagrad-Norm [204] and Adagrad [194]. This comparison is summarized in the following table.

Regime	Algorithm	Constant dependence
Non-Convex [204]	$w_{i,k} = \sqrt{\varsigma + \sum_{j=0}^k \|g_j\|^2}$	$\mathcal{O}\left(L^2 \log(L)^2\right)$
Convex [194]	(2.3.1) with ($\theta = 1$, $\vartheta = 1$)	$\mathcal{O}\left(n^2 L^2 \log(L)^2\right)$
Non-convex (2.3.9)	(2.3.1) with ($\theta = \sqrt{n}$, $\vartheta = 1$)	$\mathcal{O}\left(nL^2 \log(\sqrt{n}L)^2\right)$

Are the (good) upper bound given Theorem 2.3.2 sharp?

Theorem 2.3.3 *The bounds (2.3.6), (2.3.8) and (2.3.10) are essentially sharp in that, for each $\mu \in (0,1)$ and each $\eta \in (0,1]$, there exists a univariate function $f_{\mu,\eta}$ satisfying AS.1 and AS.2 such that, when applied to minimize $f_{\mu,\eta}$ from the origin, the ASTR1 algorithm with (2.3.1), $B_k = 0$ and $\vartheta = \theta = 1$ produces a sequence of gradient norms given by $\|g_k\| = \frac{1}{k^{\frac{1}{2}+\eta}}$.*

Proof. Following ideas of [57, Theorem 2.2.3], we first construct a sequence of iterates $\{x_k\}$ for which $f_{\mu,\eta}(x_k) = f_k$ and $\nabla_x^1 f_{\mu,\eta}(x_k) = g_k$ for associated sequences of function and gradient values $\{f_k\}$ and $\{g_k\}$, and then apply Hermite interpolation to exhibit the function $f_{\mu,\eta}$ itself. We start by defining

$$g_0 \stackrel{\text{def}}{=} -2, \quad g_k \stackrel{\text{def}}{=} -\frac{1}{k^{\frac{1}{2}+\eta}} \quad (k > 0), \tag{2.3.34}$$

$$s_0 \stackrel{\text{def}}{=} \frac{2}{(\varsigma+4)^\mu}, \quad s_k \stackrel{\text{def}}{=} \frac{1}{k^{\frac{1}{2}+\eta}(\varsigma + \sum_{j=0}^k g_j^2)^\mu} \quad (k > 0) \tag{2.3.35}$$

yielding that

$$|g_0 s_0| = \frac{4}{(\varsigma+4)^\mu}, \quad |g_k s_k| = \frac{1}{k^{1+2\eta}(\varsigma + \sum_{j=0}^k g_j^2)^\mu} \leq \frac{1}{k^{1+2\eta}} \quad (k > 0) \tag{2.3.36}$$

(remember that $g_0^2 = 4$). We then define $B_k \stackrel{\text{def}}{=} 0$ for all $k \geq 0$,

$$x_0 = 0, \quad x_{k+1} = x_k + s_k \quad (k > 0) \tag{2.3.37}$$

and

$$f_0 = \frac{4}{(\varsigma+4)^\mu} + \zeta(1+2\eta) \quad \text{and} \quad f_{k+1} = f_k + g_k s_k \quad (k \geq 0), \tag{2.3.38}$$

where $\zeta(\cdot)$ is the Riemann zeta function. Observe that the sequence $\{f_k\}$ is decreasing and that, for all $k \geq 0$,

$$f_{k+1} = f_0 - \sum_{k=0}^{k} |g_k s_k| \geq f_0 - \frac{4}{(\varsigma+4)^\mu} - \sum_{k=1}^{k} \frac{1}{k^{1+2\eta}} \geq f_0 - \frac{4}{(\varsigma+4)^\mu} - \zeta(1+2\eta)$$

where we used (2.3.38) and (2.3.36). Hence (2.3.38) implies that

$$f_k \in [0, f_0] \ \text{ for all } \ k \geq 0. \tag{2.3.39}$$

Also note that, using (2.3.38),

$$|f_{k+1} - f_k - g_k s_k| = 0, \tag{2.3.40}$$

while, using (2.3.35),

$$|g_0 - g_1| = 1 \leq \frac{1}{2}(\varsigma+4)^\mu \, s_0.$$

Moreover, using the fact that $1/x^{\frac{1}{2}+\eta}$ is a convex function of x over $[1, +\infty)$, and that from (2.3.35) $s_k \geq \frac{1}{k^{\frac{1}{2}+\eta}(\varsigma+4+k)^\mu}$, we derive that, for $k > 0$,

$$\begin{aligned}
|g_{k+1} - g_k| &= \left| \frac{1}{(k+1)^{\frac{1}{2}+\eta}} - \frac{1}{k^{\frac{1}{2}+\eta}} \right| \\
&\leq \left(\frac{1}{2} + \eta \right) \frac{1}{k^{\frac{3}{2}+\eta}} \\
&\leq \frac{3}{2} \, \frac{(\varsigma+4+k)^\mu}{k k^{\frac{1}{2}+\eta}(\varsigma+4+k)^\mu} \\
&\leq \frac{3}{2} \, \frac{(\varsigma+4+k)^\mu}{k} s_k \\
&\leq \frac{3}{2} \, (\varsigma+5)^\mu \, s_k.
\end{aligned}$$

These last bounds and (2.3.39) allow us to use standard Hermite interpolation on the data given by $\{f_k\}$ and $\{g_k\}$: see, for instance, Theorem A.9.1 in [57] with $p = 1$ and

$$\kappa_f = \max\left[\frac{3}{2}(\varsigma+5)^\mu, f_0, 2 \right] \tag{2.3.41}$$

(the second term in the max bounding $|f_k|$ because of (2.3.39) and the third bounding $|g_k|$ because of (2.3.34)). We then deduce that there exists a continuously differentiable function $f_{\mu,\eta}$ from $\mathbb{R}$ to $\mathbb{R}$ with Lipschitz continuous gradient (i.e. satisfying AS.1 and AS.2) such that, for $k \geq 0$,

$$f_{\mu,\eta}(x_k) = f_k \quad \text{and} \quad \nabla_x^1 f_{\mu,\eta}(x_k) = g_k.$$

Moreover, the range of $f_{\mu,\eta}$ and $\nabla_x^1 f_{\mu,\eta}$ are constant independent of η, hence guaranteeing AS.1 and AS.2. The definitions (2.3.34), (2.3.35), (2.3.37) and (2.3.38) imply that the sequences $\{x_k\}$, $\{f_k\}$ and $\{g_k\}$ can be seen as generated by the ASTR1 algorithm (with (2.3.1), $B_k = 0$ and $\vartheta = \theta = 1$) applied to $f_{\mu,\eta}$, starting from $x_0 = 0$ and the desired conclusion follows. □

The bounds (2.3.6), (2.3.8) and (2.3.10) are therefore *essentially sharp* (in the sense of [52]) for the ASTR1 algorithm with (2.3.1) and $\vartheta = \theta = 1$, which is to say that the lower complexity bound for the algorithm is arbitrarily close to its upper bound. Interestingly, the argument in the proof of the above theorem fails for $\eta = 0$, as this choice yields that

$$\sum_{j=0}^{k} g_j^T s_j \geq \sum_{j=0}^{k} \frac{1}{k(\varsigma + \log(k+1))^\mu}.$$

Since

$$\int_1^k \frac{dt}{t(\log(t+1))^\mu} > \int_1^k \frac{dt}{(t+1)(\log(t+1))^\mu} = \frac{(\log(k+1))^{1-\mu}}{1-\mu} - \frac{\log(2)^{1-\mu}}{1-\mu}$$

tends to infinity as k grows, this indicates (in view (2.3.38)) that AS.2 cannot hold. Also note that (2.3.34) implies that the gradients remain uniformly bounded.

2.4 A further "diminishing stepsizes" variation on this theme

We now use a different proof technique to design new variants of ASTR1 with a fast global k-order. This is achieved by modifying the definition of the scaling factors $w_{i,k}$, requiring them to satisfy a fairly general growth condition explicitly depending on k, the iteration index. More specifically, we will assume, in this section, that the scaling factors $w_{i,k}$ are chosen such that, for some power parameter $0 < \nu \leq \mu < 1$, all $i \in \{1, \ldots, n\}$ and some constants $\varsigma_i \in (0, 1]$ and $\theta > 0$,

$$\theta \max[\varsigma_i, v_{i,k}] \, (k+1)^\nu \leq w_{i,k} \leq \theta \max[\varsigma_i, v_{i,k}] \, (k+1)^\mu \quad (k \geq 0), \tag{2.4.1}$$

where, for each i, the $v_{i,k}$ satisfy the properties that

$$v_{i,k+1} > v_{i,k} \quad \text{implies that} \quad v_{i,k+1} \leq |g_{i,k+1}| \tag{2.4.2}$$

and

$$v_{i,k} \geq |g_{i,k}|/h(k) \tag{2.4.3}$$

for some positive function $h(k)$ only depending on k. Of particular choices where

$$v_{i,k} = \max_{j \in \{0,\ldots,k\}} |g_{i,j}| \quad \text{and} \quad v_{i,k} = \frac{1}{k+1} \sum_{j \in \{0,\ldots,k\}} |g_{i,j}|$$

which both satisfy (2.4.2) and (2.4.3) (with $h(k) = 1$ for the first and $h(k) = k + 1$ for the second). We further illustrate this in Section 2.5.

We start by proving a useful technical result.

Lemma 2.4.1 *Consider and arbitrary $i \in \{1, \ldots, n\}$ and suppose that there exists a j_ς such that*

$$\min\left[\frac{g_{i,j}^2}{\varsigma_i}, \frac{g_{i,j}^2}{v_{i,j}}\right] \leq \varsigma_i \quad \textit{for} \quad j \geq j_\varsigma. \tag{2.4.4}$$

Then

$$\min\left[\frac{g_{i,j}^2}{\varsigma_i}, \frac{g_{i,j}^2}{v_{i,j}}\right] \geq \frac{g_{i,j}^2}{2\varsigma_i} \quad \textit{for} \quad j \geq j_\varsigma. \tag{2.4.5}$$

Proof. Suppose that there exists a $j > j_\varsigma$ such that $v_{i,j} > 2\varsigma_i$. Assume, without loss of generality that j is the smallest such index. Then $v_{i,j} > v_{i,j-1}$ and (2.4.2) implies that $|g_{i,j}| \geq v_{i,j} \geq 2\varsigma_i$. As a consequence,

$$\min\left[\frac{g_{i,j}^2}{\varsigma_i}, \frac{g_{i,j}^2}{v_{i,j}}\right] \geq \min[4\varsigma_i, 2\varsigma_i] > \varsigma_i,$$

which contradicts (2.4.4). Thus no such j can exists and $v_{i,j} \leq 2\varsigma_i$ for all $j > j_*$ and (2.4.5) follows. □

We are now in position to state our complexity result for the ASTR1 algorithm using weight defined by (2.4.1), (2.4.2) and (2.4.3).

Theorem 2.4.2 *Suppose that AS.1 and AS.2 hold and that the ASTR1 algorithm is applied to problem (P), where the scaling factors $w_{i,k}$ are chosen in accordance with (2.4.1), (2.4.2) and (2.4.3). Then, for any $\eta \in (0, \tau\varsigma_{\min})$ and*

$$j_\eta \stackrel{def}{=} \left(\frac{\kappa_B(\kappa_B + L)}{\theta\varsigma_{\min}(\tau\varsigma_{\min} - \eta)}\right)^{\frac{1}{\nu}}, \tag{2.4.6}$$

there exist a constant $\kappa_\diamond$, a subsequence $\{k_\ell\} \subseteq \{k\}_{j_\eta+1}^{\infty}$ and an index k_ς (where $\kappa_\diamond$ and k_ς only depend on the problem and the algorithmic constants) such that, for all $k_\ell \geq k_\varsigma$,

$$\min_{j\in\{0,\ldots,k_\ell\}} \|g_j\|^2 \leq \kappa_\diamond \frac{(k_\ell+1)^\mu}{k_\ell - j_\eta} \leq \frac{2\kappa_\diamond(j_\eta+1)}{k_\ell^{1-\mu}}. \tag{2.4.7}$$

Proof. From (2.2.11) and AS.2, using $w_{\min,j} \stackrel{\text{def}}{=} \min_{i\in\{1,\ldots,n\}} w_{i,k}$ ensures that

$$\Gamma_0 \geq f(x_0) - f(x_{k+1}) \geq \sum_{j=0}^{k}\sum_{i=1}^{n} \frac{g_{i,j}^2}{2\kappa_{\mathrm{B}} w_{i,j}}\left[\tau\varsigma_{\min} - \frac{\kappa_{\mathrm{BBL}}}{w_{\min,j}}\right]. \tag{2.4.8}$$

Consider now an arbitrary $\eta \in (0, \tau\varsigma_{\min})$ and suppose first that, for some j,

$$\left[\tau\varsigma_{\min} - \frac{\kappa_{\rm BBL}}{w_{\min,j}}\right] \leq \eta, \tag{2.4.9}$$

i.e., using (2.4.1),

$$\theta\varsigma_{\min}\ j^{\nu} \leq w_{\min,j} \leq \frac{\kappa_{\rm BBL}}{\tau\varsigma_{\min} - \eta}.$$

But this is impossible for $j > j_\eta$ for j_η given by (2.4.6), and hence (2.4.9) fails for all $j > j_\eta$. As a consequence, we have that, for $k > j_\eta$,

$$\begin{aligned}
f(x_{j_\eta+1}) - f(x_k) &\geq \eta \sum_{j=j_\eta+1}^{k} \sum_{i=1}^{n} \frac{g_{i,j}^2}{2\kappa_{\rm B} w_{i,j}} \\
&\geq \frac{\eta}{2\kappa_{\rm B}} \sum_{j=j_\eta+1}^{k} \sum_{i=1}^{n} \frac{g_{i,j}^2}{\max[\varsigma_i, v_{i,j}]\,\theta\,(j+1)^{\mu}} \\
&\geq \frac{\eta}{2\kappa_{\rm B}(k+1)^{\mu}\theta} \sum_{j=j_\eta+1}^{k} \sum_{i=1}^{n} \min\left[\frac{g_{i,j}^2}{\varsigma_i}, \frac{g_{i,j}^2}{v_{i,j}}\right] \\
&\geq \frac{\eta(k-j_\eta)}{2\kappa_{\rm B}(k+1)^{\mu}\theta} \min_{j\in\{j_\eta+1,\ldots,k\}} \left(\sum_{i=1}^{n} \min\left[\frac{g_{i,j}^2}{\varsigma_i}, \frac{g_{i,j}^2}{v_{i,j}}\right]\right)
\end{aligned} \tag{2.4.10}$$

But we also know from (2.2.10), (2.4.1) and (2.4.3) that

$$\begin{aligned}
f(x_0) - f(x_{j_\eta+1}) &\geq \sum_{j=0}^{j_\eta} \sum_{i=1}^{n} \frac{\tau\varsigma_{\min} g_{i,j}^2}{2\kappa_{\rm B} w_{i,j}} - \tfrac{1}{2}(\kappa_{\rm B}+L) \sum_{j=0}^{j_\eta} \sum_{i=1}^{n} \frac{g_{i,j}^2}{w_{i,j}^2} \\
&\geq -\tfrac{1}{2}(\kappa_{\rm B}+L) \sum_{j=0}^{j_\eta} \sum_{i=1}^{n} \frac{g_{i,j}^2}{w_{i,j}^2} \\
&\geq -\tfrac{1}{2}(\kappa_{\rm B}+L) \sum_{j=0}^{j_\eta} \sum_{i=1}^{n} \frac{g_{i,j}^2}{\max[\varsigma, v_{i,k}]^2 (j+1)^{2\nu}\theta^2} \\
&\geq -\tfrac{1}{2}\frac{(\kappa_{\rm B}+L)}{\theta^2} \sum_{j=0}^{j_\eta} \sum_{i=1}^{n} \frac{g_{i,j}^2}{v_{i,k}^2 (j+1)^{2\nu}} \\
&\geq -\tfrac{1}{2}\frac{(\kappa_{\rm B}+L)n}{\theta^2} \sum_{j=0}^{j_\eta} h(j)^2.
\end{aligned} \tag{2.4.11}$$

Combining (2.4.10) and (2.4.11), we obtain that

$$\Gamma_0 \geq f(x_0) - f(x_{k+1}) \geq -\tfrac{1}{2}\frac{(\kappa_{\rm B}+L)n}{\theta^2} \sum_{j=0}^{j_\eta} h(j)^2 + \frac{\eta(k-j_\eta)}{2\kappa_{\rm B}(k+1)^{\mu}\theta} \min_{j\in\{j_\theta+1,\ldots,k\}} \left(\sum_{i=1}^{n} \min\left[\frac{g_{i,j}^2}{\varsigma_i}, \frac{g_{i,j}^2}{v_{i,j}}\right]\right)$$

and thus that

$$\min_{j\in\{j_\theta+1,\ldots,k\}}\left(\sum_{i=1}^{n}\min\left[\frac{g_{i,j}^2}{\varsigma_i},\frac{g_{i,j}^2}{v_{i,j}}\right]\right)\leq\frac{2\kappa_{\rm B}(k+1)^\mu}{\eta(k-j_\eta)}\left[\Gamma_0\theta+\tfrac{1}{2}\frac{n(\kappa_{\rm B}+L)}{\theta}\sum_{j=0}^{j_\eta}h(j)^2\right]$$

and we deduce that there must exist a subsequence $\{k_\ell\}\subseteq\{k\}_{j_\eta+1}^{\infty}$ such that, for each ℓ,

$$\sum_{i=1}^{n}\min\left[\frac{g_{i,k_\ell}^2}{\varsigma_i},\frac{g_{i,jk_\ell}^2}{v_{i,k_\ell}}\right]\leq\frac{2\kappa_{\rm B}(k_\ell+1)^\mu}{\eta(k_\ell-j_\eta)}\left[\Gamma_0\theta+\tfrac{1}{2}\frac{n(\kappa_{\rm B}+L)}{\theta}\sum_{j=0}^{j_\eta}h(j)^2\right]. \tag{2.4.12}$$

But

$$\frac{(k_\ell+1)^\mu}{k_\ell-j_\eta}<\frac{2^\mu k_\ell^\mu}{k_\ell-j_\eta}<\frac{2k_\ell^\mu}{k_\ell-j_\eta}=\frac{2k_\ell^\mu k_\ell}{(k_\ell-j_\eta)k_\ell}=\frac{k_\ell}{k_\ell-j_\eta}\cdot\frac{2}{k_\ell^{1-\mu}}\leq\frac{2(j_\eta+1)}{k_\ell^{1-\mu}}, \tag{2.4.13}$$

where we used the facts that $\mu<1$ and that $\frac{k_\ell}{k_\ell-j_\theta}$ is a decreasing function for $k_\ell\geq j_\theta+1$. Using this inequality, we thus obtain from (2.4.12) that, for each ℓ,

$$\sum_{i=1}^{n}\min\left[\frac{g_{i,k_\ell}^2}{\varsigma_i},\frac{g_{i,jk_\ell}^2}{v_{i,k_\ell}}\right]\leq\frac{4\kappa_{\rm B}(j_\eta+1)}{\eta\,k_\ell^{1-\mu}}\left[\Gamma_0\theta+\tfrac{1}{2}n\frac{(\kappa_{\rm B}+L)}{\theta}\sum_{j=0}^{j_\eta}h(j)^2\right].$$

As a consequence,

$$k_\varsigma\stackrel{\rm def}{=}\left(\frac{4\kappa_{\rm B}(j_\eta+1)\left[\Gamma_0\theta+\frac{1}{2}\frac{n(\kappa_{\rm B}+L)}{\theta}\sum_{j=0}^{j_\eta}h(j)^2\right]}{\eta\varsigma_{\min}}\right)^{\frac{1}{1-\mu}}$$

is such that, for all $k_\ell\geq k_\varsigma$,

$$\min\left[\frac{g_{i,k_\ell}^2}{\varsigma_i},\frac{g_{i,\ell}^2}{v_{i,k\ell}}\right]\leq\varsigma_{\min}.$$

Lemma 2.4.1 then yields that, for all $k_\ell\geq k_\varsigma$,

$$\sum_{i=1}^{n}\frac{g_{i,k_\ell}^2}{2\varsigma_i}\leq\frac{2\kappa_{\rm B}(k_\ell+1)^\mu}{\eta(k_\ell-j_\eta)}\left[\Gamma_0\theta+\tfrac{1}{2}\frac{n(\kappa_{\rm B}+L)}{\theta}\sum_{j=0}^{j_\eta}h(j)^2\right]$$

which, because $\varsigma_i\leq 1$, gives that, for all $k_\ell\geq k_\varsigma$,

$$\|g_{k_\ell}\|^2\leq\frac{(k_\ell+1)^\mu}{k_\ell-j_\eta}\left(\frac{4\kappa_{\rm B}}{\eta}\right)\left[\Gamma_0\theta+\tfrac{1}{2}n\frac{(\kappa_{\rm B}+L)}{\theta}\sum_{j=0}^{j_\eta}h(j)^2\right], \tag{2.4.14}$$

finally implying (2.4.7) because of (2.4.13). □

We again provide some comments on this last result.

1. The choice (2.4.1) is of course reminiscent, in a smooth and nonconvex setting, of the "diminishing stepsize" subgradient method for stochastic problems (see [23, Theorem 1.2.4]

or [9, Theorems 8.25 and 8.40] and the many references therein), for which a $\mathcal{O}(1/\sqrt{k})$ global rate of convergence is known.

2. Theorem 2.4.2 provides information on the speed of convergence for iterations that are beyond an *a priori* computable iteration index. Indeed j_θ and k_ς only depends on ν $h(\ell)$ and problem's constants and, in particular do not depend on k. However, the formulation of the theorem is slightly weaker than that of Theorem 2.3.2. Because (2.4.7) only holds for iterates along the subsequence $\{k_\ell\}$, there is no guarantee that the bounnd given by the right-hand-side is valid at other iterations. But note that this right-hand side depends on k_ℓ, which is an index in the complete sequence of iterates, rather than on ℓ (the subsequence index).

   This slightly weaker formulation is no longer necessary if one is ready to assume bounded gradients, as can be seen in Theorem 4.1 in [115].

3. As the chosen values of μ and ν approach zero, then the k-order of convergence beyond j_θ tends to $\mathcal{O}(1/\sqrt{k_\ell})$, which the order derived for the methods of the previous section and is the standard k-order for first-order methods using evaluations of the objective function, albeit the value of j_θ might increase.

4. As in Section 2.3 and considering (2.4.14), we may choose θ according to (2.3.33) in an attempt to balance the two terms of the left-hand side and improve the explicit dependence of the complexity bound on n.

We are now again interested to estimate how sharp the k-order bound (2.4.7) in $\mathcal{O}(\frac{1}{k^{(1-\mu)/2}})$ is.

Theorem 2.4.3 *The bound (2.4.7) is essentially sharp in that, for any $\omega > \frac{1}{2}(1-\nu)$, there exists a univariate function $f_\omega(x)$ satisfying AS.1 and AS.2 such that the* ASTR1 *algorithm with (2.4.1), $B_k = 0$ and $\theta = 1$ applied to this function produces a sequence of gradient norms given by $\|g_k\| = \frac{1}{(k+1)^\omega}$.*

Proof. Consider the sequence defined, for some $\omega \in (\frac{1}{2}(1-\nu), 1]$ and all $k \geq 0$, by

$$g_k = -\frac{1}{(k+1)^\omega} \quad w_k = \max\left[\varsigma, \max_{\ell \in \{0,\ldots,k\}} |g_\ell|\right] (k+1)^\nu = (k+1)^\nu, \tag{2.4.15}$$

$$s_k = \frac{1}{(k+1)^{2\omega-\nu}} < 1 \quad \text{and} \quad f_{k+1} = f_k + g_k s_k, \tag{2.4.16}$$

where we have chosen $\varsigma \in (0,1)$ and $f_0 = \zeta(2\omega + \frac{1}{2})$ where $\zeta(\cdot)$ is the Riemann zeta function. Immediately note that

$$\lim_{k\to\infty} |g_k| = 0,$$

and $|g_k| \leq= 1 = \kappa_g$ for all k. We now verify that, if

$$x_0 = 0 \quad \text{and} \quad x_k = x_{k-1} + s_{k-1} \ \text{ for } \ k \geq 1,$$

then exists a function $f_\omega(x)$ satisfying AS.1 and AS.2 such that, for all $k \geq 0$,

$$f_\omega(x_k) = f_k, \quad \text{and} \quad g_\omega(x_k) = g_k,$$

and such that the sequence defined by (2.4.15)-(2.4.16) is generated by applying the ASTR1 algorithm using $B_k = 0$ and $\theta = 1$. The function $f_\omega(x)$ is constructed using Hermite interpolation on each interval $[x_k, x_{k+1}]$ (note that the x_k are monotonically increasing), which known (see [49] or [57, Th. A.9.2]) to exist whenever there exists a constant $\kappa_f \geq 0$ such that, for each k,

$$|f_{k+1} - f_k - g_k s_k| \leq \kappa_f |s_k|^2 \quad \text{and} \quad |g_{k+1} - g_k| \leq \kappa_f |s_k|.$$

The first of these conditions holds by construction of the $\{f_k\}_{k\geq 0}$. To verify the second, we first note that, because $1/(k+1)^\omega$ is a convex function of k and $|1/(k+1)| \leq 1$,

$$\frac{|g_{k+1} - g_k|}{|s_k|} \leq \frac{\omega(k+1)^{2\omega-\nu}}{(k+1)^{1+\omega}} = \frac{\omega}{(k+1)^{\nu-\omega+1}} \leq \omega \quad (k \geq 0), \tag{2.4.17}$$

where $\nu - \omega + 1 \geq \nu > 0$, so that the desired inequality holds with $\kappa_f = \omega$.

Moreover, Hermite interpolation guarantees that $f_\omega(x)$ is bounded below whenever $|f_k|$ and $|s_k|$ remain bounded. We have already verified the second of these conditions in (2.4.16). We also have from (2.4.16) that

$$f_0 - f_{k+1} = \sum_{j=0}^{k} \frac{1}{(j+1)^{2\omega}(j+1)^\nu} \tag{2.4.18}$$

which converges to the finite limit $\zeta(2\omega + \nu)$ because we have chosen $\omega > \frac{1}{2}(1-\nu)$. Thus $f_k \in (0, \zeta(2\omega+\nu)]$ for all k and the first condition is also satisfied and AS.2 holds. This completes our proof. □

The conclusions which can be drawn from this theorem parallel those drawn after Theorem 2.3.3. The bound (2.4.7) is essentially sharp (in the sense of [52][5]) for the ASTR1 algorithm with (2.4.1).

2.5 Numerical illustration

We now provide some numerical illustration on problems which are commonly used for the evaluation of optimization algorithms. For the sake of clarity and conciseness, we needed to keep the list of algorithmic variants reported here reasonably limited, and have taken the following considerations into account for our choice.

1. Both weights' definitions (2.3.1) and (2.4.1) are illustrated. Moreover, since the Adam algorithm using (2.3.32) is so commonly used the stochastic context, we also included it in the comparison.

[5] Observe that f_0 now tends to infinity when ω tends to $\frac{1}{2}(\nu-1)$ and hence that AS.2 fails in the limit. As before, the structure of (2.4.7) implies that the complexity bound deteriorates when the gap $\Gamma_0 = f(x_0) - f_{\text{low}}$ grows.

2. Despite Theorems 2.3.2 and 2.4.2 covering a wide choice of the parameters μ and ν, we have chosen to focus here on the most common choice for (2.3.1) and (2.3.32) (i.e. $\mu = \frac{1}{2}$ and $\beta_2 = \frac{9}{10}$, corresponding to Adagrad, Adagrad-Norm and Adam). When using (2.4.1), we have also restricted our comparison to the single choice of μ and ν namely $\mu = \nu = \frac{1}{10}$. By contrast, we have included results for the variants of the weights' definitions for values of $\theta = 1$ (the standard choice) and $\theta = \sqrt{n}$ suggested in (2.3.33) for unknown Γ_0.

3. In order to be able to test enough algorithmic variants on enough problems in reasonable computing time, we have decided to focus our experiments on low-dimensional problems in the case where $\vartheta = 1$. We have nevertheless considered a few large-scale instance for the purpose of illustrating the effect of the scaling $\theta = \sqrt{n}$ which is only expected to be significant for such instances.

4. We have chosen to define the step s_k in Step 3 of the ASTR1 algorithm by approximately minimizing the quadratic model (2.2.9) within the ℓ_∞ trust-region using a projected truncated conjugate-gradient approach [158, 159] which is terminated as soon as

$$\|g_k + B_k s_k\|_2 \leq \max\left[10^{-12}, 10^{-5}\|g_k\|_2\right].$$

   We also considered an alternative, namely that of minimizing the quadratic model in an Euclidean ℓ_2 trust region (with the same accuracy requirement) using a Generalized Lanczos Trust Region (GLTR) technique [102].

5. We thought it would be interesting to compare "purely first-order" variants (that is variants for which $B_k = 0$ for all k) with methods using some kind of Hessian approximation. Among many possibilities, we selected three types of approximations of interest. The first is the diagonal Barzilai-Borwein approximation [8]

$$B_{k+1} = \frac{\|s_k\|_2^2}{y_k^T s_k} I_n \tag{2.5.1}$$

   where I_n is the identity matrix of dimension n, $y_k = g_{k+1} - g_k$ and $y_k^T s_k \geq 10^{-15}\|s_k\|_2^2$. The second is limited-memory BFGS approximations [143], where a small number (3) of BFGS updates are added to the matrix (2.5.1), each update corresponding to a secant pair (y_k, s_k) with $y_k^T s_k \geq 10^{-15}\|s_k\|_2^2$. The third is not to approximate the Hessian at all, but to use its exact value, that is $B_k = \nabla_x^2 f(x_k)$ for all k.

Given these considerations, we have selected the algorithmic ASTR1 variants using $\alpha_k = 1$ and detailed in Table 2.1, where the second column indicates the norm used to define the trust-region.

Note that all variants for which $B_{k+1} = 0$ are "purely first-order" in the sense discussed above. Note also that, under AS.3, `maxgnorm` and `maxg` satisfy (2.4.1) with $\mu = \nu = \frac{1}{10}$, $\varsigma_i = \varsigma = \frac{1}{100}$ and $\kappa_w = \kappa_g$. All algorithms were run[6] on the low dimensional instances of the problems[7] of the OPM collection [110] (April 2023), a subset of widely used CUTEst testing environment [106]. The instances are listed with their dimension in Table 1, until

[6] In Matlab® running under Ubuntu on a Dell Precision with 16 cores and 64 GB of memory.
[7] From their standard starting point.

Name	Norm	$w_{i,k}$ definition ($i \in \{1,\dots,n\}$)	B_{k+1}	params
`adagnorm`	$\|\cdot\|_2$	$w_{i,k} = \left[\frac{1}{100} + \sum_{j=0}^{k} \|g_j\|_2^2\right]^{\frac{1}{2}}$	0	
`adagrad`	$\|\cdot\|_\infty$	$w_{i,k} = \left[\frac{1}{100} + \sum_{j=0}^{k} g_{i,j}^2\right]^{\frac{1}{2}}$	0	
`adamnorm`	$\|\cdot\|_2$	$w_{i,k} = \left[\frac{1}{100} + \sum_{j=0}^{k} \beta_2^{k-j}\|g_j\|_2^2\right]^{\frac{1}{2}}$	0	$\beta = \frac{9}{10}$
`adam`	$\|\cdot\|_\infty$	$w_{i,k} = \left[\frac{1}{100} + \sum_{j=0}^{k} \beta_2^{k-j} g_{i,j}^2\right]^{\frac{1}{2}}$	0	$\beta = \frac{9}{10}$
`maxgnorm`	$\|\cdot\|_2$	$w_{i,k} = (k+1)^{\frac{1}{10}} \max\left[\frac{1}{100}, \max_{j\in\{0,\dots,k\}} \|g_j\|_2\right]$	0	
`maxg`	$\|\cdot\|_\infty$	$w_{i,k} = (k+1)^{\frac{1}{10}} \max\left[\frac{1}{100}, \max_{j\in\{0,\dots,k\}} \|g_{i,j}\|\right]$	0	
`adagbb`	$\|\cdot\|_\infty$	$w_{i,k} = \left[\frac{1}{100} + \sum_{j=0}^{k} g_{i,j}^2\right]^{\frac{1}{2}}$	(2.5.1)	
`adagbfgs3`	$\|\cdot\|_\infty$	$w_{i,k} = \left[\frac{1}{100} + \sum_{j=0}^{k} g_{i,j}^2\right]^{\frac{1}{2}}$	LBFGS(3)	
`adagH`	$\|\cdot\|_\infty$	$w_{i,k} = \left[\frac{1}{100} + \sum_{j=0}^{k} g_{i,j}^2\right]^{\frac{1}{2}}$	$\nabla_x^2 f(x_{k+1})$	
`adagrads`	$\|\cdot\|_\infty$	$w_{i,k} = \sqrt{n}\left[\frac{1}{100} + \sum_{j=0}^{k} g_{i,j}^2\right]^{\frac{1}{2}}$	0	
`adams`	$\|\cdot\|_\infty$	$w_{i,k} = \sqrt{n}\left[\frac{1}{100} + \sum_{j=0}^{k} \beta_2^{k-j} g_{i,j}^2\right]^{\frac{1}{2}}$	0	$\beta = \frac{9}{10}$
`maxgs`	$\|\cdot\|_\infty$	$w_{i,k} = \sqrt{n}(k+1)^{\frac{1}{10}} \max\left[\frac{1}{100}, \max_{j\in\{0,\dots,k\}} \|g_{i,j}\|\right]$	0	
`adagbbs`	$\|\cdot\|_\infty$	$w_{i,k} = \sqrt{n}\left[\frac{1}{100} + \sum_{j=0}^{k} g_{i,j}^2\right]^{\frac{1}{2}}$	(2.5.1)	
`adagbfgs3s`	$\|\cdot\|_\infty$	$w_{i,k} = \sqrt{n}\left[\frac{1}{100} + \sum_{j=0}^{k} g_{i,j}^2\right]^{\frac{1}{2}}$	LBFGS(3)	
`adagHs`	$\|\cdot\|_\infty$	$w_{i,k} = \sqrt{n}\left[\frac{1}{100} + \sum_{j=0}^{k} g_{i,j}^2\right]^{\frac{1}{2}}$	$\nabla_x^2 f(x_{k+1})$	
`sdba`	standard steepest-descent with backtracking (e.g. [57, Algorithm 2.2.1])			

Table 2.1: The considered algorithmic variants.

either $\|\nabla_x^1 f(x_k)\|_2 \leq 10^{-6}$, or a maximum of 100000 iterations was reached, or evaluation of the derivatives returned an error.

Before considering the results, we make two additional comments. The first is that very few of the test functions have bounded gradients on the whole of $\mathbb{R}^n$. While this is usually not a problem when testing standard first-order descent methods (because it may then be true in the level set determined by the starting point), this is no longer the case for significantly non-monotone methods like the ones tested here. As a consequence, it may (and does) happen that the gradient evaluation is attempted at a point where its value exceeds the Matlab overflow limit, causing the algorithm to fail on the problem. The second comment is that the (sometimes quite wild) non-monotonicity of the methods considered here has another practical consequence: it happens on several nonconvex problems[8] that convergence of different

[8] broyden3d, broydenbd, curly10, gottfr, hairy, indef, jensmp, osborneb, sensors, wmsqrtals, wmsqrtbls, woods.

Method	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$	algo	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$
adagbfgs3	0.75	69.75	adagrad	0.69	73.11
sdba	0.73	68.91	adagbfgs3	0.75	69.75
adagH	0.72	69.75	adagH	0.72	69.75
adagrad	0.69	73.11	sdba	0.73	68.91
maxg	0.66	66.39	adagHs	0.63	67.23
adagHs	0.63	67.23	maxg	0.66	66.39
adagbb	0.63	64.71	adagrads	0.60	65.55
adagbfgs3s	0.62	63.87	adagbb	0.63	64.72
maxgs	0.60	62.18	adagbfgs3s	0.62	63.87
adagrads	0.59	65.55	maxgs	0.60	62.18
adagnorm	0.58	61.34	adagnorm	0.58	61.34
maxgnorm	0.56	57.98	adagbbs	0.56	60.50
adagbbs	0.56	60.50	maxgnorm	0.56	57.98
adamnorm	0.55	34.45	adamnorm	0.55	34.45
adam	0.54	30.25	adams	0.52	33.61
adams	0.52	33.61	adam	0.54	30.25

Table 2.2: Performance and reliability statistics for deterministic OFFO and steepest descent algorithms on small OPM problems ($\epsilon_1 = 10^{-6}$).

algorithmic variants occurs to points with gradient norm within termination tolerance (the methods are thus achieving their objective), but these points can be quite far apart and may have very different function values.

We discuss the results of our tests from the efficiency and reliability points of view. Efficiency is measured in number of derivatives' evaluations (or, equivalently, iterations)[9]: the fewer evaluations the more efficient the algorithm. Because the standard performance profiles [86] for our selection of 16 algorithms would be too crowded to read, we follow [179] and consider the derived "global" measure $\pi_{\texttt{algo}}$ to be $\frac{1}{50}$ of the area below the curve corresponding to algo in this performance profile, for abscissas in the interval $[1, 50]$. The larger this area and closer $\pi_{\texttt{algo}}$ to one, the closer the curve to the right and top borders of the plot and the better the global performance. When reporting reliability, we say that the run of an algorithmic variant on a specific test problem is successful if the gradient norm tolerance has been achieved.

In what follows, $\rho_{\texttt{algo}}$ denotes the percentage of successful runs taken on all problems. Table 2.2 presents the values of these statistics in two columns: for easier reading, the variants are sorted by decreasing global performance ($\pi_{\texttt{algo}}$) in the first, and by decreasing reliability ($\rho_{\texttt{algo}}$) in the second. A total of 18 problems[10] could not be successfully solved by any of the above algorithms, we believe mostly because of ill-conditioning.

The authors are of course aware that the very limited experiments presented here do not replace extended numerical practice and could be completed in various ways. They nevertheless suggest the following (very tentative) comments.

[9] For sdba, gradient and objective-function evaluations.

[10] biggs5, brownbs, cliff, genhumps, gulf, heart6ls, heart8ls, himm29, mexhat, meyer3, nondquar, osbornea, penalty2, powellbs, powellsg, scurly10, watson, yfitu.

1. There often seems to be a definite advantage in using the $\|\cdot\|_\infty$ norm over $\|\cdot\|_2$, as can be seen by comparing `adagnorm` with `adagrad` and `maxgnorm` with `maxg`. While this may be due in part to the fact that the trust region in ℓ_∞ norm is larger than that in ℓ_2 norm (and thus allows larger steps), it is also the case that the disaggregate definition of the scaling factors $w_{i,k}$ ((2.3.1), (2.3.32) or (2.4.1)) used in conjunction with the ℓ_∞ norm may allow a better exploitation of differences of scale between coordinates.

2. Among the "purely first-order" methods, `sdba`, `maxg` and `adagrad` are almost indistinguishable form the performance point of view, with a reliability advantage for `adagrad` (the most reliable method in our tests). This means that, at least in those experiments, the suggestion resulting from the theory that OFFO methods may perform comparably to standard first-order methods seems vindicated.

3. The Adam variants (`adamnorm` and `adam`) are clearly outperformed in our tests by the Adagrad ones (`adagnorm` and `adagrad`). We recall that analytical examples where Adam fails do exist, while the convergence of Adagrad is guaranteed.

4. The theoretical difference in global rate of convergence between `adagrad` and `maxg` does not seem to have much impact on the relative performance of these two methods.

5. The use of limited memory Hessian approximation (`adagbfgs3`) appears to enhance the performance of `adagrad`, but this is not the case of the Barzilai-Borwein approximation (`adagbb`) or, remarkably, for the use of the exact Hessian (`adagH`). When these methods fail, this is often because the steplength is too small to allow the truncated conjugate-gradient solver to pick up second-order information in other directions than the negative gradient. What favours the limited memory approach remains unclear at this stage.

We also note that the variants scaled with (2.3.33) (with Γ_0 unknown) did not perform better on small dimensional problems; possibly because the factor n does not dominate the complexity bounds in this case. To illustrate the impact of this scaling in larger cases, we ran a subset of six methods which performed well in Table 2.2 (namely `adagnorm`, `adagrad`, `adagrads`, `maxgnorm`, `maxg` and `maxgs`) on the broyden3d and nlminsurf problems with increasing problem dimension (we used $\epsilon_1 = 10^{-3}$). The results are reported in Table 2.3.

Method	broyden3d					nlminsurf				
n	10	100	1000	10000	100000	256	1034	4096	16384	65536
`adagnorm`	37	71	467	4257	43400	166	503	1791	6038	19239
`adagrad`	200	37809	37809	37809	37809	7966	30795	121164	482025	NR
`adagrads`	134	190	1452	13042	125503	138	532	2981	19414	121934
`maxgnorm`	46	76	285	1138	4520	1699	3978	3867	5355	19424
`maxg`	458	410	462	3362	36609	NC	NC	NC	NC	NR
`maxgs`	76	155	567	2048	7370	1142	1155	5049	6407	30661

Table 2.3: Number of iterations for convergence on the broyden3d and nlminsurf problems as a function of dimension ($\epsilon_1 = 10^{-3}$, NC = more than 10^6 iterations, NR = not run).

Despite the improvement in complexity due to choosing $\theta > 1$ being theoretical (and applies to the worst-case performance), we may still note some positive (if not completely uniform) effect in this table. The interpretation is also blurred somewhat by the fact that `maxg` and `adagrad` converged to local minimas of broyden3d rather than the global one. We

nevertheless note the consistently better performance of `adagnorm` compared to `adagrad` and `adagrads`, possibly illustrating the fact that its complexity bound does not explicitly involve n.

Finally, and although this is a slight digression from the paper's main topic, we report in Table 2.4 how reliability of our selection of OFFO variants is impacted by noise. To obtain these results, we ran the considered methods on all test problems where the evaluations (function[11] and derivatives) are contaminated by 5, 15, 25 or 50 % of relative Gaussian noise with unit variance. In mathematical terms, the i-th entry of the noisy gradient is set to $[g_k]_i = [\nabla_x(x_k)]_i(1+\phi\xi_i)$ where ξ_i is drawn from $\mathcal{N}(0,1)$) and ϕ is the relative noise level. The reliability percentages in the table result from averaging results obtained for ten independent runs.

	$\rho_{\texttt{algo}}$/relative noise level				
algo	0%	5%	15%	25%	50%
adagH	83.19	84.96	84.20	84.71	82.18
adagHs	81.51	81.85	81.91	80.50	77.82
adagbfgs3	78.15	80.50	80.50	80.84	80.18
adagrad	77.31	80.50	80.25	80.17	80.17
adagbb	75.69	80.08	80.17	79.58	79.41
adagbfgs3s	78.99	79.50	70.67	79.41	78.66
adagbbs	73.95	78.15	78.40	78.49	77.06
adagrads	78.15	78.07	78.66	78.74	77.23
maxgs	75.63	76.39	75.46	76.05	74.54
adagnorm	75.63	75.21	75.80	75.71	74.03
maxg	74.79	74.37	75.55	78.15	78.07
maxgnorm	69.75	68.74	69.75	70.84	71.01
adams	42.86	37.98	40.25	44.79	50.84
adamnorm	42.02	37.56	44.96	50.84	55.29
adam	40.34	35.55	36.30	44.03	45.80
sdba	81.51	30.92	31.85	34.87	29.58

Table 2.4: Reliability of OFFO algorithms and steepest descent as a function of the level of relative Gaussian noise ($\epsilon_1 = 10^{-3}$)

As can be seen in the table, the reliability of the `sdba` methods dramatically drops as soon as noise is present, while that of the other OFFO methods is barely affected and remains globally unchanged[12] for increasing noise level. This is consistent with widespread experience in the deep learning context, where noise is caused by sampling among the very large number of terms defining the objective function. This observation vindicates the popularity of methods such as Adagrad in the noisy context and suggests that the new OFFO algorithms may have some practical potential.

We conclude by noting that the algorithms' reliability ($\rho_{\texttt{algo}}$ is (expectedly) better for $\epsilon_1 = 10^{-3}$ (first column of Table 2.4) than for $\epsilon_1 = 10^{-6}$ (Table 2.2), but that the improvement remains modest, the reliability of Adagrad decreasing marginally slower.

[11] For `sdba`.

[12] It is interesting that reliability is slightly better for the noisy cases and the better OFFO methods.

2.5.1 Deep Learning experiments

We now provide some numerical illustrations of the algorithmic variants discussed in the previous sections in the context of image recognition with artificial neural networks. We trained a simple convolutional network from [99] (denoted in the paper as cifar10-nv) and a small resnet18 model [122] on the CIFAR-10 and CIFAR-100 image classification datasets, on the SVHN dataset. For these experiments, we used Haiku [124] and optax [6] two JAX [35]-based libraries on a workstation with four GTX 1080TI. We now compare the numerical performance of (2.3.1) for different μ values in $(0.1, 0.5, 0.9)$ in with both ϑ and θ equal to one. We also consider two algorithms from the "divergent step-size" class, denoted as `avrgi` and `maxgi`, which we describe in detail in Table 2.5. Of course, we can no longer compute the true gradient in (2.2.1), so we need to compute an approximate one. The experimental setup is briefly outlined here, and more detailed explanations on the objective function, datasets, and terminology can be found in the appendix A.1 and B.2.

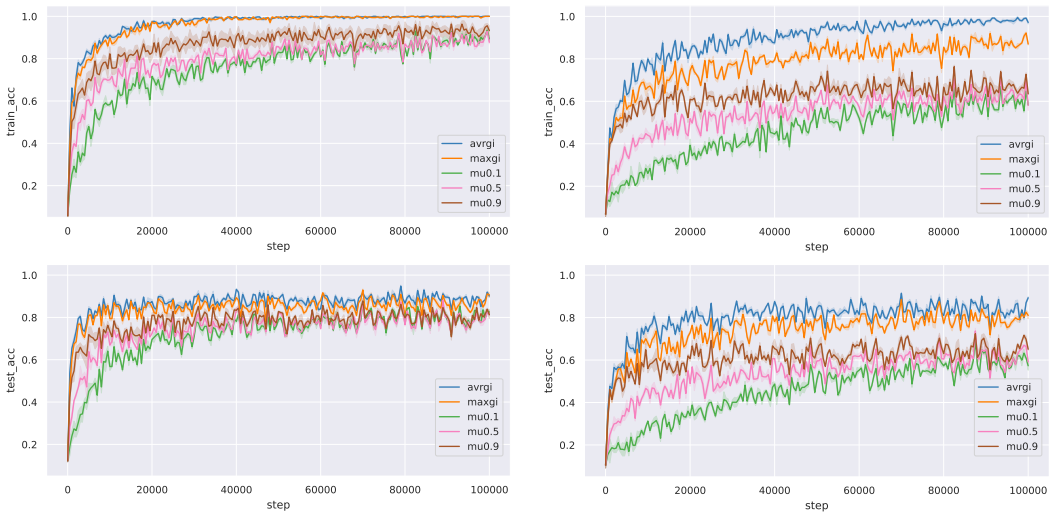

Figure 2.1: CIFAR10: Training (top) and test (bottom) accuracies for `adagrad`-like ($\mu \in (0.1, 0.5, 0.9)$), `maxgi` and `avrgi` variants with $\alpha = 5.10^{-4}$ (left) and $\alpha = 5.10^{-5}$ (right) on the *cifar-nv* architecture

Name	Norm	$w_{i,k}$ definition ($i \in \{1, \dots, n\}$)	B_{k+1}	params
`maxgi`	$\|\cdot\|_\infty$	$w_{i,k} = (k+1)^{\frac{1}{10}} \max\left[\frac{1}{100}, \max_{j\in\{0,\dots,k\}} \|g_{i,j}\|\right]$	0	
`avrgi`	$\|\cdot\|_\infty$	$w_{i,k} = (k+1)^{\frac{1}{10}} \max\left[\frac{1}{100}, \frac{1}{k+1}\sum_{j=0}^{k} \|g_{i,j}\|\right]$	0	

Table 2.5: algorithmic variants for Deep Learning.

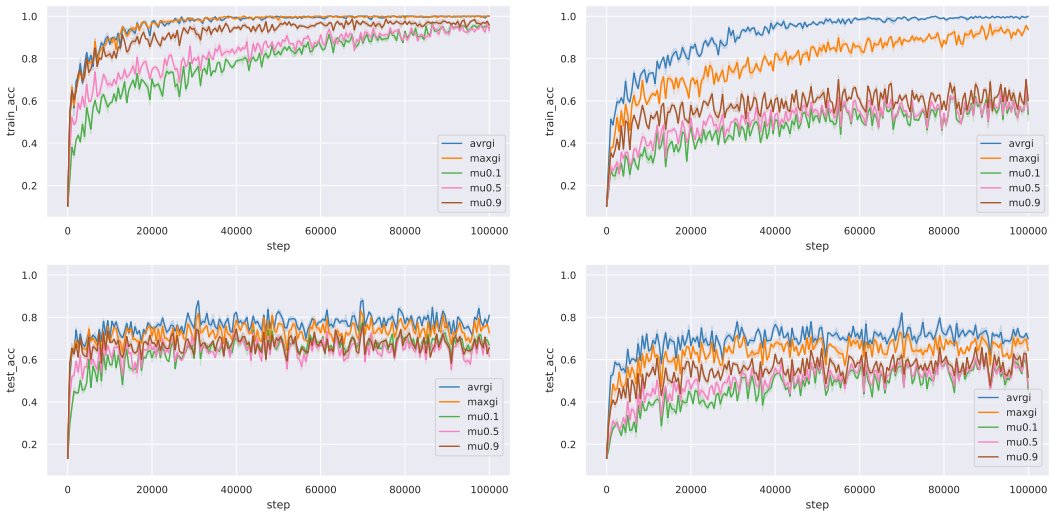

Figure 2.2: CIFAR10: Training (top) and test (bottom) accuracies for `adagrad`-like ($\mu \in (0.1, 0.5, 0.9)$), `maxgi` and `avrgi` variants with $\alpha = 5.10^{-4}$ (left) and $\alpha = 5.10^{-5}$ (right) on the *resnet18* architecture

Figure 2.3: CIFAR100: Training (top) and test (bottom) accuracies for `adagrad`-like ($\mu \in (0.1, 0.5, 0.9)$), `maxgi` and `avrgi` variants with $\alpha = 5.10^{-4}$ (left) and $\alpha = 5.10^{-5}$ (right) on the *cifar-nv* architecture

For all experiments, we also used a fixed learning rate policy with $\alpha_k = \alpha = 5.10^{-4}, 10^{-5}$ for all $k \geq 0$. We used the same random initialization for all scaling choices and followed the data augmentation procedure from [99], for training dataset. We trained the models for a total of 100000 steps with a batch size of 128 using the mean cross entropy loss function. We report the training and testing accuracies (the latter on a sample of size 128 from the test dataset) every 500 steps.

The results of these experiments (averaged over three random runs) are shown in Fig-

Figure 2.4: CIFAR100: Training (top) and test (bottom) accuracies for `adagrad`-like ($\mu \in (0.1, 0.5, 0.9)$), `maxgi` and `avrgi` variants with $\alpha = 5.10^{-4}$ (left) and $\alpha = 5.10^{-5}$ (right) on the *resnet18* architecture

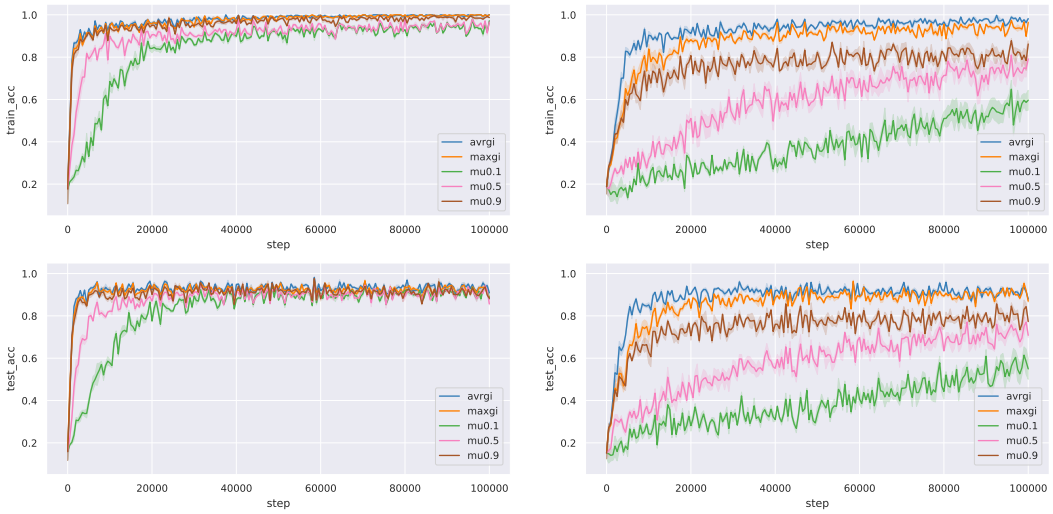

Figure 2.5: SVHN: Training (top) and test (bottom) accuracies for `adagrad`-like ($\mu \in (0.1, 0.5, 0.9)$), `maxgi` and `avrgi` variants with $\alpha = 5.10^{-4}$ (left) and $\alpha = 5.10^{-5}$ (right) on the *cifar-nv* architecture

ures 2.1–2.8. In each figure, the top panel shows the evolution of the training accuracy (as a function of the number of steps), and the bottom panel shows the evolution of the test accuracy. The average values are shown as thick lines, and the shaded areas of the corresponding color give the 67% confidence intervals.

Obviously, these numerical experiments are not intended to replace significant numerical tests, but, while caution must be exercised in extrapolating from limited data, they do suggest some tentative comments.

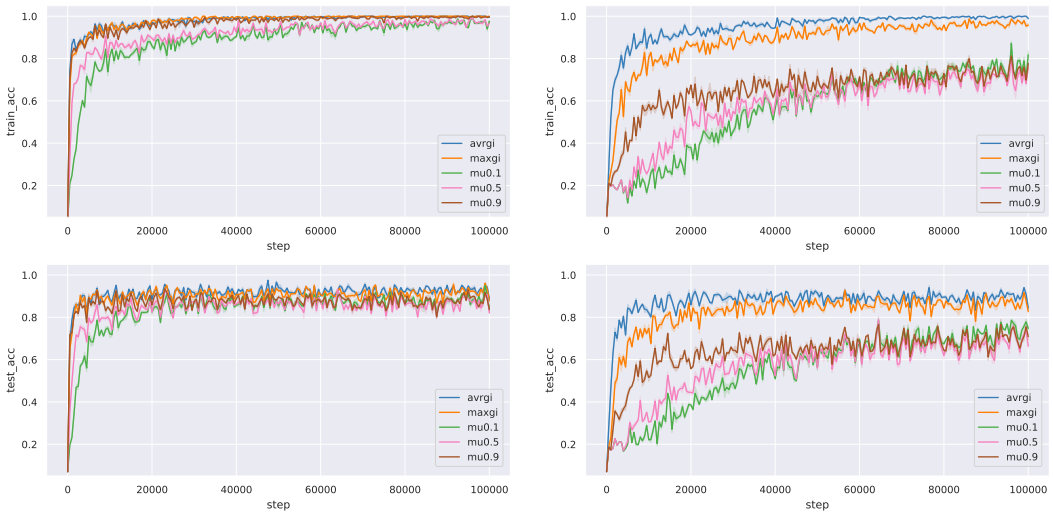

Figure 2.6: SVHN: Training (top) and test (bottom) accuracies for `adagrad`-like ($\mu \in (0.1, 0.5, 0.9)$), `maxgi` and `avrgi` variants with $\alpha = 5.10^{-4}$ (left) and $\alpha = 5.10^{-5}$ (right) on the *resnet18* architecture

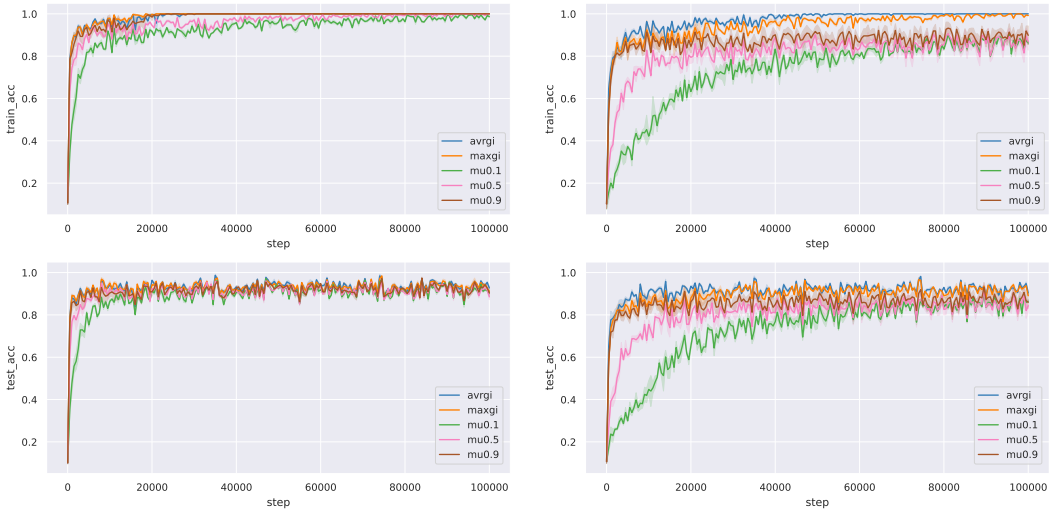

Figure 2.7: FMNIST: Training (top) and test (bottom) accuracies for `adagrad`-like ($\mu \in (0.1, 0.5, 0.9)$), `maxgi` and `avrgi` variants with $\alpha = 5.10^{-4}$ (left) and $\alpha = 5.10^{-5}$ (right) on the *cifar-nv* architecture

- For fixed learning rates, the methods `maxgi` and `avrgi` of the class introduced in Section 2.4 seem to produce relatively good results on our example, both in training and in testing, often outperforming the `adagirad` variants of the first class.
- The relative behavior of the tested variants does not differ significantly between the two network architectures, although the test accuracy is slightly lower for the *resnet18* case.
- Among the `adagrad` variants, those with a larger μ handle smaller learning rates better

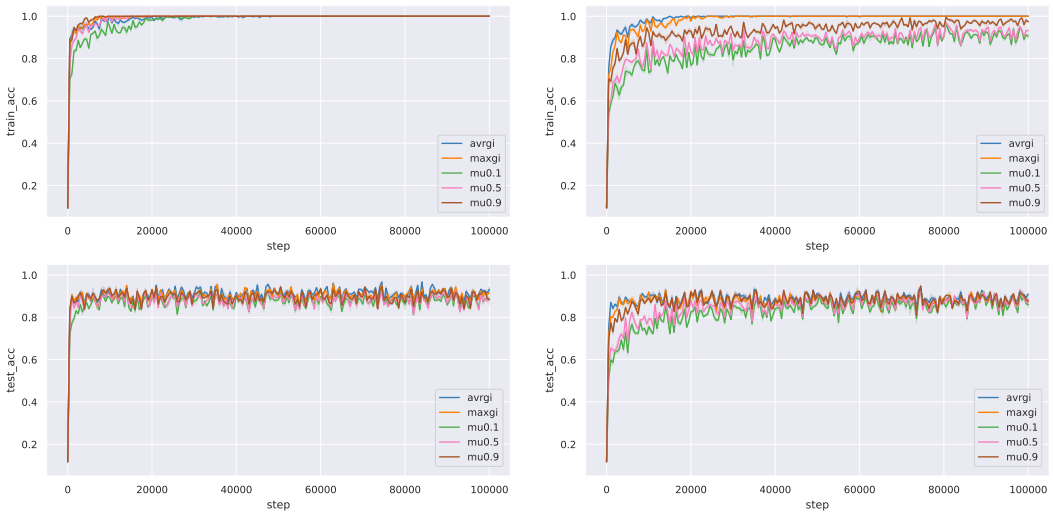

Figure 2.8: FMNIST: Training (top) and test (bottom) accuracies for `adagrad`-like ($\mu \in (0.1, 0.5, 0.9)$), `maxgi` and `avrgi` variants with $\alpha = 5.10^{-4}$ (left) and $\alpha = 5.10^{-5}$ (right) on the *resnet18* architecture

for these examples.

Of course, a more comprehensive numerical analysis is essential to thoroughly evaluate the potential of these methods within modern machine learning paradigms. Specifically, with regard to determining the optimal power μ for `adagrad`, it is worth noting that [61] discovered that an exponent of $\frac{1}{8}$ yielded the best results when applied to the `adam` variant of Table 2.1. It is worth noting that our initial findings have also been corroborated to some extent by the research of [119] in the context of multiobjective OFFO settings.

2.6 Discussion

We have presented a parametric class of deterministic "trust-region minded" extensions of the Adagrad method, allowing for the use of second-order information, should it be available. We then prove that, for OFFO algorithms in this class, $\min_{j\in\{0,\ldots,k\}} \|g_j\| = \mathcal{O}(1/\sqrt{k+1})$. We have also shown that this bound, which does not require any uniform bound on the gradient, is essentially sharp. It is *identical to the global rate of convergence of standard first-order methods using both objective-function and gradient evaluations*, despite the fact that the latter exploit significantly more information. Thus, *if one considers the order of global convergence only, evaluating the objective-function values is an unnecessary effort.* We have also considered another class of OFFO algorithms inspired by the "diminishing stepsize" paradigm in non-smooth convex optimization and have provided an essentially sharp (but slighlty worse) global rate of convergence for this latter class. Limited numerical experiments suggest that the above theoretical conclusions may translate to practice and remain, for OFFO methods, relatively independent of noise.

Although discussed here in the context of unconstrained optimization, adaptation of the above OFFO algorithms to problems involving convex constraints (such as bounds on the

variables) is relatively straightforward and practical: one then needs to intersect the trust-region with the feasible set and minimize the quadratic model in this intersection (see [68, Chapter 12]).

Chapter 3

Higher Order Objective-Function-Free Optimization

Chapter Abstract

We develop an adaptive regularization algorithm for unconstrained nonconvex optimization in which the objective function is never evaluated, but only derivatives are used. It is shown that these excellent complexity bounds are also valid for the new algorithm, despite the fact that significantly less information is used. In particular, it is shown that, if derivatives of degree one to p are used and with the assumption that the pth derivative is β Hölder smooth, the algorithm will find an ϵ_1-approximate first-order minimizer in at most $\mathcal{O}(\epsilon_1^{-(p+\beta)/(p-1+\beta)})$ iterations, and an (ϵ_1, ϵ_2)-approximate second-order minimizer in at most $\mathcal{O}(\max(\epsilon_1^{-(p+\beta)/(p-1+\beta)}, \epsilon_2^{-(p+\beta)/(p-2+\beta)}))$ iterations. Initial experiments with noisy derivatives demonstrate the benefits of the objective-free optimization framework compared to standard adaptive regularization algorithms.

Reference: This chapter builds upon a publication in the SIAM Journal on Optimization [116]. In comparison to the aforementioned work, we extend our scope to encompass the broader class of Hölder-smooth derivatives. Furthermore, our analysis has been refined to eliminate the prerequisite of a negative curvature condition for Hölder-smooth Hessian functions.

3.1 Introduction

In this chapter, we delve deeper into the elements mentioned in Section 1.5 regarding high-order OFFO algorithms that were previously introduced.

The theory developed here combines elements of standard adaptive regularization methods such as ARp [29] and of the OFFO approaches of [207] and [109]. We exhibit an OFFO regularization method whose iteration complexity is identical to that obtained when objective function values are used. In particular, we consider convergence to approximate first-order and second-order critical points, and provide sharp complexity bounds depending on the degree of derivatives used.

The Chapter is organized as follows. After introducing the new algorithm in Section 3.2, we present and discuss a first-order worst-case complexity analysis in Section 3.3, while convergence to approximate second-order minimizers is considered in Section 3.4. Some numerical experiments showing the impact of noise are then presented in Section 3.5. Discussions are outlined in Section 3.6.

3.2 An OFFO adaptive regularization algorithm

We now consider the problem of finding approximate minimizers of the unconstrained non-convex optimization problem (P) where f is a sufficiently smooth function from $\mathbb{R}^n$ into $\mathbb{R}$. As motivated in the introduction, our aim is to design an algorithm in which the objective function value is never computed. Our approach is based on regularization methods. In such methods, a model of the objective function is build by "regularizing" a truncated Taylor expansion of degree $p \geq 1$. We now detail the assumption on the problems that ensure this approach makes sense.

AS.1 f is a function of the $\mathcal{C}^{p,\beta}(\mathbb{R}^n;\mathbb{R})$ class. To revisit the definition, we refer to paragraph 1.3.3.1 in the Introduction.

AS.2 There exists a constant $f_{\rm low}$ such that $f(x) \geq f_{\rm low}$ for all $x \in \mathbb{R}^n$.

AS.3 If $p > 1$, there exists a constant $\kappa_{\rm high} \geq 0$ such that

$$\min_{\|d\|\leq 1} \nabla_x^i f(x)[d]^i \geq -\kappa_{\rm high} \ \text{ for all } \ x \in \mathbb{R}^n \ \text{ and } \ i \in \{2,\ldots,p\}, \tag{3.2.1}$$

(For notational convenience, we set $\kappa_{\rm high} = 0$ if $p = 1$.)

We note that AS.3 is weaker than assuming uniform boundedness of the derivative tensors of degree two and above (there is no upper bound on the value of $\nabla_x^i f(x)[d]^i$), or, equivalently, Lipschitz continuity of derivatives of degree one to $p-1$.

3.2.1 The OFFARp algorithm

Our proposed algorithm follows the outline line of existing ARp regularization methods [51, 29, 57] that were showcased in Subsubsection 1.3.3, with the significant difference that the objective function $f(x_k)$ is never computed, and therefore that the ratio of achieved to predicted reduction (a standard feature for these methods) cannot be used to accept or reject a potential new iterate and to update the regularization parameter. Instead, such potential iterates are always accepted and the regularization parameter is updated in a manner independent of this ratio. We now state the resulting OFFARp algorithm in detail on the next page.

The test (3.2.7) follows [111] and extends the more usual condition where the step s_k is chosen to ensure that

$$\|\nabla_s^1 m_k(s_k)\| \leq \theta_1 \|s_k\|^{p-1+\beta}.$$

It is indeed easy to verify that (3.2.7) holds at a local minimizer of m_k with $\theta_1 \geq 1$ (see [111] for details). The motivation for the introduction of $\mu_{1,k}$ in (3.2.4) and (3.2.5) will become clearer after Lemma 3.3.3.

We emphasize the fact that our algorithm and subsequent development assume knowledge of the Hölder exponent, unlike the approaches proposed in [53] or [84].

Algorithm 3.2.1: OFFO adaptive regularization of degree p (OFFARp)

Step 0: Initialization: An initial point $x_0 \in \mathbb{R}^n$, a regularization parameter $\nu_0 > 0$ and a requested final gradient accuracy $\epsilon_1 \in (0, 1]$ are given, as well as the parameters

$$\theta_1 > 1 \quad \text{and} \quad \vartheta \in (0, 1]. \tag{3.2.2}$$

Set $k = 0$.

Step 1: Check for termination: Evaluate $g_k = \nabla_x^1 f(x_k)$. Terminate with $x_\epsilon = x_k$ if

$$\|g_k\| \leq \epsilon_1. \tag{3.2.3}$$

Else, evaluate $\{\nabla_x^i f(x_k)\}_{i=2}^p$.

Step 2: Step calculation: If $k > 0$, set

$$\mu_{1,k} = \frac{(p-1+\beta)!\|g_k\|}{\|s_{k-1}\|^{p-1+\beta}} - \theta_1 \sigma_{k-1} \tag{3.2.4}$$

and select

$$\sigma_k \in \Big[\vartheta\nu_k, \max\left(\nu_k, \mu_{1,k}\right)\Big]. \tag{3.2.5}$$

Otherwise (i.e. if $k = 0$), set $\sigma_0 = \mu_{1,0} = \nu_0$.
Then compute a step s_k which sufficiently reduces the model m_k defined in (1.3.18) in the sense that

$$m_k(s_k) - m_k(0) < 0 \tag{3.2.6}$$

and

$$\|\nabla_s^1 T_{f,p}(x_k, s_k)\| \leq \theta_1 \frac{\sigma_k}{(p-1+\beta)!}\|s_k\|^{p-1+\beta}. \tag{3.2.7}$$

Step 3: Updates: Set

$$x_{k+1} = x_k + s_k \tag{3.2.8}$$

and

$$\nu_{k+1} = \nu_k + \nu_k\|s_k\|^{p+\beta}. \tag{3.2.9}$$

Increment k by one and go to Step 1.

3.3 Evaluation complexity for the OFFARp algorithm

Before discussing our analysis of evaluation complexity, we first restate some classical lemmas for ARp algorithms, starting with Lipschitz error bounds.
We start by stating a simple lower bound on the Taylor series' decrease.

Lemma 3.3.1

$$\Delta T_{f,p}(x_k, s_k) \stackrel{def}{=} T_{f,p}(x_k, 0) - T_{f,p}(x_k, s_k) > \frac{\sigma_k}{(p+\beta)!}\|s_k\|^{p+\beta}. \tag{3.3.1}$$

Proof. The bound directly results from (3.2.6) and (1.3.18). □

This and AS.2 allow us to establish a lower bound on the decrease in the objective function (although it is never computed).

Lemma 3.3.2 *Suppose that AS.1 holds and that $\sigma_k \geq 2L_p$. Then*

$$f(x_k) - f(x_{k+1}) > \frac{\sigma_k}{2(p+\beta)!}\|s_k\|^{p+\beta}. \tag{3.3.2}$$

Proof. From (1.3.15) and (3.3.1), we obtain that

$$f(x_k) - f(x_{k+1}) > \frac{\sigma_k - L_p}{(p+\beta)!}\|s_k\|^{p+\beta}$$

and (3.3.2) immediately follows from our assumption on σ_k. □

The next lemma provides a useful lower bound on the step length, in the spirit of [29, Lemma 2.3] or [111].

Lemma 3.3.3 *Suppose that AS.1 holds. Then, for all $k \geq 0$,*

$$\|s_k\|^{p-1+\beta} \geq \frac{(p-1+\beta)!}{L_p + \theta_1\sigma_k}\|g_{k+1}\|, \tag{3.3.3}$$

$$\mu_{1,k} \leq \max(\nu_0, L_p) \tag{3.3.4}$$

and

$$\sigma_k \leq L_p + \nu_k. \tag{3.3.5}$$

Proof. Successively using the triangle inequality, condition (3.2.7) and (1.3.16), we

deduce that

$$\begin{aligned}\|g_{k+1}\| &\leq \|g_{k+1} - \nabla_s^1 T_{f,p}(x_k, s_k)\| + \|\nabla_s^1 T_{f,p}(x_k, s_k)\| \\ &\leq \frac{1}{(p-1+\beta)!} L_p \|s_k\|^{p-1+\beta} + \theta_1 \frac{\sigma_k}{(p-1+\beta)!} \|s_k\|^{p-1+\beta}.\end{aligned}$$

The inequality (3.3.3) follows by rearranging the terms. Combining this inequality with (3.2.4) and the identity $\mu_{1,0} = \nu_0$ then gives (3.3.4), from which (3.3.5) directly follows using (3.2.5). □

Observe that (3.3.4) motivates our choice in (3.2.5) to allow the regularization parameter to be of size $\mu_{1,k}$.

Inspired by [109, Lemma 7], we now establish an upper bound on the number of iterations needed to enter the algorithm's phase where Lemma 3.3.2 applies and thus all iterations produce a decrease in the objective function.

Lemma 3.3.4 *Suppose that AS.1 holds, and that the OFFARp algorithm does not terminate before or at iteration of index*

$$k \geq k_* \stackrel{def}{=} \left\lceil \left(\frac{2L_p}{\epsilon_1 \vartheta (p-1+\beta)!} \left((1+\theta_1)\frac{L_p}{\sigma_0} + \theta_1 \right) \right)^{\frac{p+\beta}{p-1+\beta}} \right\rceil. \tag{3.3.6}$$

Then for $k \geq k_*$,

$$\nu_k \geq \frac{2L_p}{\vartheta} \tag{3.3.7}$$

which implies that

$$\sigma_k \geq 2L_p. \tag{3.3.8}$$

Proof. Note that (3.3.8) is a direct consequence of (3.2.5) if (3.3.7) is true. Suppose the opposite and that for some $k \geq k_*$, $\nu_k < \frac{2L_p}{\vartheta}$. Since ν_k is a non-decreasing sequence, we have that $\nu_j < \frac{2L_p}{\vartheta}$ for $j \in \{0, \ldots, k\}$. Successively using the form of the ν_k update rule (3.2.9), (3.3.3), (3.3.5) and the fact that, if the algorithm has reached iteration k_*, it must be that (3.2.3) has failed for all iterations of index at most k_*, we derive that

$$\begin{aligned}\nu_k &\overset{(3.2.9)}{>} \sum_{j=0}^{k-1} \nu_j \|s_j\|^{p+\beta} \overset{(3.3.3)}{\geq} \sum_{j=0}^{k-1} \nu_j \left(\frac{(p-1+\beta)!\|g_{j+1}\|}{L_p + \theta_1 \sigma_j} \right)^{\frac{p+\beta}{p-1+\beta}} \\ &\overset{(3.2.5)}{\geq} \sum_{j=0}^{k-1} \nu_j \left(\frac{(p-1+\beta)!\|g_{j+1}\|}{L_p + \theta_1 (L_p + \nu_j)} \right)^{\frac{p+\beta}{p-1+\beta}} = \sum_{j=0}^{k-1} \nu_j^{-\frac{1}{p-1+\beta}} \left(\frac{(p-1+\beta)!\|g_{j+1}\|}{(1+\theta_1)\frac{L_p}{\nu_j} + \theta_1} \right)^{\frac{p+\beta}{p-1+\beta}} \\ &> \sum_{j=0}^{k-1} \nu_j^{-\frac{1}{p}} \left(\frac{(p-1+\beta)!\|g_{j+1}\|}{(1+\theta_1)\frac{L_p}{\sigma_0} + \theta_1} \right)^{\frac{p+\beta}{p-1+\beta}} > \frac{k_* \, \vartheta^{\frac{1}{p-1+\beta}} ((p-1+\beta)!\epsilon_1)^{\frac{p+\beta}{p-1+\beta}}}{(2L_p)^{\frac{1}{p-1+\beta}} \left((1+\theta_1)\frac{L_p}{\sigma_0} + \theta_1 \right)^{\frac{p+\beta}{p-1+\beta}}}.\end{aligned}$$

Substituting the definition of k_* in the last inequality, we obtain that

$$\frac{2L_p}{\vartheta} < \nu_{k_*} < \frac{2L_p}{\vartheta},$$

which is impossible. Hence no index $k \geq k_*$ exists such that $\nu_k < \frac{2L_p}{\vartheta}$ and (3.3.7) and (3.3.8) hold. □

Observe that (3.3.6) depends on the ratio $\frac{L_p}{\sigma_0}$ which is the fraction by which σ_0 underestimates the Lipschitz constant. This ratio will percolate in the rest of our analysis. We now define

$$k_1 = \min\left\{k \geq 1 \mid \nu_k \geq \frac{2L_p}{\vartheta}\right\}, \tag{3.3.9}$$

the first iterate such that significant objective function decrease is guaranteed. The next series of Lemmas provides bounds on $f(x_{k_1})$ and σ_{k_1}, which in turn will allow establishing an upper bound on the regularization parameter. An initial step is to provide an upper-bound on $\|s_k\|$. But first let us proof a bound on the root of a polynomial where the degree is not an integer. The latter resuly will be useful to derive an upper-bound on $\|s_k\|$.

Lemma 3.3.5 *Let $(a_0, \cdots, a_n) \in \mathbb{R}^{n+1}$ with $a_n \neq 0$, $n \geq 1$ and $\beta \in (0, 1]$. Define*

$$\kappa_m \stackrel{def}{=} \max\left\{\left(\frac{a_{n-i}}{a_n}\right)^{\frac{1}{i-1+\beta}}; 1 \leq i \leq n\right\} \tag{3.3.10}$$

and let $a_n x^{n-1+\beta} + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0 = 0$. Then

$$|x| \leq 2\kappa_m \tag{3.3.11}$$

which directly implies that

$$|x| \leq 2\sum_{i=1}^{n}\left(\frac{a_{n-i}}{a_n}\right)^{\frac{1}{i-1+\beta}}. \tag{3.3.12}$$

Proof. The proof closely follows the approach presented in [216, Lecture VI, Lemma 5] with minor adjustments and is therefore deffered to the Appendix. □

Equipped with the last lemma, we are now able to proof the following bound on the stepsize length.

Lemma 3.3.6 *Suppose that AS.1 and AS.3 hold. At each iteration k, we have that*

$$\|s_k\| \leq 2\eta + 2\left(\frac{(p+\beta)!\|g_k\|}{\sigma_k}\right)^{\frac{1}{p-1+\beta}}, \tag{3.3.13}$$

where

$$\eta = \sum_{i=2}^{p}\left[\frac{\kappa_{high}(p+\beta)!}{i!\,\vartheta\nu_0}\right]^{\frac{1}{p-i+\beta}}. \tag{3.3.14}$$

Proof. If $p = 1$, we obtain from (3.2.6) and the Cauchy-Schwarz inequality that

$$\tfrac{1}{(\beta+1)}\sigma_k\|s_k\|^{\beta+1} < -g_k^T s_k \leq \|g_k\|\,\|s_k\|$$

and (3.3.13) holds with $\eta = 0$. Suppose now that $p > 1$. Again (3.2.6) and (3.2.1) of AS.3 give that

$$\frac{\sigma_k}{(p+\beta)!}\|s_k\|^{p+\beta} \leq -g_k^T s_k - \sum_{i=2}^{p}\frac{1}{i!}\nabla_x^i f(x_k)[s_k]^i \leq \|g_k\|\|s_k\| + \sum_{i=2}^{p}\frac{\kappa_{\rm high}}{i!}\|s_k\|^i.$$

Applying now Lemma 3.3.5 with $x = \|s_k\|$, $n = p+1$, $a_0 = 0$, $a_1 = \|g_k\|$, $\beta = \beta$, $a_i = \kappa_{\rm high}/i!$ $i \in \{2,\ldots,n-1\}$ and $a_{n+1} = -\sigma_k/(p+\beta)!$, we know from (3.2.6) that the equation $\sum_{i=0}^{n-1} a_i x^i + a_n x^{p+\beta} = 0$ admits at least one strictly positive root, and we may thus derive that

$$\begin{aligned}
\|s_k\| &\leq 2\left(\frac{(p+\beta)!\|g_k\|}{\sigma_k}\right)^{\frac{1}{p-1+\beta}} + 2\sum_{i=2}^{p}\left[\frac{\kappa_{\rm high}(p+\beta)!}{i!\,\sigma_k}\right]^{\frac{1}{p-i+\beta}} \\
&\leq 2\left(\frac{(p+\beta)!\|g_k\|}{\sigma_k}\right)^{\frac{1}{p-1+\beta}} + 2\sum_{i=2}^{p}\left[\frac{\kappa_{\rm high}(p+\beta)!}{i!\,\vartheta\nu_k}\right]^{\frac{1}{p-i+\beta}} \\
&\leq 2\left(\frac{(p+\beta)!\|g_k\|}{\sigma_k}\right)^{\frac{1}{p-1+\beta}} + 2\sum_{i=2}^{p}\left[\frac{\kappa_{\rm high}(p+\beta)!}{i!\,\vartheta\nu_0}\right]^{\frac{1}{p-i+\beta}},
\end{aligned}$$

and (3.3.13) holds with (3.3.14). □

Our next step is to prove that ν_{k_1} is bounded by constants only depending on the problem and the fixed algorithmic parameters.

Lemma 3.3.7 *Suppose that AS.1 and AS.3 hold. Let k_1 be defined by (3.3.9). We have that,*

$$\nu_{k_1} \leq \nu_{\max} = \max\left(\sigma_0 + \sigma_0\left(2\eta + 2\left(\frac{(p+\beta)!\|g_0\|}{\sigma_0}\right)^{\frac{1}{p-1+\beta}}\right)^{p+\beta}, \frac{2\kappa_1 L_p}{\vartheta}\right) \tag{3.3.15}$$

where η is defined in (3.3.14) and

$$\kappa_1 \stackrel{def}{=} 1 + 2^{2p+2\beta-1}\eta^{p+\beta} + 2^{2p+2\beta-1}\left[\frac{p+\beta}{\vartheta}\left((1+\theta_1)\frac{L_p}{\sigma_0} + \theta_1\right)\right]^{\frac{p+\beta}{p-1+\beta}}. \tag{3.3.16}$$

Proof. If $k_1 = 1$, we have that $\nu_1 = \sigma_0 + \sigma_0\|s_0\|^{p+\beta}$. Using Lemma 3.3.6 to bound $\|s_0\|^{p+\beta}$, we derive the bound corresponding to the first term in the maximum of (3.3.15). Suppose now that $k_1 \geq 2$. Successively using (3.2.9), Lemma 3.3.6, the fact that by convexity $(x+y)^{p+\beta} \leq 2^{p+\beta-1}(x^{p+\beta} + y^{p+\beta})$ for non-negative x, y, the updates rule for ν_k (3.2.9) and σ_k (3.2.5), we derive that

$$\begin{aligned}
\nu_{k_1} &= \nu_{k_1-1} + \nu_{k_1-1}\|s_{k_1-1}\|^{p+\beta} \\
&\stackrel{(3.3.13)}{\leq} \nu_{k_1-1} + \nu_{k_1-1}\left(2\left((p+\beta)!\frac{\|g_{k_1-1}\|}{\sigma_{k_1-1}}\right)^{\frac{1}{p-1+\beta}} + 2\eta\right)^{p+\beta} \\
&\leq \nu_{k_1-1} + 2^{p+\beta-1}\nu_{k_1-1}\left[2^{p+\beta}\eta^{p+\beta} + 2^{p+\beta}\left(\frac{(p+\beta)!\|g_{k_1-1}\|}{\sigma_{k_1-1}}\right)^{\frac{p+\beta}{p-1+\beta}}\right] \qquad (3.3.17) \\
&\stackrel{(3.2.5)}{\leq} \nu_{k_1-1} + 2^{2p+2\beta-1}\nu_{k_1-1}\left[\eta^{p+\beta} + \left(\frac{(p+\beta)!\|g_{k_1-1}\|}{\vartheta\,\nu_{k_1-1}}\right)^{\frac{p+\beta}{p-1+\beta}}\right] \qquad (3.3.18)
\end{aligned}$$

Rearranging the last inequality and using (3.3.3)

$$\begin{aligned}
\nu_{k_1} &\leq \nu_{k_1-1} + 2^{2p+2\beta-1}\nu_{k_1-1}\eta^{p+\beta} + 2^{2p-2\beta+1}\left(\frac{(p+\beta)!}{\vartheta}\right)^{\frac{p+\beta}{p-1+\beta}}\frac{\|g_{k_1-1}\|^{\frac{p+\beta}{p-1+\beta}}}{\nu_{k_1-1}{}^{\frac{1}{p-1+\beta}}} \\
&\stackrel{(3.3.3)}{\leq} \nu_{k_1-1} + 2^{2p+2\beta-1}\nu_{k_1-1}\eta^{p+\beta} \\
&\qquad + 2^{2p+2\beta-1}\left[\frac{(p+\beta)!}{\vartheta\,(p-1+\beta)!}(L_p + \theta_1\sigma_{k_1-2})\right]^{\frac{p+\beta}{p-1+\beta}}\nu_{k_1-1}^{-\frac{1}{p-1+\beta}}\|s_{k_1-2}\|^{p+\beta}.
\end{aligned}$$

Now ν_k is a non decreasing sequence and using (3.3.5),

$$\begin{aligned}
\nu_{k_1} &\overset{(3.3.5)}{\leq} \nu_{k_1-1} + 2^{2p-2\beta+1}\nu_{k_1-1}\eta^{p+\beta} \\
&\quad + 2^{2p+2\beta-1}\left[\frac{p+\beta}{\vartheta}\big((1+\theta_1)L_p + \theta_1\nu_{k_1-2}\big)\right]^{\frac{p+\beta}{p-1+\beta}} \nu_{k_1-2}^{-\frac{1}{p-1+\beta}} \|s_{k_1-2}\|^{p+\beta} \\
&\leq \nu_{k_1-1} + 2^{2p+2\beta-1}\nu_{k_1-1}\eta^{p+\beta} \\
&\quad + 2^{2p+2\beta-1}\left[\frac{p+\beta}{\vartheta}\left((1+\theta_1)\frac{L_p}{\nu_{k_1-2}} + \theta_1\right)\right]^{\frac{p+\beta}{p-1+\beta}} \nu_{k_1-2}^{-\frac{1}{p-1+\beta}} \nu_{k_1-2}^{\frac{p+\beta}{p-1+\beta}} \|s_{k_1-2}\|^{p+\beta}
\end{aligned}$$

From the fact that $\nu_{k_1-2} \geq \nu_0 = \sigma_0$ and the update rule of ν_k (3.2.9)

$$\begin{aligned}
\nu_{k_1} &\leq \nu_{k_1-1} + 2^{2p+2\beta-1}\nu_{k_1-1}\eta^{p+\beta} + \\
&\qquad 2^{2p+2\beta-1}\left[\frac{p+\beta}{\vartheta}\left((1+\theta_1)\frac{L_p}{\sigma_0} + \theta_1\right)\right]^{\frac{p+\beta}{p-1+\beta}} \nu_{k_1-2}\|s_{k_1-2}\|^{p+\beta} + \\
&\overset{(3.2.9)}{=} \nu_{k_1-1} + 2^{2p+2\beta-1}\nu_{k_1-1}\eta^{p+\beta} \\
&\qquad \left[\frac{p+\beta}{\vartheta}\left((1+\theta_1)\frac{L_p}{\sigma_0} + \theta_1\right)\right]^{\frac{p+\beta}{p-1+\beta}} (\nu_{k_1-1} - \nu_{k_1-2}) \\
&\leq \nu_{k_1-1} + 2^{2p-2\beta+1}\nu_{k_1-1}\eta^{p+\beta} + \\
&\qquad 2^{2p+2\beta-1}\left[\frac{p+\beta}{\vartheta}\left((1+\theta_1)\frac{L_p}{\sigma_0} + \theta_1\right)\right]^{\frac{p+\beta}{p-1+\beta}} \nu_{k_1-1}.
\end{aligned}$$

We then obtain the second part of (3.3.15) by observing that $\nu_{k_1-1} \leq \dfrac{2L_p}{\vartheta}$. □

This result allows us to establish an upperbound on $f(x_{k_1})$ as a function of $\nu_{\max}$.

Lemma 3.3.8 *Suppose that AS.1 and AS.3 hold. Then*

$$f(x_{k_1}) \leq f(x_0) + \frac{L_p\nu_{\max}}{(p+\beta)!\sigma_0}. \tag{3.3.19}$$

Proof. From (1.3.15) and (3.3.1), we know that

$$f(x_{j+1}) - f(x_j) \leq (L_p - \sigma_j)\frac{\|s_j\|^{p+\beta}}{(p+\beta)!}. \tag{3.3.20}$$

Using now the identity $\sigma_0 = \nu_0$, (3.2.9) and the fact that ν_k is a non-decreasing function, we derive that

$$\nu_{k_1} \geq \sigma_0 + \sigma_0 \sum_{j=0}^{k_1-1} \|s_j\|^{p+\beta}. \tag{3.3.21}$$

Summing the inequality (3.3.20) for $j \in \{0, \ldots, k_1 - 1\}$, ignoring negative terms, and using (3.3.21), (3.2.9) and (3.2.5), we deduce that

$$\begin{aligned} f(x_{k_1}) &\leq f(x_0) + \frac{L_p}{(p+\beta)!} \sum_{j=0}^{k_1-1} \|s_j\|^{p+\beta} - \frac{1}{(p+\beta)!} \sum_{j=0}^{k_1-1} \sigma_j \|s_j\|^{p+\beta} \\ &\leq f(x_0) + \frac{L_p}{(p+\beta)!} \left(\frac{\nu_{k_1} - \sigma_0}{\sigma_0} \right). \end{aligned}$$

We then obtain (3.3.19) by using Lemma 3.3.7 to bound ν_{k_1}. □

The two bounds in Lemma 3.3.8 and Lemma 3.3.7 are useful in that they now imply an upper bound on the regularization parameter, a crucial step in standard theory for regularization methods.

Lemma 3.3.9 *Suppose that AS.1, AS.2 and AS.3 hold. Suppose also that $k \geq k_1$. Then*

$$\sigma_k \leq \sigma_{\max} \stackrel{def}{=} \max \left(\frac{2(p+\beta)!}{\vartheta} \left(f(x_0) - f_{\rm low} + \frac{L_p \nu_{\max}}{(p+\beta)! \sigma_0} \right) + \nu_{\max},\, L_p,\, \nu_0 \right). \quad (3.3.22)$$

Proof. Let $j \in \{k_1, \ldots, k\}$. By the definition of k_1 in (3.3.9), $\sigma_j \geq 2L_p$. From Lemma 3.3.2, we then have that

$$f(x_j) - f(x_{j+1}) \geq \frac{\sigma_j}{2(p+\beta)!} \|s_j\|^{p+\beta} \geq \vartheta \frac{\nu_j}{2(p+\beta)!} \|s_j\|^{p+\beta}.$$

Summing the previous inequality from $j = k_1$ to $k-1$ and using the ν_j update rule (3.2.9) and AS.2, we deduce that

$$f(x_{k_1}) - f_{\rm low} \geq f(x_{k_1}) - f(x_k) \geq \frac{\vartheta}{2(p+\beta)!} (\nu_k - \nu_{k_1}).$$

Rearranging the previous inequality and using Lemma 3.3.7,

$$\nu_k \leq \frac{2(p+\beta)!}{\vartheta} \left(f(x_{k_1}) - f_{\rm low} \right) + \nu_{\max}. \quad (3.3.23)$$

Combining now Lemma 3.3.8 (to bound $f(x_{k_1})$), (3.2.5) and (3.3.4) gives (3.3.22). □

We may now resort to the standard "telescoping sum" argument to obtain the desired evaluation complexity bound.

Theorem 3.3.10 *Suppose that AS.1–AS.3 hold. Then the OFFARp algorithm requires at most*

$$\left[\kappa_{OFFARp}\left(f(x_0)-f_{\rm low}+\frac{L_p\nu_{\max}}{(p+\beta)!\sigma_0}\right)\right.$$
$$\left.+\left(\frac{2L_p}{\vartheta(p-1+\beta)!}\left(\frac{L_p}{\sigma_0}(1+\theta_1)+\theta_1\right)\right)^{\frac{p+\beta}{p-1+\beta}}\right]\epsilon_1^{-\frac{p+\beta}{p-1+\beta}}+2$$

iterations and evaluations of $\{\nabla_x^i f\}_{i=1}^p$ *to produce a vector* $x_\epsilon \in \mathbb{R}^n$ *such that* $\|\nabla_x^1 f(x_\epsilon)\| \le \epsilon_1$*, where*

$$\kappa_{OFFARp} \stackrel{def}{=} 2(p+\beta)!\,\sigma_{\max}^{1/(p-1+\beta)}\left(\frac{1}{\vartheta(p-1+\beta)!}\left(\frac{L_p}{\sigma_0}+\vartheta\theta_1\right)\right)^{\frac{p+\beta}{p-1+\beta}}$$

where $\sigma_{\max}$ *is defined in Lemma 3.3.9 and* $\nu_{\max}$ *is defined in Lemma 3.3.7.*

Proof. Suppose that the algorithm terminates at an iteration $k < k_1$, where k_1 is given by (3.3.9). The desired conclusion then follows from the fact that, by this definition and Lemma 3.3.4,

$$k_1 \le k_* \le \left(\frac{2L_p}{\epsilon_1\vartheta(p-1+\beta)!}\left(\frac{L_p}{\sigma_0}(1+\theta_1)+\theta_1\right)\right)^{\frac{p+\beta}{p-1+\beta}}+1. \tag{3.3.24}$$

Suppose now that the algorithm has not terminated at iteration k_1 and consider an iteration $j \ge k_1$. From k_1 definition (3.3.9) and Lemma 3.3.9, we have that $2L_p \le \sigma_j \le \sigma_{\max}$. Since $\sigma_j \ge 2L_p$, Lemma 3.3.2 is valid for iteration j. Combining Lemma 3.3.2, inequality (3.3.3) of Lemma 3.3.3, the fact that $\sigma_j \in [\vartheta\sigma_0, \sigma_{\max}]$ because of Lemma 3.3.9 and that $\|g_{j+1}\| \ge \epsilon_1$ before termination, we therefore deduce that

$$\begin{aligned} f(x_j)-f(x_{j+1}) &\ge \frac{\sigma_j\|s_j\|^{p+\beta}}{2(p+\beta)!} \ge \frac{\sigma_j(p-1+\beta)!^{\frac{p+\beta}{p-1+\beta}}\|g_{j+1}\|^{\frac{p+\beta}{p-1+\beta}}}{2(p+\beta)!(L_p+\theta_1\sigma_j)^{\frac{p+\beta}{p-1+\beta}}} \\ &\ge \frac{(p-1+\beta)!^{\frac{p+\beta}{p-1+\beta}}\epsilon_1^{\frac{p+\beta}{p-1+\beta}}}{2(p+\beta)!\sigma_{\max}^{\frac{1}{p-1+\beta}}\left(\frac{L_p}{\vartheta\sigma_0}+\theta_1\right)^{\frac{p+\beta}{p-1+\beta}}}. \end{aligned} \tag{3.3.25}$$

Summing this inequality from k_1 to $k \ge k_1$ and using AS.3, we obtain that

$$f(x_{k_1})-f_{\rm low} \ge f(x_{k_1})-f(x_k) \ge \frac{(k-k_1)}{\kappa_{\rm OFFARp}}\epsilon_1^{\frac{p+\beta}{p-1+\beta}}. \tag{3.3.26}$$

Rearranging the terms of the last inequality and using (3.3.24) and Lemma 3.3.8 then yields the desired result. □

While this theorem covers all model's degrees, it is worthwhile to isolate the most commonly

used cases.

Corollary 3.3.1 *Suppose that both AS.1 and AS.2 hold and that $p = 1$ with $\beta = 1$. Then the OFFAR1 algorithm requires at most*

$$\left[\frac{4\sigma_{\max}}{\vartheta^2}\left(\frac{L_1}{\sigma_0}+\vartheta\theta_1\right)^2\left[f(x_0)-f_{\rm low}+\frac{L_1\nu_{\max}}{2\sigma_0}\right]+\left(\frac{2L_1}{\vartheta}\left(\frac{L_1}{\sigma_0}(1+\theta_1)+\theta_1\right)\right)^2\right]\epsilon_1^{-2}+2$$

iterations and evaluations of the gradient to produce a vector $x_\epsilon \in \mathbb{R}^n$ such that $\|\nabla_x^1 f(x_\epsilon)\| \leq \epsilon_1$, where $\sigma_{\max}$ is defined in Lemma 3.3.9 and $\nu_{\max}$ is defined in Lemma 3.3.7. If $p = 2$, $\beta = 1$ and AS.3 holds, the OFFAR2 algorithm requires at most

$$\left[\frac{12\sigma_{\max}^{1/2}}{(2\vartheta)^{\frac{3}{2}}}\left(\frac{L_2}{\sigma_0}+\vartheta\theta_1\right)^{\frac{3}{2}}\left[f(x_0)-f_{\rm low}+\frac{L_2\nu_{\max}}{6\sigma_0}\left(\frac{L_2}{\sigma_0}\nu_{\max}+\vartheta\sigma_0\right)\right]+\left(\frac{L_2}{\vartheta}\left(\frac{L_2}{\sigma_0}(1+\theta_1)+\theta_1\right)\right)^{\frac{3}{2}}\right]\epsilon_1^{-\frac{3}{2}}+2$$

iterations and evaluations of the gradient and Hessian to achieve the same result.

We now prove that the complexity bound stated by Corollary 3.3.10 is sharp in order.

Theorem 3.3.11 *Let $\epsilon_1 \in (0,1]$, $\beta = 1$ and $p \geq 1$. Then there exists a p times continuously differentiable function f_p from $\mathbb{R}$ into $\mathbb{R}$ such that the OFFARp applied to f_p starting from the origin takes exactly $k_\epsilon = \lceil \epsilon_1^{-\frac{p+1}{p}} \rceil$ iterations and derivative's evaluations to produce an iterate x_{k_ϵ} such that $|\nabla_x^1 f_p(x_{k_\epsilon})| \leq \epsilon_1$.*

Proof. To prove this result, we first define a sequence of function and derivatives' values such that the gradients converge sufficiently slowly and then show that these sequences can be generated by the OFFARp algorithm and also that there exists a function f_p satisfying AS.1–AS.3 which interpolate them.

First select $\vartheta = 1$, some $\sigma_0 = \nu_0 > 0$ and define, for all $k \in \{0, \ldots, k_\epsilon\}$,

$$\omega_k = \epsilon_1 \frac{k_\epsilon - k}{k_\epsilon} \in [0, \epsilon_1] \tag{3.3.27}$$

and

$$g_k = -(\epsilon_1 + \omega_k) \quad \text{and} \quad D_{i,k} = 0, \quad (i = 2, \ldots, p), \tag{3.3.28}$$

so that

$$|g_k| \in [\epsilon_1, 2\epsilon_1] \subset [0,2] \text{ for all } k \in \{0,\ldots,k_\epsilon\}. \tag{3.3.29}$$

We then set, for all $k \in \{0,\ldots,k_\epsilon\}$,

$$s_k = \left(\frac{p!|g_k|}{\sigma_k}\right)^{\frac{1}{p}}, \tag{3.3.30}$$

Using (3.3.30), we compute $\mu_{1,k}$ for $k > 0$, yielding that

$$\mu_{1,k} = \frac{p!|g_k|}{|s_{k-1}|^p} - \theta_1\sigma_{k-1} = \left(\frac{|g_k|}{|g_{k-1}|} - \theta_1\right)\sigma_{k-1} < 0, \tag{3.3.31}$$

where the last inequality results from the fact that $|g_k|$ is a non-increasing sequence and that $\theta_1 > 1$. Using (3.3.31) and the equality $\vartheta = 1$, we obtain that $\sigma_k = \nu_k$ for all k. Computing now an upper-bound on σ_k

$$\begin{aligned}
\sigma_k &\stackrel{\rm def}{=} \sigma_0 + \sum_{j=0}^{k-1} \sigma_j |s_j|^{p+1} \qquad (3.3.32)\\
&= \sigma_0 + \sum_{j=0}^{k-1} \sigma_j \left(\frac{p!|g_j|}{\sigma_j}\right)^{\frac{p+1}{p}} = \sigma_0 + (p!)^{\frac{p+1}{p}} \sum_{j=0}^{k-1} \frac{(\epsilon_1+\omega_j)^{\frac{p+1}{p}}}{\sigma_j^{\frac{1}{p}}}\\
&\le \sigma_0 + \left(\frac{(2p!)^{p+1}}{\sigma_0}\right)^{\frac{1}{p}} \sum_{j=0}^{k-1} \epsilon_1^{\frac{p+1}{p}} \le \sigma_0 + \left(\frac{(2p!)^{p+1}}{\sigma_0}\right)^{\frac{1}{p}} k_\epsilon \epsilon_1^{\frac{p+1}{p}} \le \sigma_0 + 2\left(\frac{(2p!)^{p+1}}{\sigma_0}\right)^{\frac{1}{p}}\\
&\stackrel{\rm def}{=} \sigma_{\max},
\end{aligned}$$

where we successively used (3.3.30), (3.3.28), (3.3.27) and the definition of k_ϵ. We finally set

$$f_0 = 2^{\frac{2p+1}{p}} \left(\frac{p!}{\sigma_0}\right)^{\frac{1}{p}} \text{ and } f_{k+1} \stackrel{\rm def}{=} f_k + g_k s_k + \sum_{i=2}^{p} \frac{1}{i!} D_{i,k}[s_k]^i = f_k - \left(\frac{p!}{\sigma_k}\right)^{\frac{1}{p}} (\epsilon_1+\omega_k)^{\frac{p+1}{p}},$$

yielding, using (3.3.32) and the definition of k_ϵ, that

$$f_0 - f_{k_\epsilon} = \sum_{k=0}^{k_\epsilon-1} \left(\frac{p!}{\sigma_k}\right)^{\frac{1}{p}} (\epsilon_1+\omega_k)^{\frac{p+1}{p}} \le 2^{\frac{p+1}{p}} \left(\frac{p!}{\sigma_0}\right)^{\frac{1}{p}} k_\epsilon \epsilon_1^{\frac{p+1}{p}} \le 2^{\frac{2p+1}{p}} \left(\frac{p!}{\sigma_0}\right)^{\frac{1}{p}} = f_0.$$

As a consequence

$$f_k \in [0, f_0] \text{ for all } k \in \{0,\ldots,k_\epsilon\}. \tag{3.3.33}$$

Observe that (3.3.30) satisfies (3.2.6) (for the model (1.3.18)) and (3.2.7) for $\theta_1 = 1$. Moreover (3.3.32) is the same as (3.2.9)-(3.2.5). Hence the sequence $\{x_k\}$ generated by

$$x_0 = 0 \text{ and } x_{k+1} = x_k + s_k$$

may be viewed as produced by the OFFARp algorithm given (3.3.28). Defining

$$T_{k,p}(s) = f_k + g_k s + \sum_{i=2}^{p} \frac{1}{i!} D_{i,k} s^i,$$

observe also that

$$|f_{k+1} - T_{k,p}(s_k)| = 0 \leq \frac{2\sigma_{\max}}{p!} |s_k|^{p+1} \tag{3.3.34}$$

and

$$|g_{k+1} - \nabla_s^1 T_{k,p}(s_k)| = |g_{k+1} - g_k| \leq |\omega_k - \omega_{k+1}| = \frac{\epsilon_1}{k_\epsilon} \leq \epsilon_1^{\frac{2p+1}{p}} \leq \frac{\sigma_{\max}}{\sigma_k}(\epsilon_1 + \omega_k) = \frac{\sigma_{\max}}{p!}|s_k|^p \tag{3.3.35}$$

(we used $k_\epsilon \leq \epsilon_1^{-\frac{p+1}{p}} + 1$ and $\epsilon_1 \leq 1$), while, if $p > 1$,

$$|D_{i,k+1} - \nabla_s^i T_{k,p}(s_k)| = |D_{i,k+1} - D_{i,k}| = 0 \leq \frac{\sigma_{\max}}{p!} |s_k|^{p+1-i} \tag{3.3.36}$$

for $i = 2, \ldots, p$. In view of (3.3.29), (3.3.33) and (3.3.34)-(3.3.36), we may then apply classical Hermite interpolation to the data given by $\{(x_k, f_k, g_k, D_{2,k}, \ldots, D_{p,k})\}_{k=0}^{k_\epsilon}$ (see [57, Theorem A.9.2] with $\kappa_f = \max(2, f_0, 2\sigma_{\max}/p!)$, for instance) and deduce that there exists a p times continuously differentiable piecewise polynomial function f_p satisfying AS.1–AS.3 and such that, for $k \in \{0, \ldots, k_\epsilon\}$,

$$f_k = f_p(x_k), \quad g_k = \nabla_x^1 f_p(x_k) \text{ and } D_{i,k} = \nabla_x^i f_p(x_k), \quad (i = 2, \ldots, p).$$

The sequence $\{x_k\}$ may thus be interpreted as being produced by the OFFARp algorithm applied to f_p starting from $x_0 = 0$. The desired conclusion then follows by observing that, from (3.3.27) and (3.3.28),

$$|g_k| > \epsilon_1 \text{ for } k \in \{0, \ldots, k_\epsilon - 1\} \text{ and } |g_{k_\epsilon}| = \epsilon_1.$$

□

It is remarkable that the complexity bound stated by Theorems 3.3.10 and 3.4.5 are identical (in order) to that known for the standard setting where the objective function is evaluated at each iteration. Moreover, the $\mathcal{O}(\epsilon^{-3/2})$ bound for Lipschitz Hessian objective function was shown in [52] to be optimal within a very large class of second-order methods. One then concludes that, from the sole viewpoint of evaluation complexity, the computation of the objective function's values is an unnecessary effort for achieving convergence at optimal speed.

One may also ask, at this point, if keeping track of ν_k is necessary, that is, when considering OFFAR2, if a simplified update of the form

$$\sigma_k = \max(\sigma_{k-1}, \mu_{1,k}) \tag{3.3.37}$$

would not be sufficient to ensure convergence at the desired rate. As we show in appendix, this is not the case, because $\mu_{1,k}$ only measures change in second derivatives along the direction s_{k-1}, thereby producing an underestimate of L_p. As a result, σ_k may fail to reach this

value and a simplified OFFAR2 algorithm using (3.3.37) instead of (3.2.5) may fail to converge altogether. Another mechanism (such as that provided by ν_k) is thus necessary to force the growth of the regularization parameter beyond the Lipschitz constant.

3.4 Second-order optimality

If second-derivatives are available and $p \geq 2$, it is also possible to modify the OFFARp algorithm to obtain second-order optimality guarantees. We thus assume in this section that $p \geq 2$ and restate the algorithm on the following page. In this following, we improve on the previous result of [116, Section 4] by introducing a milder AS.3, denoted AS.3bis, which will be stated later on.

The modified algorithm only differs from that of page 75 by the addition of the second part of (3.4.2), the inclusion of $\mu_{2,k}$ (whose purpose parallels that of $\mu_{1,k}$) and condition (3.4.7) on the step s_k. As was the case for (3.2.7)/(3.4.6), note that (3.4.7) holds with $\theta_2 = 1$ at a second-order minimizer of the model $m_k(s)$, and is thus achievable for $\theta_2 > 1$. Moreover, because the modified algorithm subsumes the original one, all properties derived in the previous section continue to hold. In addition, As now $p \geq 2$, (1.3.17) of Lemma 1.3.1 also holds.

Now that the computed step s_k satisfies the second-order requirement (3.4.7), we introduce the new assumption AS.3bis for the analysis of the MOFFARp algorithm.

AS.3bis if $p > 2$, there exists a constant $\kappa_{\rm high} \geq 0$ such that

$$\min_{\|d\| \leq 1} \nabla_x^i f(x)[d]^i \geq -\kappa_{\rm high} \text{ for all } x \in \mathbb{R}^n \text{ and } i \in \{3, \cdots, p\}, \tag{3.4.10}$$

We note that for the practical numerical case $p = 2$, no condition on the negative curvature is imposed whereas (3.2.1) bounds it.

We now derive a second-order analog of the step lower bound of Lemma 3.3.3.

Lemma 3.4.1 *Suppose that AS.1 holds and that the modified algorithm is applied. Then, for all $k \geq 0$,*

$$\|s_k\|^{p-2+\beta} \geq \frac{(p-2+\beta)!}{L_p + \theta_2 \sigma_k} \Big[-\lambda_{\min}(H_{k+1}) \Big]_+ \tag{3.4.11}$$

and

$$\mu_{2,k} \leq \max(\nu_0, L_p). \tag{3.4.12}$$

Algorithm 3.4.1: Modified OFFO adaptive regularization of degree p (MOFFARp)

Step 0: Initialization: An initial point $x_0 \in \mathbb{R}^n$, a regularization parameter $\nu_0 > 0$, a requested final gradient accuracy $\epsilon_1 \in (0,1]$ and a requested final curvature accuracy $\epsilon_2 \in (0,1]$ are given, as well as the parameters

$$\theta_1, \theta_2 > 1 \quad \text{and} \quad \vartheta \in (0,1] \tag{3.4.1}$$

Set $k = 0$.

Step 1: Check for termination: Evaluate $g_k = \nabla_x^1 f(x_k)$ and $H_k = \nabla_x^2 f(x_k)$. Terminate with $x_\epsilon = x_k$ if

$$\|g_k\| \leq \epsilon_1 \quad \text{and} \quad \lambda_{\min}(H_k) \geq -\epsilon_2. \tag{3.4.2}$$

Else, evaluate $\{\nabla_x^i f(x_k)\}_{i=3}^p$.

Step 2: Step calculation: If $k > 0$, set

$$\mu_{1,k} = \frac{(p-1+\beta)!\|g_k\|}{\|s_{k-1}\|^{p-1+\beta}} - \theta_1 \sigma_{k-1}, \quad \mu_{2,k} = \frac{(p-2+\beta)!\Big[-\lambda_{\min}(H_k)\Big]_+}{\|s_{k-1}\|^{p-2+\beta}} - \theta_2 \sigma_{k-1} \tag{3.4.3}$$

and select

$$\sigma_k \in \Big[\vartheta \nu_k, \max\left(\nu_k, \mu_{1,k}, \mu_{2,k}\right)\Big]. \tag{3.4.4}$$

Otherwise (i.e. if $k = 0$), set $\sigma_0 = \mu_{1,0} = \mu_{2,0} = \nu_0$.
Then compute a step s_k which sufficiently reduces the model m_k defined in (1.3.18) in the sense that

$$m_k(s_k) - m_k(0) < 0, \tag{3.4.5}$$

$$\|\nabla_s^1 T_{f,p}(x_k, s_k)\| \leq \theta_1 \frac{\sigma_k}{(p-1+\beta)!}\|s_k\|^{p-1+\beta} \tag{3.4.6}$$

and

$$\lambda_{\min}\left(\nabla_s^2 T_{f,p}(x_k, s_k)\right) \geq -\theta_2 \frac{\sigma_k}{(p-2+\beta)!}\|s_k\|^{p-2+\beta}. \tag{3.4.7}$$

Step 3: Updates. Set

$$x_{k+1} = x_k + s_k, \tag{3.4.8}$$

and

$$\nu_{k+1} = \nu_k + \nu_k \|s_k\|^{p+\beta}. \tag{3.4.9}$$

Increment k by one and go to Step 1.

Proof. Successively using the triangle inequality, (1.3.17) and (3.4.7), we obtain that

$$\begin{aligned}
\lambda_{\min}(H_{k+1}) &= \min_{\|d\|\leq 1} \nabla_x^2 f(x_{k+1})[d]^2 \\
&= \min_{\|d\|\leq 1} \left(\nabla_x^2 f(x_{k+1})[d]^2 - \nabla_s^2 T_{f,p}(x_k,s_k)[d]^2 + \nabla_s^2 T_{f,p}(x_k,s_k)[d]^2 \right) \\
&\geq \min_{\|d\|\leq 1} \left(\nabla_x^2 f(x_{k+1})[d]^2 - \nabla_s^2 T_{f,p}(x_k,s_k)[d]^2 \right) + \min_{\|d\|\leq 1} \nabla_s^2 T_{f,p}(x_k,s_k)[d]^2 \\
&= \min_{\|d\|\leq 1} \left((\nabla_x^2 f(x_{k+1}) - \nabla_s^2 T_{f,p}(x_k,s_k))[d]^2 \right) + \lambda_{\min}\left(\nabla_s^2 T_{f,p}(x_k,s_k) \right) \\
&\geq -\|\nabla_x^2 f(x_{k+1}) - \nabla_s^2 T_{f,p}(x_k,s_k)\| - \theta_2 \frac{\sigma_k}{(p-2+\beta)!} \|s_k\|^{p-2\beta} \\
&\geq -\frac{L_p}{(p-2+\beta)!} \|s_k\|^{p-2+\beta} - \theta_2 \frac{\sigma_k}{(p-2+\beta)!} \|s_k\|^{p-2+\beta},
\end{aligned}$$

which proves (3.4.11). The bound (3.4.12) then results from the identity $\mu_{2,0} = \nu_0$, (3.4.4) and (3.4.12). □

Observe that (3.4.4), (3.3.4) and (3.4.12) ensure that (3.3.5) continues to hold.

We now have to adapt our argument since the termination test (3.4.2) may fail if either its first or its second part fails. Lemma 3.3.3 then gives a lower bound on the step if the first part fails, while we have to use Lemma 3.4.1 otherwise. This is formalized in the following lemma.

Lemma 3.4.2 *Suppose that AS.1 holds, and that the* MOFFARp *algorithm has reached iteration of index*

$$k \geq k_{**} \stackrel{def}{=} \left\lceil \frac{2L_p}{\kappa_{both}^{p+\beta}\vartheta} \max\left(\left(\frac{2L_p}{\vartheta}\right)^{\frac{1}{p-1+\beta}}, \left(\frac{2L_p}{\vartheta}\right)^{\frac{2}{p-2+\beta}} \right) \max\left(\epsilon_1^{-\frac{p+\beta}{p-1+\beta}}, \epsilon_2^{-\frac{p+\beta}{p-2+\beta}} \right) \right\rceil, \tag{3.4.13}$$

where

$$\kappa_{both} \stackrel{def}{=} \min\left(\left(\frac{(p-1+\beta)!}{(1+\theta_1)\frac{L_p}{\sigma_0}+\theta_1} \right)^{\frac{1}{p-1+\beta}}, \left(\frac{(p-2+\beta)!}{(1+\theta_2)\frac{L_p}{\sigma_0}+\theta_2} \right)^{\frac{1}{p-2+\beta}} \right). \tag{3.4.14}$$

Then, for $k \geq k_{**}$,

$$\nu_k \geq \frac{2L_p}{\vartheta}, \tag{3.4.15}$$

which implies that, for $k \geq k_{**}$,

$$\sigma_k \geq 2L_p. \tag{3.4.16}$$

Proof. As in Lemma 3.3.4, (3.4.16) is a direct consequence of (3.4.4) if (3.4.15) is true. In order to adapt the proof of Lemma 3.3.4, we observe that, at iteration k, (3.3.3)

and (3.4.11) hold and

$$\|s_k\| \geq \min\left(\left(\frac{(p-1+\beta)!}{L_p+\theta_1\sigma_k}\|g_{k+1}\|\right)^{\frac{1}{p-1+\beta}}, \left(\frac{(p-2+\beta)!}{L_p+\theta_2\sigma_k}\max\left[-\lambda_{\min}(H_{k+1})\right]\right)^{\frac{1}{p-2+\beta}}\right)$$

which, given (3.3.5), that termination has not yet occured and that $\nu_k > \nu_0 = \sigma_0$, implies that

$$\begin{aligned}\|s_k\| &\geq \min\left(\nu_k^{-\frac{1}{p-1+\beta}}\left(\frac{(p-1+\beta)!}{(1+\theta_1)\frac{L_p}{\sigma_0}+\theta_1}\right)^{\frac{1}{p-1+\beta}}, \nu_k^{-\frac{1}{p-2+\beta}}\left(\frac{(p-2+\beta)!}{(1+\theta_2)\frac{L_p}{\sigma_0}+\theta_2}\right)^{\frac{1}{p-2+\beta}}\right)\\ \min\left(\epsilon_1^{\frac{1}{p-1+\beta}}, \epsilon_2^{\frac{1}{p-2+\beta}}\right) &\geq \kappa_{\text{both}}\min\left(\nu_k^{-\frac{1}{p-1+\beta}}, \nu_k^{-\frac{1}{p-2+\beta}}\right)\min\left(\epsilon_1^{\frac{1}{p-1+\beta}}, \epsilon_2^{\frac{1}{p-2+\beta}}\right).\end{aligned} \tag{3.4.17}$$

Suppose now that (3.4.15) fails, i.e. that for some $k \geq k_{**}$, $\nu_k < \frac{2L_p}{\vartheta}$. Since ν_k is a non-decreasing sequence, we have that $\nu_j < \frac{2L_p}{\vartheta}$ for $j \in \{0,\ldots,k\}$. Successively using (3.4.9) and (3.4.17), we obtain that

$$\begin{aligned}\nu_k &> \sum_{j=0}^{k-1}\nu_j\|s_j\|^{p+\beta} \geq \sum_{j=0}^{k_{**}-1}\nu_j\|s_j\|^{p+\beta} \geq \sum_{j=0}^{k_{**}-1}\kappa_{\text{both}}^{p+\beta}\min\left(\nu_j^{-\frac{1}{p-1+\beta}}, \nu_j^{-\frac{2}{p-2+\beta}}\right)\min\left(\epsilon_1^{\frac{1}{p-1+\beta}}, \epsilon_2^{\frac{1}{p-2+\beta}}\right)^{p+\beta}\\ &\geq \sum_{j=0}^{k_{**}-1}\kappa_{\text{both}}^{p+\beta}\min\left(\left(\frac{2L_p}{\vartheta}\right)^{-\frac{1}{p-1+\beta}}, \left(\frac{2L_p}{\vartheta}\right)^{-\frac{2}{p-2+\beta}}\right)\min\left(\epsilon_1^{\frac{1}{p-1+\beta}}, \epsilon_2^{\frac{1}{p-2+\beta}}\right)^{p+\beta}\\ &= k_{**}\kappa_{\text{both}}^{p+\beta}\min\left(\left(\frac{2L_p}{\vartheta}\right)^{-\frac{1}{p-1+\beta}}, \left(\frac{2L_p}{\vartheta}\right)^{-\frac{2}{p-2+\beta}}\right)\min\left(\epsilon_1^{\frac{1}{p-1+\beta}}, \epsilon_2^{\frac{1}{p-2+\beta}}\right)^{p+\beta}.\end{aligned}$$

Using the definition of k_{**} in the last inequality, we see that

$$\frac{2L_p}{\vartheta} < \nu_{k_{**}} < \frac{2L_p}{\vartheta},$$

which is impossible. Hence no index $k \geq k_{**}$ exists such that $\nu_k < \frac{2L_p}{\vartheta}$ and (3.4.15) and (3.4.16) hold. □

Up until now, we have not utilized our newly introduced AS.3bis assumption. Since AS.3 played a crucial role in establishing both Lemma 3.3.6 and Lemma 3.3.7, we need to establish new variants of these lemmas with AS.3bis instead. We begin by proving a new bound on the step, taking into account negative curvature this time.

Lemma 3.4.3 *Suppose that AS.1 and AS.3bis hold. At each iteration k, we have that*

$$\|s_k\| \leq 2\eta_{bis} + 2\left(\frac{(p+\beta)!\|g_k\|}{\sigma_k}\right)^{\frac{1}{p-1+\beta}} + 2\left((p+\beta)!\frac{[-\lambda_{\min}(H_k)]_+}{2\sigma_k}\right)^{\frac{1}{p-2+\beta}} \tag{3.4.18}$$

where

$$\eta_{bis} = \sum_{i=3}^{p}\left[\frac{\kappa_{high}(p+\beta)!}{i!\,\vartheta\sigma_0}\right]^{\frac{1}{p-i+\beta}}. \tag{3.4.19}$$

Proof. As the proof is similar in essence to the one in Lemma 3.3.6, it will be left in the appendix. □

With the help of the previously established lemma, we are now able to proof a bound on ν_{k_1} under AS.3bis assumption.

Lemma 3.4.4 *Suppose that AS.1 and AS.3bis hold. Let k_1 be defined by* (3.3.9)*. There exists a $\nu_{\max}$ depending on problem constants such that*

$$\nu_{k_1} \leq \nu_{\max}, \tag{3.4.20}$$

where

$$\nu_{\max} \stackrel{def}{=} \max\left(\sigma_0 + \sigma_0\left[2\eta_{bis} + 2\left(\frac{(p+\beta)!\|g_0\|}{\sigma_0}\right)^{\frac{1}{p-1+\beta}} + 2\left((p+\beta)!\frac{[-\lambda_{\min}(H_0)]_+}{2\sigma_0}\right)^{\frac{1}{p-2+\beta}}\right]^{p+\beta}, \right. \tag{3.4.21}$$

$$\left.\left(\kappa_2 + 2^{p+\beta}3^{p-1+\beta}((p-1+\beta)(p+\beta))^{\frac{p+\beta}{p-2+\beta}}\left(\frac{\frac{L_p}{\sigma_0}+\theta_2}{2\vartheta}\right)^{\frac{p+\beta}{p-2+\beta}}\right)\frac{2L_p}{\vartheta}\right).$$

where κ_2 is defined in (3.B.5)*.*

Proof. As the proof is similar to that of Lemma 3.3.7, but with the usage of (3.4.18) instead, we have deferred it to the appendix for clarity and brevity. □

We then continue to use the theory of the previous section as Lemma 3.3.8 and Lemma 3.3.9 still apply but with $\nu_{\max}$ given by (3.4.21). Moreover, the value of k_1 now satisfies the improved bound

$$k_1 \leq k_{\star\star} \tag{3.4.22}$$

instead of $k_1 \leq k_\star$. This directly leads us to the following strengthened complexity result.

Theorem 3.4.5 *Suppose that AS.1, AS.2 and AS.3bis hold and that $p > 1$. Then the* MOFFARp *algorithm requires at most*

$$\left[\kappa_{MOFFARp}\left(f(x_0) - f_{\rm low} + \frac{L_p \nu_{\max}}{(p+\beta)!\sigma_0}\right) + \frac{2L_p}{\kappa_{both}^{p+\beta}\vartheta}\max\left(\left(\frac{2L_p}{\vartheta}\right)^{\frac{1}{p-1+\beta}}, \left(\frac{2L_p}{\vartheta}\right)^{\frac{2}{p-2+\beta}}\right)\right]\max\left(\epsilon_1^{-\frac{p+\beta}{p-1+\beta}}, \epsilon_2^{-\frac{p+1}{p-2+\beta}}\right) + 2$$

iterations and evaluations of $\{\nabla_x^i f\}_{i=1}^p$ to produce a vector $x_\epsilon \in \mathbb{R}^n$ such that $\|\nabla_x^1 f(x_\epsilon)\| \leq \epsilon_1$ and $\lambda_{\min}(\nabla_x^2 f(x_\epsilon)) \geq -\epsilon_2$, where

$$\kappa_{MOFFARp} \stackrel{def}{=} 2(p+\beta)!\max\left(\sigma_{\max}^{1/(p-1+\beta)}\left(\frac{\frac{L_p}{\sigma_0}+\vartheta\theta_1}{\vartheta(p-1+\beta)!}\right)^{\frac{p+\beta}{p-1+\beta}}, \sigma_{\max}^{2/(p-2+\beta)}\left(\frac{\frac{L_p}{\sigma_0}+\vartheta\theta_2}{\vartheta(p-2+\beta)!}\right)^{\frac{p+\beta}{p-2+\beta}}\right)$$

and where $\sigma_{\max}$ is defined in Lemma 3.3.9, $\nu_{\max}$ is defined in Lemma 3.4.4 and κ_{both} in (3.4.14).

Proof. The bound of Theorem 3.3.10 remains valid for obtaining a vector $x_\epsilon \in \mathbb{R}^n$ such that $\|g(x_\epsilon)\| \leq \epsilon_1$, but we are now interested to satisfy the second part of (3.4.2) as well. Using (3.4.11) instead of (3.3.3), we deduce (in parallel to (3.3.25)) that before termination,

$$\begin{aligned} f(x_j) - f(x_{j+1}) &\geq \frac{\sigma_j \|s_j\|^{p+\beta}}{2(p+\beta)!} \\ &\geq \frac{\sigma_j((p-2+\beta)!)^{\frac{p+\beta}{p-2+\beta}}\max(0, -\lambda_{\min}(H_{k+1}))^{\frac{p+\beta}{p-2+\beta}}}{2(p+\beta)!(L_p+\theta_2\sigma_j)^{\frac{p+\beta}{p-2+\beta}}} \\ &\geq \frac{((p-2+\beta)!)^{\frac{p+\beta}{p-2+\beta}}\epsilon_2^{\frac{p+\beta}{p-2+\beta}}}{2(p+\beta)!\sigma_{\max}^{\frac{2}{p-2+\beta}}\left(\frac{L_p}{\vartheta\sigma_0}+\theta_2\right)^{\frac{p+\beta}{p-2+\beta}}}, \end{aligned}$$

so that, summing this inequality from k_1 to $k \geq k_1$ and using AS.3 now gives (in parallel to (3.3.26)) that, before the second part of (3.4.2) is satisfied,

$$f(x_{k_1}) - f_{\rm low} \geq f(x_{k_1}) - f(x_k) \geq \frac{(k-k_1)}{\kappa_{2\rm nd}}\epsilon_2^{\frac{p+\beta}{p-2+\beta}}$$

where

$$\kappa_{2\rm nd} \stackrel{\rm def}{=} 2(p+\beta)!\,\sigma_{\max}^{2/(p-2+\beta)}\left(\frac{\frac{L_p}{\sigma_0}+\vartheta\theta_2}{\vartheta(p-2+\beta)!}\right)^{\frac{p+\beta}{p-2+\beta}}.$$

As a consequence, we deduce, using (3.4.22) and Lemma 3.3.8 to bound $f(x_{k_1})$, that the second part of (3.4.2) must hold at the latest after

$$\kappa_{2\text{nd}} \left(f(x_0) - f_{\text{low}} + \frac{L_p \nu_{\max}}{(p+\beta)! \sigma_0} \right) \epsilon_2^{-\frac{p+\beta}{p-2+\beta}} + k_{**} + 2$$

iterations and evaluations of the derivatives, where k_{**} is defined in (3.4.13). Combining this result with that of Theorem 3.3.10 then yields the desired conclusion. □

Focusing again on the case where $p = 2$ and upperbounding complicated constants, we may state the following corollary.

Corollary 3.4.1 *Suppose that AS.1 (with $\beta = 1$) and AS.2 hold and that $p = 2$. Then there exists constants κ_* such that the MOFFAR2 algorithm requires at most*

$$\kappa_* \max\left(\epsilon_1^{-3/2}, \epsilon_2^{-3}\right)$$

iterations and evaluations of the gradient and Hessian to produce a vector $x_\epsilon \in \mathbb{R}^n$ such that $\|\nabla_x^1 f(x_\epsilon)\| \leq \epsilon_1$ and $\lambda_{\min}(\nabla_x^2 f(x_{k_\epsilon})) \geq -\epsilon_2$.

We finally prove that the complexity for reaching approximate second order points, as stated by Theorem 3.4.5, is also sharp.

Theorem 3.4.6 *Let $\epsilon_1, \epsilon_2 \in (0, 1]$, $\beta = 1$ and $p > 1$. Then there exists a p times continuously differentiable function f_p from $\mathbb{R}$ into $\mathbb{R}$ with Lipschitz pth derivative such that the MOFFARp applied to f_p starting from the origin takes exactly $k_\epsilon = \lceil \epsilon_2^{-\frac{p+1}{p-1}} \rceil$ iterations and derivative's evaluations to produce an iterate x_{k_ϵ} such that $|\nabla_x^1 f_p(x_{k_\epsilon})| \leq \epsilon_1$ and $\lambda_{\min}(\nabla_x^2 f(x_{k_\epsilon})) \geq -\epsilon_2$.*

Proof. The proof is very similar to that of Theorem 3.3.11, this time taking a uniformly zero gradient but a minimal eigenvalue of the Hessian slowly converging to $-\epsilon_2$ from below. It is detailed in appendix. □

3.5 The effect of noise

We have mentioned in the introduction that OFFO algorithms like that presented above are interesting not only because of their remarkable theoretical properties covered in the previous sections, but also because they show remarkable insensitivity to noise[1]. This section is devoted to illustrating this statement while, at the same time, proposing a more detailed algorithm

[1] A similar behaviour was observed in [114] for first-order OFFO methods.

which exploits the freedom left in (3.2.5) (or (3.4.4)). Focusing on the OFFAR2 and MOFFAR2 algorithms (that is (M)OFFARp for $p = 2$), we rewrite (3.2.5) as

$$\sigma_k = \max(\vartheta\nu_k, \xi_k\mu_{1,k})$$

for some factor $\xi_k \in [\vartheta, 1]$ which we adaptively update as follows. We first define, for some power $\beta \in (0, 1]$, an initial gradient-norm "target" $t_0 = \frac{9}{10}\|g_k\|^\beta$ and an initial factor $\xi_0 = 1$. For $k > 0$, we then update t_k and ξ_k, according to the rules

$$\xi_k = \begin{cases} \max(\vartheta, \frac{1}{2}\xi_{k-1}) & \text{if} \quad \|g_k\| \leq t_{k-1}, \\ \frac{1}{2}(1 + \xi_{k-1}) & \text{if} \quad \|g_k\| > \max(t_{k-1}, \|g_{k-1}\|) \ \text{ and } \ \xi_{k-1} < 1, \\ \xi_{k-1} & \text{otherwise} \end{cases} \tag{3.5.1}$$

and

$$t_k = \begin{cases} \frac{9}{10}\|g_k\|^\beta & \text{if} \quad \|g_k\| \leq t_{k-1}, \\ t_{k-1} & \text{otherwise.} \end{cases} \tag{3.5.2}$$

Thus the factor ξ_k decreases when the gradient's norms converge to zero at a sufficiently fast rate (determined by the power β), while ξ_k increases to one if the gradient norms increase[2]. In addition, we choose the value of $\nu_0 = \sigma_0$ with the objective of getting $\|s_0\|$ of the order of unity[3], and set

$$\nu_0 = \max(\varsigma, 6\|g_0\|).$$

Finally, we use $\vartheta = 0.001$ and $\varsigma = 0.0001$.

In what follows, we compare the standard second-order regularization algorithms AR2[4] with three variants of the OFFAR2 and MOFFAR2 methods we just described. The first variant, (M)OFFAR2a , uses $\beta = 1$ in the definition of t_0 and (3.5.2), the second, (M)OFFAR2b, uses $\beta = 2/3$ and the third, (M)OFFAR2c, uses $\beta = 1/2$. The MAR2 algorithm is identical to AR2 except that it uses the more stringent second-order conditions (3.4.5)-(3.4.7) for the step computation. All on the algorithms were run[5] on the low dimensional instances of the problems[6] of the OPM collection [110, January 2022] listed with their dimension in Table 1 of Section 1. The specified algorithmic variant is iterated until either $\|\nabla_x^1 f(x_k)\|_2 \leq \epsilon_1$, or a maximum of 50000 iterations was reached, or evaluation of the gradient or Hessian returned an error.

Before considering the results, we recall an important comment made in [114]. Very few of the test functions satisfy AS.3 on the whole of $\mathbb{R}^n$. While this is usually not a problem when testing descent methods (because AS.3 may then be true in the level set determined by the starting point), this is no longer the case for significantly non-monotone methods like the ones tested here. As a consequence, it may (and does) happen that the gradient evaluation is attempted at a point where its value exceeds the Matlab overflow limit, causing the algorithm to fail on the problem.

We first consider the noiseless case, in which all algorithms were terminated as soon as an iterate x_k was found such that $\|\nabla_x^1 f(x_k)\| \leq 10^{-6}$. Figure 3.1 shows the corresponding

[2]The constants $\frac{1}{2}$ and $\frac{9}{10}$ were chosen somewhat arbitrarily, but we found these to work well in our tests and the algorithm to be quite insensitive to their precise values.

[3]Making the second ratio in the right-hand side of (3.3.13) equal to one.

[4]See [57, page 65] and MAR2 with $\eta_1 = 10^{-4}$, $\eta_2 = 0.95$, $\gamma_1 = 2 = 1/\gamma_2$,$\gamma_3 = 10^{20}$. $\sigma_{\min} = 10^{-4}$, $\theta_1 = 1.1$ and $\theta_2 = +\infty$.

[5]In Matlab on a Dell portable computer under Ubuntu 20.04 with sixteen cores and 64 GB of memory.

[6]From their standard starting point.

performance profile [86] for the algorithms, but we also follow [179] and consider the derived "global" measure $\pi_{\texttt{algo}}$ to be $\frac{1}{50}$ of the area below the curve corresponding to `algo` in the performance profile, for abscissas in the interval $[1, 50]$. The larger this area and closer $\pi_{\texttt{algo}}$ to one, the closer the curve to the right and top borders of the plot and the better the global performance. In addition, $\rho_{\texttt{algo}}$ denotes the percentage of successful runs taken on all problems. Table 3.1 presents the values of the $\pi_{\texttt{algo}}$ and $\rho_{\texttt{algo}}$ statistics. Failure[7] of a given (M)OFFAR2 variant essentially occurs on ill-conditioned problems.

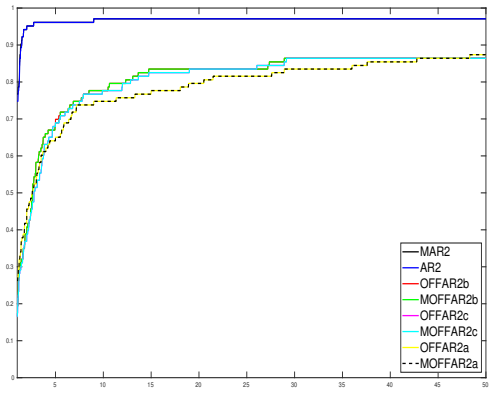

Figure 3.1: Performance profile for deterministic OFFO algorithms on noiseless OPM problems. We report on the vertical axis the proportion of problems for which the number of iterations of each algorithm is at most a fraction (given by the horizontal axis) of the smallest among all algorithms (see [86]).

	AR2	OFFAR2a	OFFAR2b	OFFAR2c	MAR2	MOFFAR2a	MOFFAR2b	MOFFAR2c
$\pi_{\texttt{algo}}$	0.97	0.79	0.81	0.81	0.97	0.79	0.81	0.81
$\rho_{\texttt{algo}}$	95.8	88.24	87.39	87.39	95.80	88.24	87.29	87.39

Table 3.1: Performance and reliability statistics on OPM problems without noise

Notice that the second-order variants (M*) are, in these statistics, undistinguishable from their first-order counterparts, although (very minor) differences in performance occur for some nonconvex examples. For the AR2 and the OFFAR2 variants, this seems to vindicate the folklore observation that second-order optimality is most often obtained by first-order algorithms, although no guarantee can be provided.

Obviously, if we were to consider noiseless problems only, our results provide little motivation (beyond theoretical curiosity) to consider the (M)OFFAR2 algorithms. This is not unexpected since (M)AR2 does exploit objective function values and we know that identical global convergence orders may not directly translate into similar practical behaviour. We nevertheless note that the global behaviour of the (M)OFFAR2 algorithms is far from poor on reasonably well-conditioned cases.

[7] On biggs5, brownbs, cliff, chebyqad, cosine, curly10, himm32, eg2s, genhumps, gulf, meyer3, osbornea, osborneb, powellbs, scurly10, scosine, vibrbeam.

However, as we have announced, the situation is quite different when considering the noisy case. When noise is present and because the value of $\mu_{1,k}$ (which is crucial to the performance of the OFFAR2 algorithm) directly depends on the now noisy gradient, we have modified (3.2.4) slightly in an attempt to attenuate the impact of noise. We then use the smoothed update

$$\delta_k = \frac{9}{10}\delta_{k-1} + \frac{1}{10}\left(\frac{2\|g_k\|}{\|s_{k-1}\|^2}\right), \quad \mu_{1,k} = \delta_k - \theta_1\sigma_{k-1},$$

where $\delta_0 = \max(\varsigma, \|g_0\|)$, instead of (3.2.4). Similarly we use a smoothed version of the gradient's norm

$$\tau_k = \frac{9}{10}\tau_{k-1} + \frac{1}{10}\|g_k\|$$

(with $\tau_{-1} = \|g_0\|$) instead of $\|g_k\|$ in (3.5.1) and (3.5.2).

Table 3.2 shows the reliability score $\rho_{\texttt{algo}}$ of the three algorithms averaged over 10 runs for four increasing percentages of relative random Gaussian noise added to the derivatives (and the objective function for AR2). By "relative random Gaussian noise", we mean that the i-th entry of the noisy gradient is set to $[g_k]_i = [\nabla_x(x_k)]_i(1+\phi\xi_i)$ where ξ_i is drawn from $\mathcal{N}(0,1))$ and ϕ is the relative noise level, the definition for the noisy Hessian (and noisy function value when relevant) being similar. The tolerance ϵ_1 was set to 10^{-3} for these runs.

ϕ	0.05	0.15	0.25	0.50
AR2	43.70	31.26	26.30	10.00
OFFAR2a	86.05	82.10	79.83	68.99
OFFAR2b	86.89	79.08	73.45	62.18
OFFAR2c	87.90	78.49	73.53	61.18
MAR2	44.20	31.51	25.38	9.08
MOFFAR2a	84.03	81.60	79.92	68.24
MOFFAR2b	85.38	77.39	71.93	61.43
MOFFAR2c	85.63	77.06	71.43	60.92

Table 3.2: Reliability statistics $\rho_{\texttt{algo}}$ for 5%, 15%, 25% and 50% relative random Gaussian noise (averaged on 10 runs)

As announced, the reliability of (M)AR2 decreases sharply when the noise level increases, essentially because the decision to accept or reject an iteration, which is based on function values, is strongly affected by noise. By contrast, the reliability of the (M)OFFAR2 variants decreases remarkably slowly. That the (M)OFFAR2 variants are able to solve more that 60% of the problems with 50% relative Gaussian noise is quite encouraging.

3.6 Discussions

We have presented an adaptive regularization algorithm for nonconvex unconstrained minimization where the objective function is never calculated and which has, for a given degree of used derivatives, the best-known worst-case complexity order, not only among OFFO methods, but also among all known optimization algorithms seeking first-order critical points. In particular, the algorithm using gradients and Hessians requires at most $\mathcal{O}(\epsilon_1^{-(p+\beta)/(p-1+\beta)})$ iterations to produce an iterate such that $\|\nabla_x^1 f(x_k)\| \leq \epsilon_1$, and at most $\mathcal{O}(\epsilon_2^{-(p+\beta)/(p-2+\beta)})$

iterations to additionally ensure that $\lambda_{\min}(\nabla_x^2 f(x_k)) \geq -\epsilon_2$. Moreover, all stated complexity bounds are sharp.

These results may be extended in different ways, which we have not included in our development to avoid too much generality and reduce the notational burden. The first is to allow errors in derivatives of orders 2 to p. If we denote by $\overline{\nabla_x^i f}$ the approximation of $\nabla_x^i f$, it is easily seen in the proof of Lemma 3.3.3 that the argument remains valid as long as, for some $\kappa_D \geq 0$,

$$\|\overline{\nabla_x^i f}(x_k) - \nabla_x^i f(x_k)\| \leq \kappa_D \|s_k\|^{p+1-i}. \tag{3.6.1}$$

Since, for first-order analysis, the accuracy of derivatives of degree larger than one only occurs in this lemma, we conclude that Theorem 3.3.10 still hold if (3.6.1) holds.

Further likely generalizations include imposing convex constraints on the variables [57, Chapter 6]. An extension to guarantee third-order optimality conditions (in the case where third derivatives are available) may also be possible along the lines discussed in [57, Chapter 4].

We also observe that the lower bound $\sigma_k \geq \max(\mu_{1,k}, \mu_{2,k})$ has not yet, to the best of our knowledge, been used in implementations of the standard AR2 algorithm. Its incorporation in this method is clearly possible and worth investigating.

More extensive numerical experiments involving both regularization and trust-region methods using a variety of approximate derivatives are the subject of future work and will be reported separately.

Appendices

3.A Proof of Lemma 3.3.5

Proof. The proof closely follows the approach presented in [216, Lecture VI, Lemma 5]. However, minor adjustments are introduced to accommodate the scenario where $n-1+\beta$ is not an integer, as is the case when $\beta < 1$. Let us consider first

$$|x| > \kappa_m \tag{3.A.1}$$

Where κ_m is defined in (3.3.10). Since $a_n x^{n-1+\beta} = -a_{n-1}x^{n-1} - \cdots - a_1 x - a_0$. Taking the absolute value in the last equality and using κ_m definition (3.3.10) and (3.A.1),

$$1 \leq \sum_{i=1}^{n} \frac{|a_{n-i}||x|^{n-i}}{|a_n||x|^{n-1+\beta}} \leq \sum_{i=1}^{n} \frac{|a_{n-i}|}{|a_n||x|^{i-1+\beta}} \leq \sum_{i=1}^{n} \left(\frac{\kappa_m}{|x|}\right)^{i-1+\beta}$$

$$1 \leq \left(\frac{\kappa_m}{|x|}\right)^{-1+\beta} \frac{\frac{\kappa_m}{|x|}}{1-\frac{\kappa_m}{|x|}} \leq \frac{\left(\frac{\kappa_m}{|x|}\right)^{\beta}}{1-\frac{\kappa_m}{|x|}}.$$

Rearranging the last inequality, (3.A.1) and the fact that $\beta \in (0,1]$, we obtain that

$$1 \leq \left(\frac{\kappa_m}{|x|}\right)^{\beta} + \frac{\kappa_m}{|x|} \leq 2\left(\frac{\kappa_m}{|x|}\right)^{\beta}$$

Using the last inequality to bound $|x|$ and use again the fact that $\beta \in (0,1]$

$$|x| \leq 2^{\frac{1}{\beta}}\kappa_m \leq 2\kappa_m \tag{3.A.2}$$

Combining (3.A.2) and the fact that we supposed at first that $|x|$ verifies (3.A.1), we derive (3.3.11). □

3.B Proof of Lemmas and Theorem from Section (3.4)

3.B.1 Proof of Lemma 3.4.3

Proof. Suppose now that $p > 1$. Again (3.4.5) gives that

$$\begin{aligned}\frac{\sigma_k}{(p+\beta)!}\|s_k\|^{p+\beta} &\leq -g_k^T s_k - \frac{1}{2}s_k^T H_k s_k - \sum_{i=3}^{p}\frac{1}{i!}\nabla_x^i f(x_k)[s_k]^i \\ &\leq \|g_k\|\|s_k\| + \frac{1}{2}\max\left(0, -\lambda_{\min}(H_k)\right)\|s_k\|^2 + \sum_{i=3}^{p}\frac{\kappa_{\rm high}}{i!}\|s_k\|^i.\end{aligned}$$

Where we used the definition of $\lambda_{\min}(H_k)$ and AS.3bis to obtain the last inequality.

Applying now Lemma 3.3.5 with $x = \|s_k\|$, $n = p+1$, $a_0 = 0$, $a_1 = \|g_k\|$, $a_2 = \frac{1}{2}[-\lambda_{\min}(H_k)]_+$, $\beta = \beta$, $a_i = \kappa_{\rm high}/i!$ $i \in \{2,\ldots,p\}$ and $a_{p+1} = \sigma_k/(p+\beta)!$, we know from (3.2.6) that the equation $\sum_{i=0}^{n} a_i x^i = 0$ admits at least one strictly positive root, and we may thus derive that

$$\begin{aligned}\|s_k\| &\leq 2\left(\frac{(p+\beta)!\|g_k\|}{\sigma_k}\right)^{\frac{1}{p-1+\beta}} + 2\left(\frac{(p+\beta)![-\lambda_{\min}(H_k)]_+}{2\sigma_k}\right)^{\frac{1}{p-2+\beta}} + 2\sum_{i=3}^{p}\left[\frac{\kappa_{\rm high}(p+\beta)!}{i!\,\sigma_k}\right]^{\frac{1}{p-i+\beta}} \\ &\leq 2\left(\frac{(p+\beta)!\|g_k\|}{\sigma_k}\right)^{\frac{1}{p-1+\beta}} + 2\left(\frac{(p+\beta)![-\lambda_{\min}(H_k)]_+}{2\sigma_k}\right)^{\frac{1}{p-2+\beta}} + 2\sum_{i=3}^{p}\left[\frac{\kappa_{\rm high}(p+\beta)!}{i!\,\vartheta\nu_k}\right]^{\frac{1}{p-i+\beta}} \\ &\leq 2\left(\frac{(p+\beta)!\|g_k\|}{\sigma_k}\right)^{\frac{1}{p-1+\beta}} + 2\left(\frac{(p+\beta)![-\lambda_{\min}(H_k)]_+}{2\sigma_k}\right)^{\frac{1}{p-2+\beta}} + 2\sum_{i=3}^{p}\left[\frac{\kappa_{\rm high}(p+\beta)!}{i!\,\vartheta\nu_0}\right]^{\frac{1}{p-i+\beta}},\end{aligned}$$

and (3.4.18) holds with (3.4.19). □

3.B.2 Proof of Lemma 3.4.4

if $k_1 = 1$, we have that

$$\nu_1 = \sigma_0 + \sigma_0\|s_0\|^{p+\beta}.$$

Using Lemma 3.4.3 to bound $\|s_0\|^{p+\beta}$, we derive that

$$\nu_1 \leq \sigma_0 + \sigma_0\left(2\eta_{bis} + 2\left(\frac{(p+\beta)!\|g_0\|}{\sigma_0}\right)^{\frac{1}{p-1+\beta}} + 2\left((p+\beta)!\frac{[-\lambda_{\min}(H_0)]_+}{2\sigma_0}\right)^{\frac{1}{p-2+\beta}}\right)^{p+\beta}. \tag{3.B.1}$$

The last inequality gives the first part in (3.4.21). Suppose now that $k_1 \geq 2$. Successively using (3.4.9), Lemma 3.4.3 and the fact $(x+y+z)^{p+\beta} \leq 3^{p+\beta-1}(x^{p+\beta}+y^{p+\beta}+z^{p+\beta})$ for

non-negative x, y, z with the latter holding by convexity, we obtain that

$$\begin{aligned}\nu_{k_1} &\leq \nu_{k_1-1} + \nu_{k_1-1}\|s_{k_1-1}\|^{p+\beta} \\ &\leq \nu_{k_1-1} + \nu_{k_1-1}\left(2\eta_{bis} + 2\left(\frac{(p+\beta)!\|g_{k_1-1}\|}{\sigma_{k_1-1}}\right)^{\frac{1}{p-1+\beta}} + 2\left((p+\beta)!\frac{[-\lambda_{\min}(H_{k_1-1})]_+}{2\sigma_{k_1-1}}\right)^{\frac{1}{p-2+\beta}}\right)^{p+\beta} \\ &\leq A + 3^{p+\beta-1}2^{p+\beta}\nu_{k_1-1}\left((p+\beta)!\frac{[-\lambda_{\min}(H_{k_1-1})]_+}{2\sigma_{k_1-1}}\right)^{\frac{p+\beta}{p-2+\beta}}, \qquad (3.B.2)\end{aligned}$$

where A is defined as

$$A \stackrel{\text{def}}{=} \nu_{k_1-1} + 3^{p+\beta-1}2^{p+\beta}\eta_{bis}^{p+\beta}\nu_{k_1-1} + 3^{p+\beta-1}2^{p+\beta}\nu_{k_1-1}\left(\frac{(p+\beta)!\|g_{k_1-1}\|}{\sigma_{k_1-1}}\right)^{\frac{p+\beta}{p-2+\beta}}. \qquad (3.B.3)$$

Up to a $\frac{2^{p-1+\beta}}{3^{p-1+\beta}}$ in the second and third term of A and with η_{bis} instead of η, an upper-bound has been derived on A in in the proof of Lemma 3.3.7 (see inequality (3.3.17)). Hence there exists $\kappa_2 > 0$ that can be deduced from (3.3.16) accordingly such that

$$A \leq \frac{2\kappa_2 L_p}{\vartheta}. \qquad (3.B.4)$$

where

$$\kappa_2 = 1 + 2^{p+\beta}3^{p-1+\beta}\eta_{bis}^{p+\beta} + 2^{p+\beta}3^{p-1+\beta}\left[\frac{(p+\beta)}{\vartheta}\left((1+\theta_1)\frac{L_p}{\sigma_0} + \theta_1\right)\right]^{\frac{p+\beta}{p-1+\beta}}. \qquad (3.B.5)$$

We will focus in this proof only on the $\nu_{k_1-1}\left(\frac{[-\lambda_{\min}(H_{k_1-1})]_+}{2\sigma_{k_1-1}}\right)^{\frac{p+\beta}{p-2+\beta}}$ term in (3.B.2). Using the fact that $\sigma_{k_1-1} \geq \vartheta\nu_{k_1-1}$, (3.4.11) and that $\sigma_{k_1-2} \leq \nu_{k_1-2}$, we obtain

$$\begin{aligned}\nu_{k_1-1}\left(\frac{[-\lambda_{\min}(H_{k_1-1})]_+}{2\sigma_{k_1-1}}\right)^{\frac{p+\beta}{p-2+\beta}} &\leq \nu_{k_1-1}\left(\frac{[-\lambda_{\min}(H_{k_1-1})]_+}{2\vartheta\nu_{k_1-1}}\right)^{\frac{p+\beta}{p-2+\beta}} \\ &\leq \nu_{k_1-1}^{\frac{-2}{p-2+\beta}}\left(\frac{L_p + \theta_2\sigma_{k_1-2}}{2(p-2+\beta)!\vartheta}\|s_{k_1-2}\|^{p-2+\beta}\right)^{\frac{p+\beta}{p-2+\beta}} \\ &\leq \left(\frac{\frac{L_p}{\nu_{k_1-2}} + \theta_2}{2\vartheta(p-2+\beta)!}\right)^{\frac{p+\beta}{p-2+\beta}} (\nu_{k_1-1})^{\frac{-2}{p-2+\beta}}\nu_{k_1-2}^{\frac{p+\beta}{p-2+\beta}}\|s_{k_1-2}\|^{p+\beta}\end{aligned}$$

Using now that ν_k is a non-decreasing sequence and that $\nu_{k_1-2}\|s_{k_1-2}\|^{p+\beta} = \nu_{k_1-1} - \nu_{k_1-2} \leq \nu_{k_1-1}$ in the last inequality, we derive

$$\begin{aligned}\nu_{k_1-1}\left(\frac{[-\lambda_{\min}(H_{k_1-1})]_+}{2\sigma_{k_1-1}}\right)^{\frac{p+\beta}{p-2+\beta}} &\leq \left(\frac{\frac{L_p}{\sigma_0} + \theta_2}{2\vartheta(p-2+\beta)!}\right)^{\frac{p+\beta}{p-2+\beta}} \nu_{k_1-2}\|s_{k_1-2}\|^{p+\beta} \\ &\leq \left(\frac{\frac{L_p}{\sigma_0} + \theta_2}{2\vartheta(p-2+\beta)!}\right)^{\frac{p+\beta}{p-2+\beta}} \nu_{k_1-1}.\end{aligned}$$

Using now the fact that $\nu_{k_1-1} \leq \frac{2L_p}{\vartheta}$ in the last inequality, we obtain that

$$3^{p-1+\beta}2^{p+\beta}(p+\beta)!^{\frac{p+\beta}{p-2+\beta}}\nu_{k_1-1}\left(\frac{[-\lambda_{\min}(H_{k_1-1})]_+}{2\sigma_{k_1-1}}\right)^{\frac{p+\beta}{p-2+\beta}}$$
$$\leq 2^{p+\beta}3^{p-1+\beta}((p-1+\beta)(p+\beta))^{\frac{p+\beta}{p-2+\beta}}\left(\frac{\frac{L_p}{\sigma_0}+\theta_2}{2\vartheta}\right)^{\frac{p+\beta}{p-2+\beta}}\frac{2L_p}{\vartheta}. \tag{3.B.6}$$

Combining the last inequality with (3.B.4) in (3.B.2) gives the second term in (3.4.21).

3.B.3 Proof of Theorem 3.4.6

Proof. The proof of this result closely follows that of Theorem 3.3.11. First select $\vartheta = 1$, some $\sigma_0 = \nu_0 > 0$ and define, for all $k \in \{0, \ldots, k_\epsilon\}$,

$$\omega_k = \epsilon_2 \frac{k_\epsilon - k}{k_\epsilon} \in [0, \epsilon_2] \tag{3.B.7}$$

and

$$g_k = 0, \quad H_k = -(\epsilon_2 + \omega_k) \quad \text{and} \quad D_{i,k} = 0, \quad (i = 3, \ldots, p), \tag{3.B.8}$$

so that

$$|H_k| \in [\epsilon_2, 2\epsilon_2] \subset [0, 2] \ \text{ for all } \ k \in \{0, \ldots, k_\epsilon\}. \tag{3.B.9}$$

We then set, for all $k \in \{0, \ldots, k_\epsilon\}$,

$$s_k = \left(\frac{p!|H_k|}{\sigma_k}\right)^{\frac{1}{p-1}}. \tag{3.B.10}$$

Since $g_k = 0$, $\mu_{1,k} < 0$ for $k > 0$, $\mu_{1,k}$ is irrelevant in (3.4.4). Computing now $\mu_{2,k}$ by using (3.B.10)

$$\mu_{2,k} = \frac{(p-1)!|H_k|}{\|s_{k-1}\|^{p-1}} - \theta_2\sigma_{k-1} = \left(\frac{|H_k|}{p|H_{k-1}|} - \theta_2\right)\sigma_{k-1} < 0$$

where the last inequality results from (3.B.9) and $p \geq 2$. Hence, as $\vartheta = 1$, $\sigma_k = \nu_k$. So that

$$\sigma_k \stackrel{\text{def}}{=} \sigma_0 + \sum_{j=0}^{k-1}\sigma_j|s_j|^{p+1} \tag{3.B.11}$$
$$= \sigma_0 + \sum_{j=0}^{k-1}\sigma_j\left(\frac{p!|H_j|}{\sigma_j}\right)^{\frac{p+1}{p-1}} = \sigma_0 + (p!)^{\frac{p+1}{p-1}}\sum_{j=0}^{k-1}\frac{(\epsilon_2+\omega_j)^{\frac{p+1}{p-1}}}{\sigma_j^{\frac{2}{p-1}}}$$
$$\leq \sigma_0 + \left(\frac{(2p!)^{p+1}}{\sigma_0^2}\right)^{\frac{1}{p-1}}\sum_{j=0}^{k-1}\epsilon_2^{\frac{p+1}{p-1}} \leq \sigma_0 + \left(\frac{(2p!)^{p+1}}{\sigma_0^2}\right)^{\frac{1}{p-1}}k_\epsilon\epsilon_2^{\frac{p+1}{p-1}} \leq \sigma_0 + 2\left(\frac{(2p!)^{p+1}}{\sigma_0^2}\right)^{\frac{1}{p-1}}$$
$$\stackrel{\text{def}}{=} \sigma_{\max},$$

where we successively used (3.B.10), (3.B.8), (3.B.7) and the definition of k_ϵ. We finally

set

$$f_0 = 2^{\frac{p+1}{p-1}} \left(\frac{p!}{\sigma_0}\right)^{\frac{2}{p-1}} \text{ and } f_{k+1} \stackrel{\text{def}}{=} f_k + \tfrac{1}{2} H_k s_k^2 + \sum_{i=2}^{p} \frac{1}{i!} D_{i,k}[s_k]^i = f_k - \tfrac{1}{2} \left(\frac{p!}{\sigma_k}\right)^{\frac{2}{p-1}} (\epsilon_2 + \omega_k)^{\frac{p+1}{p-1}},$$

yielding, using (3.3.32) and the definition of k_ϵ, that

$$f_0 - f_{k_\epsilon} = \frac{1}{2} \sum_{k=0}^{k_\epsilon - 1} \left(\frac{p!}{\sigma_k}\right)^{\frac{2}{p-1}} (\epsilon_2 + \omega_k)^{\frac{p+1}{p-1}} \leq 2^{\frac{2}{p-1}} \left(\frac{p!}{\sigma_0}\right)^{\frac{2}{p-1}} k_\epsilon \epsilon_2^{\frac{p+1}{p}} \leq 2^{\frac{p+1}{p-1}} \left(\frac{p!}{\sigma_0}\right)^{\frac{2}{p-1}} = f_0.$$

As a consequence

$$f_k \in [0, f_0] \text{ for all } k \in \{0, \ldots, k_\epsilon\}. \tag{3.B.12}$$

Observe that (3.B.10) satisfies (3.4.5) (for the model (1.3.18)), (3.4.6) for $\theta_1 = 1$ and (3.4.7) for $\theta_2 = 1$. Moreover (3.B.11) is the same as (3.4.9)-(3.4.4). Hence the sequence $\{x_k\}$ generated by

$$x_0 = 0 \text{ and } x_{k+1} = x_k + s_k$$

may be viewed as produced by the modified OFFARp algorithm given (3.B.8). Observe also that, for

$$T_{k,p}(s) = f_k + g_k s + \tfrac{1}{2} H_k s_k^2 + \sum_{i=3}^{p} \frac{1}{i!} D_{i,k} s_k^i,$$

one has that

$$|f_{k+1} - T_{k,p}(s_k)| = 0 \leq \frac{\sigma_{\max}}{p!} |s_k|^{p+1}, \tag{3.B.13}$$

$$|g_{k+1} - \nabla_s^1 T_{k,p}(s_k)| = |H_k s_k| = \frac{1}{p!} |s_k|^p, \tag{3.B.14}$$

and

$$\begin{aligned} |H_{k+1} - \nabla_s^2 T_{k,p}(s_k)| &= |H_{k+1} - H_k| \leq |\omega_k - \omega_{k+1}| \\ &= \frac{\epsilon_2}{k_\epsilon} \leq \epsilon_2^{\frac{2p}{p-1}} \leq \frac{\sigma_{\max}}{\sigma_k} (\epsilon_2 + \omega_k) = \frac{\sigma_{\max}}{p!} |s_k|^{p-1} \end{aligned} \tag{3.B.15}$$

(we used $k_\epsilon \leq \epsilon_2^{-\frac{p+1}{p-1}} + 1$ and $\epsilon_2 \leq 1$), while, if $p > 2$,

$$|D_{i,k+1} - \nabla_s^i T_{k,p}(s_k)| = 0 \leq \frac{\sigma_{\max}}{p!} |s_k|^{p+1-i} \tag{3.B.16}$$

for $i = 3, \ldots, p$. In view of (3.B.9), (3.B.12) and (3.B.13)-(3.B.16), we may then apply classical Hermite interpolation to the data given by $\{(x_k, f_k, g_k, H_k, D_{3,k}, \ldots, D_{p,k})\}_{k=0}^{k_\epsilon}$ (see [57, Theorem A.9.2] with $\kappa_f = \max(2, f_0, \sigma_{\max}/p!)$, for instance) and deduce that there exists a p times continuously differentiable piecewise polynomial function f_p satisfying AS.1–AS.4 and such that, for $k \in \{0, \ldots, k_\epsilon\}$,

$$f_k = f_p(x_k), \quad g_k = \nabla_x^1 f_p(x_k), \quad H_k = \nabla_x^2 f_p(x_k) \text{ and } D_{i,k} = \nabla_x^i f_p(x_k), \quad (i = 3, \ldots, p).$$

The sequence $\{x_k\}$ may thus be interpreted as being produced by the OFFARp algorithm applied to f_p starting from $x_0 = 0$. The desired conclusion then follows by observing that,

from (3.B.7) and (3.B.8), $g_k = 0 < \epsilon_1$ for all k while

$$\lambda_{\min}[H_k] = H_k < -\epsilon_2 \;\text{ for }\; k \in \{0, \ldots, k_\epsilon - 1\} \quad \text{and} \quad \lambda_{\min}[H_{k_\epsilon}] = H_{k_\epsilon} = -\epsilon_2.$$

□

3.C Divergence of the simplified OFFAR2 algorithm using (3.3.37)

We now prove that a modified simplified OFFAR2 algorithm using (3.3.37) instead of (3.2.5) may diverge. We again proceed by constructing an example, now in $\mathbb{R}^2$, where this undesirable behaviour occur. To this aim, we first define sequences of function, gradient and Hessian values which we will subsequently interpolate to produce the function itself. For $k \geq 0$ and some constants $H \geq 1$ and $\theta_1 \geq 1$, define

$$f_k = 1, \quad g_k = -\begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad H_k = \begin{pmatrix} 0 & 0 \\ 0 & H \end{pmatrix} \tag{3.C.1}$$

and

$$\sigma_k = \frac{2(H+1)}{\sqrt{1+(H+1)^2}} < 2. \tag{3.C.2}$$

We may now seek the step s_k which minimizes the regularized model built from these values for given k, that is satisfying

$$g_k + H_k s_k + \tfrac{1}{2}\sigma_k \|s_k\| s_k = 0.$$

Setting $[s_k]_1 = 1$, the first equation of this system gives that

$$\frac{\sigma_k \|s_k\|}{2} = \frac{\sigma_k \|s_k\|}{2}[s_k]_1 = -[g_k]_1 = 1, \tag{3.C.3}$$

which we may substitute in the second equation to obtain that

$$(H+1)[s_k]_2 = -[g_k]_2 = 1.$$

This gives that

$$s_k = \begin{pmatrix} 1 \\ \dfrac{1}{H+1} \end{pmatrix} \quad \text{and thus} \quad \|s_k\| = \sqrt{1 + \frac{1}{(H+1)^2}}, \tag{3.C.4}$$

which is consistent with (3.C.2) and (3.C.3), and we may construct a sequence $\{x_k\}$ by setting

$$x_0 = 0 \quad \text{and} \quad x_{k+1} = x_k + s_k \quad (k \geq 0).$$

Note that both $\{[x_k]_1\}$ and $\{[x_k]_2\}$ are strictly increasing. We also verify, using (3.2.4), (3.C.1), (3.C.3) and the bound $\theta_1 \geq 1$, that

$$\begin{aligned}\mu_{1,k} &= \frac{2\|g_k\|}{\|s_{k-1}\|^2} - \theta_1\sigma_{k-1} = \frac{2\sqrt{2}(H+1)^2}{1+(H+1)^2} - \theta_1\sigma_{k-1} \\ &= \frac{\sigma_{k-1}^2}{\sqrt{2}} - \theta_1\sigma_{k-1} = \sigma_{k-1}\left(\frac{\sigma_{k-1}}{\sqrt{2}} - \theta_1\right) < \sigma_{k-1}(\sqrt{2}-1) < \sigma_{k-1},\end{aligned}$$

where we used the fact that $\sigma_k < 2$ and $\theta_1 \geq 1$ to obtain the last inequality. As a consequence, (3.3.37) gives that $\sigma_k = \sigma_{k-1}$, in accordance with (3.C.2) and the process we have just described may be interpreted as a divergent run of the simplified OFFAR2 algorithm (note that $f_k = 1$ and $\|g_k\| = \sqrt{2}$ for all $k \geq 0$), provided (3.C.1) can be interpolated by a function from $\mathbb{R}^2$ to $\mathbb{R}$ satisfying AS.1–AS.4. This is achieved by defining $f_{k,1} = f_{k,2} = \frac{1}{2}$ for $k \geq 0$ and using unidimensional Hermite interpolation on the two datasets

$$\{([x_k]_1, f_{k,1}, [g_k]_1, [H_k]_{1,1})\}_{k\geq 0}, \quad \text{and} \quad \{([x_k]_2, f_{k,2}, [g_k]_2, [H_k]_{2,2})\}_{k\geq 0}.$$

Thus we once more invoke [57, Theorem A.9.2] with $\kappa_f = (H+1)^2$ for each dataset, which is possible because

$$\begin{aligned}|f_{k+1,1} - T_{k,1,2}([s_k]_1)| &= |[g_k]_1[s_k]_1| = 1 = |[s_k]_1|^3, \\ |[g_{k+1]_1} - \nabla_s^1 T_{k,1,2}([s_k]_1)| &= |[g_{k+1}]_1 - [g_k]_1| = 0 < 1 = |[s_k]_1|^2, \\ |[H_{k+1}]_{1,1} - \nabla_s^2 T_{k,1,2}([s_k]_1)| &= |0-0| = 0 < 1 = |[s_k]_1|,\end{aligned}$$

and

$$|f_{k+1,2} - T_{k,2,2}([s_k]_2)| = \left|-\frac{1}{H+1} + \frac{H}{2(H+1)^2}\right| \leq \frac{1}{2(H+1)} < (H+1)^2|[s_k]_2|^3,$$

$$|[g_{k+1}]_2 - \nabla_s^1 T_{k,2,2}([s_k]_2)| = |[H_k]_{2,2}[s_k]_2| = \frac{H}{H+1} < (H+1)^2|[s_k]_2|^2,$$

$$|[H_{k+1}]_{2,2} - \nabla_s^2 T_{k,2,2}([s_k]_2)| = |[H_{k+1}]_{2,2} - [H_k]_{2,2}| = 0 < \frac{1}{H+1} = |[s_k]_2|,$$

where we have defined

$$T_{k,j,2}(s) = f_{k,j} + [g_k]_j s + \tfrac{1}{2}[H_k]_{j,j}[s_k]_j^2, \quad (j = 1, 2).$$

We therefore deduce that there exist twice continuously differentiable piecewise polynomial functions $f_1(x_1)$ and $f_2(x_2)$ satisfying AS.1–AS.4 such that

$$f(x) = f_1(x_1) + f_2(x_2)$$

is also twice continuously differentiable, satisfies AS.1–AS.4 and interpolates (3.C.1). This completes the argument.

Chapter 4

Yet Another Fast Variant of Newton's Method

Chapter Abstract

A class of second-order algorithms is proposed for minimizing smooth nonconvex functions that alternates between regularized Newton and negative curvature steps in an iteration-dependent subspace. In most cases, the Hessian matrix is regularized with the square root of the current gradient and an additional term taking moderate negative curvature into account, a negative curvature step being taken only exceptionally. Practical variants have been detailed where the subspaces are chosen to be the full space, or Krylov subspaces. In the first case, the proposed method only requires the solution of a single linear system at nearly all iterations. We establish that at most $\mathcal{O}\left(|\log\epsilon|\,\epsilon^{-3/2}\right)$ evaluations of the problem's objective function and derivatives are needed for algorithms in the new class to obtain an ϵ-approximate first-order minimizer, and at most $\mathcal{O}\left(|\log\epsilon|\,\epsilon^{-3}\right)$ to obtain a second-order one. Encouraging initial numerical experiments with two full-space and two Krylov-subspaces variants are finally presented.

Reference: This chapter is based upon the work submitted for publication [117].

4.1 Introduction

The objective of this chapter is to extend the new regularization technique proposed by [156, 82] in the convex case. At an iterate x_k, the step s_k is computed as

$$s = -(\nabla_x^2 f(x_k) + \lambda_k I_n)^{-1} \nabla_x^1 f(x_k) \tag{4.1}$$

where $\lambda_k \sim \sqrt{\|\nabla_x^1 f(x_k)\|}$. This new approach exhibits the best complexity rate of second-order methods for convex optimization and retains the local superlinear convergence of standard Newton method, while showing remarkable numerical promise [156]. Devising an algorithm for nonconvex functions that can use similar ideas whenever possible appears as a natural extension.

In the nonconvex case, the Hessian may be indefinite and it is well-known that negative curvature can be exploited to ensure progress towards second-order points. Mixing gradient-related (possibly Newton) and negative curvature directions has long been considered and can be traced back to [153], which initiated a line of work using curvilinear search to find a

step combining both types of directions. The length of the step is typically tuned using an Armijo-like condition [153, 157]. Improvements were subsequently proposed by incorporating the curvilinear step in a nonmonotone algorithm [95], allowing the resolution of large-scale problems [149] or by choosing between the two steps based on model decrease [103]. Alternatively, negative curvature has also been used to regularize the Hessian matrix, yielding the famous Goldfeld-Quandt-Trotter (GQT) method [100]. Unfortunately, this method also involves more complex computation to find the step and has the same global convergence rate as first-order algorithms [197]. The negative curvature regularization was also the subject of the more recent paper [26], in which various Newton steps are tried at each iteration in order to ensure the optimal ARC [50, 51] global rate of convergence.

One may then wonder if it is possible to devise an adaptive second-order method using a single explicitly regularized Newton step when possible and a negative curvature direction only when necessary, with a near-optimal complexity rate. The objective of this paper is to show that it is indeed possible (and efficient). To this aim, we propose a fast Newton's method that exploits negative curvature for nonconvex optimization problems and generalizes the method proposed in [156, 82] to the nonconvex case. The new algorithm automatically adjusts the regularization parameter (without knowledge of the Hessian's Lipschitz constant). The method either uses an appropriately regularized Newton step taking the smallest negative eigenvalue of the Hessian also into account or simply follows the negative curvature otherwise. It first attempts a step along a direction regularized by the square root of the gradient only, as in the convex setting [82, 156]. In that sense, it is inspired by the "convex until proved guilty" strategy advocated by [44]. If this attempt fails, it obtains negative curvature information of the Hessian, which is then used either for regularization or to define a step along a negative-curvature direction. In what follows, all these operations are carried out in a specific, iteration-dependent subspace, whose choice leads to different algorithmic variants. We prove that these methods require at most $\mathcal{O}\left(|\log\epsilon|\,\epsilon^{-3/2}\right)$ iterations and evaluations of the problem data to obtain an ϵ-approximate first-order critical point, which is very close to the optimal convergence rate of second-order methods for Lipschitz Hessian functions [52]. We also introduce an further algorithmic variant which is guaranteed to find a second-order critical point in at most $\mathcal{O}\left(|\log\epsilon|\,\epsilon^{-3}\right)$ iterations.

The Chapter is organized as follows. Section 4.2 describes the general algorithmic framework and compares it with recent work on second-order methods. Section 4.3 states our assumptions and derives a bound on its worst-case complexity for finding first-order critical points. Section 4.2 presents the second-order algorithmic variant and states its complexity, the corresponding analysis being detailed in appendix. Section 4.5 then discusses some choices of the iteration-dependent subspace, including Krylov spaces. Section 4.6 finally illustrates the numerical behavior of the proposed methods. Conclusions are drawn in Section 4.7.

4.2 Adaptive Newton with Negative Curvature

We consider the problem of finding approximate first-order critical points of the smooth unconstrained nonconvex optimization problem (P) and discuss our algorithm called AN2C (for Adaptive Newton with Negative Curvature) on the next page. The algorithm, whose purpose is to compute first-order critical points, is presented in the framework of adaptive regularization methods [29, 51] [57, Section 3.3] and proceeds as follows, using two subroutines RegStep and NewtonEigenStep.

Algorithm 4.2.1: Adaptive Newton with Negative Curvature (AN2C)

Step 0: Initialization An initial point $x_0 \in \mathbb{R}^n$, a regularization parameter $\sigma_0 > 0$ and a gradient accuracy threshold $\epsilon \in (0, 1]$ are given, as well as the parameters

$$\sigma_{\min} > 0,\ \kappa_C, \kappa_\theta > 0,\ \kappa_a, \kappa_b \geq 1,\ \varsigma_1 \in (0, 1),\ \varsigma_2 \in [0, \tfrac{1}{2}),\ \varsigma_3 \in [0, 1),\ \theta \in (0, 1],$$

$$0 < \gamma_1 < 1 < \gamma_2 \leq \gamma_3 \ \text{ and } \ 0 < \eta_1 \leq \eta_2 < 1.$$

Set $k = 0$.

Step 1: Check termination Evaluate $g_k \stackrel{\text{def}}{=} \nabla_x^1 f(x_k)$. Terminate if $\|g_k\| \leq \epsilon$.

Step 2: Compute subspace derivatives Choose $p \in \{1, \ldots, n\}$ and form $V_p \in \mathbb{R}^{n \times p}$. Compute $\widehat{g}_k \stackrel{\text{def}}{=} V_p^T g_k$ and $\widehat{H}_k \stackrel{\text{def}}{=} V_p^T H_k V_p$ where $H_k \stackrel{\text{def}}{=} \nabla_x^2 f(x_k)$.

Step 3 (Optionnal): Attempt a regularization step

$$s_k = s_k^{def} = \mathsf{RegStep}(\, \widehat{g}_k, \widehat{H}_k, V_p, \|g_k\|, \sigma_k, \kappa_a, \kappa_b, \kappa_\theta, \varsigma_1, \varsigma_2\,). \tag{4.2}$$

If s_k^{def} has been successfully defined, go to Step 5.

Step 4 : Newton Step Computation

$$s_k = \mathsf{NewtonEigenStep}(\, \widehat{g}_k, \widehat{H}_k, V_p, \|g_k\|, \sigma_k, \kappa_C, \kappa_b, \kappa_\theta, \varsigma_3, \theta\,). \tag{4.3}$$

Step 5: Acceptance ratio computation Evaluate $f(x_k + s_k)$ and compute the acceptance ratio

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{-(g_k^T s_k + \frac{1}{2} s_k^T H_k s_k)}. \tag{4.4}$$

If $\rho_k \geq \eta_1$, set $x_{k+1} = x_k + s_k$ else $x_{k+1} = x_k$.

Step 6: Regularization parameter update Set

$$\sigma_{k+1} \in \begin{cases} [\max(\sigma_{\min}, \gamma_1 \sigma_k), \sigma_k] & \text{if } \rho_k \geq \eta_2, \\ [\sigma_k, \gamma_2 \sigma_k] & \text{if } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_2 \sigma_k, \gamma_3 \sigma_k] & \text{if } \rho_k < \eta_1. \end{cases} \tag{4.5}$$

Increment k by one and go to Step 1.

The selection of the iteration-dependent subspace defined as the range of V_p in Step 2 is of course crucial for the algorithm. At this stage of the algorithm description, cycling may possibly occur between Step 2 and (4.13) in Step 4, should the choice of the subspace be consistently inadequate. We will however discuss some practical choices in Section 4.5, for

Algorithm 4.2.2: **RegStep**($\widehat{g}_k, \widehat{H}_k, V_p, \|g_k\|, \sigma_k, \kappa_a, \kappa_b, \kappa_\theta, \varsigma_1, \varsigma_2$)

Attempt to solve the linear system

$$(\widehat{H}_k + \sqrt{\kappa_a \sigma_k \|g_k\|} I_p) y_k^{def} = -\widehat{g}_k. \tag{4.6}$$

If a solution y_k^{def} of this system can be obtained such that

$$(y_k^{def})^\intercal (\widehat{H}_k + \sqrt{\kappa_a \sigma_k \|g_k\|} I_p) y_k^{def} > 0, \tag{4.7}$$

$$\|y_k^{def}\| \leq \frac{(1+\kappa_\theta)}{\varsigma_1} \sqrt{\frac{\|g_k\|}{\kappa_a \sigma_k}}, \tag{4.8}$$

$$\|H_k V_p y_k^{def} + g_k\| \leq \kappa_b \|\widehat{H}_k y_k^{def} + \widehat{g}_k\|, \tag{4.9}$$

$$\|r_k^{def}\| \leq \min\left(\varsigma_2 \sqrt{\kappa_a \sigma_k \|g_k\|} \|y_k^{def}\|, \kappa_\theta \|\widehat{g}_k\| \right) \tag{4.10}$$

where $r_k^{def} = (\widehat{H}_k + \sqrt{\kappa_a \sigma_k \|g_k\|} I_p) y_k^{def} + \widehat{g}_k$, then return $s_k^{def} \overset{\text{def}}{=} V_p y_k^{def}$.

which this situation cannot happen. For our subsequent analysis, we therefore assume the following.

AS.0 For each iteration k, condition (4.13) is satisfied after finitely many choices of V_p. Moreover, there exists a constant $V_{\max} \geq 1$ such that

$$\|V_p\| \leq V_{\max} \text{ for all } \quad p \in \{1, \ldots, n\}. \tag{4.16}$$

After selecting the subspace[1] and projecting the current gradient and Hessian, we first attempt a step that avoids computing negative curvature information. Indeed, the s_k^{def} notation, where def stands for "definite", in (4.6) makes the connection with the two conditions (4.6) and (4.7). The condition (4.7) is significantly less restrictive than checking the positive-definess of the regularized matrix in (4.6). This is at variance with the work of [27] where a factorization is required at each step, and coherent with the 'capped-CG' subroutine proposed at [215] and [185]. Should the problem be (locally) convex, (4.7) would automatically hold (see [156, 82]). The test (4.8) is required as to avoid steps whose magnitude is too large compared to the gradient (the motivation for its particular form of the test will become clear in Section 4.3).

When computing a vector s_k^{def} satisfying (4.7) to (4.10) is not possible, we (approximately) solve a linear system in $\mathbb{R}^p$ (4.11) whose definition involves $[-\lambda_{\min}(\widehat{H}_k)]_+$. Even if an exact solution can be obtained at a marginal cost for small p, we still allow an approximate solution satisfying (4.12). We note that $[-\lambda_{\min}(\widehat{H}_k)]_+$ could have been replaced in (4.11) by $\kappa_C \sqrt{\sigma_k \|g_k\|}$ and the remainder of the complexity analysis would remain valid. However, directly using the gradient only as a regularizer is not possible, because theoretically a negative curvature computation is needed to know if a Newton step can be used or not. An interesting connection can also be established between the regularization in (4.11) for $\widehat{H}_k = H_k$ and the GQT method [100], as the regularization parameter $(\sqrt{\sigma_k \|g_k\|} + [-\lambda_{\min}(H_k)]_+)$ is very

[1] Since we do not specify at this point how to make this selection, AN2C may be viewed as a *class of algorithms* depending on the choice of V_p.

Algorithm 4.2.3: NewtonEigenStep($\widehat{g}_k, \widehat{H}_k, V_p, \|g_k\|, \sigma_k, \kappa_C, \kappa_b, \kappa_\theta, \varsigma_3, \theta$)

Step 1: Test negative curvature If $\lambda_{\min}(\widehat{H}_k) \leq -\kappa_C\sqrt{\sigma_k\|g_k\|}$, go to Step 4.

Step 2: Newton Step Solve

$$\left(\widehat{H}_k + (\sqrt{\sigma_k\|g_k\|} + [-\lambda_{\min}(\widehat{H}_k)]_+)I_p\right) y_k^{neig} = -\widehat{g}_k \tag{4.11}$$

to ensure the residual condition

$$\begin{aligned}\|r_k^{neig}\| &\stackrel{\rm def}{=} \left\|\left(\widehat{H}_k + (\sqrt{\sigma_k\|g_k\|} + [-\lambda_{\min}(\widehat{H}_k)]_+)I_p\right) y_k^{neig} + \widehat{g}_k\right\| \\ &\leq \min\left(\varsigma_3\sqrt{\sigma_k\|g_k\|}\|y_k^{neig}\|, \kappa_\theta\|\widehat{g}_k\|\right).\end{aligned} \tag{4.12}$$

Step 3: Check global quality of the solution If

$$\|H_kV_py_k^{neig} + g_k\| \leq \kappa_b\|\widehat{H}_ky_k^{neig} + \widehat{g}_k\| \quad \text{then set} \quad s_k = s_k^{neig} \stackrel{\rm def}{=} V_py_k^{neig}. \tag{4.13}$$

Else, go back AN2C[Step 2].

Step 4: Eigenvector direction Compute u_k such that

$$\widehat{g}_k^Tu_k \leq 0,\ \|u_k\| = 1,\ u_k^T\widehat{H}_ku_k \leq \theta\lambda_{\min}(\widehat{H}_k) \ \text{ and } \ u_k^T\widehat{H}_k^2u_k \leq \frac{\lambda_{\min}(\widehat{H}_k)^2}{\theta^2} \tag{4.14}$$

and set

$$s_k = s_k^{curv} \stackrel{\rm def}{=} \frac{\theta\kappa_C\sqrt{\sigma_k\|g_k\|}}{\sigma_k}V_pu_k. \tag{4.15}$$

similar in spirit to that used in this method. We should note that a complexity analysis of the GQT technique has only been provided when the algorithm is equipped with an Armijo linesearch. And it naturally retrieves the suboptimal standard rate of linesearch gradient descent, see [196] for more details. In the closely related algorithm of [26], a term μ is added to $[-\lambda_{\min}(H_k)]_+$ and multiple μ's are tested so as to ensure 'cubic' descent. In our case, $\sqrt{\sigma_k\|g_k\|}$ directly yields a regularization of the desired order. Also observe that, in most cases, the "approximate minimum curvature direction" u_k is already available when computing $\lambda_{\min}(\widehat{H}_k)$. It can be also retrieved via a Lanczos procedure as proposed in [184, Lemma 9].

We now provide some comments that apply to the definition of both s_k^{def} and s_k^{neig}. Specifically, focusing on the latter, condition (4.13) or condition (4.9) serves to ensure the appropriateness of the subspace spanned by V_p. This condition guarantees that the projected residual (4.12) is sufficiently small compared to both the projected and unprojected gradients. In a more standard setting, where $V_p = I_n$, this condition simplifies to

$$\|(H_k + (\sqrt{\sigma_k\|g_k\|} + [-\lambda_{\min}(H_k)]_+)I_n)s_k + g_k\| \leq \min(\varsigma_3\sqrt{\sigma_k\|g_k\|}\|s_k\|, \kappa_\theta\|g_k\|), \tag{4.17}$$

where the $\kappa_\theta\|g_k\|$ term is standard when devising truncated CG algorithms. The other term ensures the typical condition required for the approximate minimization of the cubic model m_k, namely that

$$\|\nabla_s^1 m_k(s_k)\| \leq \mathcal{O}(\|s_k\|^2). \tag{4.18}$$

This condition is typically used to derive the optimal complexity rate $\mathcal{O}\left(\epsilon^{-3/2}\right)$, see [51, 29] and the references therein.

Regarding the negative curvature step, even though the last condition in (4.14) is not usually required when using negative curvature, it will be needed later on in our analysis. Note that a properly chosen eigenvector associated to $\lambda_{\min}(\widehat{H}_k)$ satisfy the requirements of (4.14). Finally we note that the condition of Step 1 in the NewtonEigenStep algorithm, which forces the negative curvature step (4.15), can be interpreted as the comparison of the minimal curvature of the quadratic $(\lambda_{\min}(V_p^T H_k V_p))$ with the quantity $\sigma_k\sqrt{\|g_k\|/\sigma_k}$, which itself can be viewed as the curvature of the regularization term $\frac{1}{6}\sigma_k\|s\|^3$ for some s whose length $\sqrt{\|g_k\|/\sigma_k}$ is of the order order of the analog of the trust-region radius for standard cubic regularization (see [50, Lemma 2.1], for instance). The test thus ensures a "regularization-like" step when the quadratic's negative curvature is strong enough to dominate that of the regularization for too small steps (see (4.29) below).

Once the step has been computed, the mechanisms for accepting/rejecting the new iterate (Step 5) and updating the regularization parameter (Step 6) are typical of adaptive regularization algorithms (see [29, 51] or [57, Section 3.3.1], for instance).

Before delving into the complexity analysis of AN2C, we further explore its fundamental properties and discuss its relationships with closely related nonconvex optimization algorithms. The method presented in [71] differs from AN2C in that it employs a gradient step followed by a negative curvature step. On the other hand, [145] adopts a condition-based approach to choose between gradient descent and negative curvature directions, relying on known smoothness parameters, while our methods remain fully adaptive. Another related approach is presented in [184], which, unlike AN2C, examines various conditions to select a specific direction (gradient, Newton, negative curvature) and performs a linesearch. Furthermore, [75] proposes a trust-region algorithm (in contrast to adaptive regularization) that tackles the trust-region subproblem using a combination of conjugate gradients and negative curvature. Notably, their condition on the residuals of this subproblem [75, Inequality (3.2)] can be related to (4.17). A final work worth mentioning before we begin our analysis is that of [72]. The latter extended the exact trust-region of [74] to large scale settings by allowing inexact step computation and the use of Krylov methods. Two conditions are proposed in the theoretical analysis on the residuals and one of them shares some similarity with (4.17).

Following well-established practice, we now define

$$\mathcal{S} \stackrel{\text{def}}{=} \{k \geq 0 \mid x_{k+1} = x_k + s_k\} = \{k \geq 0 \mid \rho_k \geq \eta_1\},$$

the set of indexes of "successful iterations", and

$$\mathcal{S}_k \stackrel{\text{def}}{=} \mathcal{S} \cap \{0, \ldots, k\},$$

the set of indexes of successful iterations up to iteration k. We further partition $\mathcal{S}_k$ in three

subsets depending on the nature of the step taken, so that

$$\mathcal{S}_k^{neig} \stackrel{\text{def}}{=} \mathcal{S}_k \cap \{s_k = s_k^{neig}\}, \quad \mathcal{S}_k^{curv} \stackrel{\text{def}}{=} \mathcal{S}_k \cap \{s_k = s_k^{curv}\}, \quad \mathcal{S}_k^{def} \stackrel{\text{def}}{=} \mathcal{S}_k \cap \{s_k = s_k^{def}\}.$$

We also recall a well-known result bounding the total number of iterations of adaptive regularization methods in terms of the number of successful ones.

Lemma 4.2.1 *Suppose that the AN2C algorithm is used and that $\sigma_k \leq \sigma_{\max}$ for some $\sigma_{\max} > 0$. Then*

$$k \leq |\mathcal{S}_k| \left(1 + \frac{|\log \gamma_1|}{\log \gamma_2}\right) + \frac{1}{\log \gamma_2} \log \left(\frac{\sigma_{\max}}{\sigma_0}\right). \tag{4.19}$$

Proof. The proof can be found in [57, Lemma 2.4.1]. We restate here for the sake of completeness. The regularization parameter update (4.5), we derive that

$$\gamma_1 \sigma_i \leq \max(\gamma_1 \sigma_i, \sigma_{\min}) \leq \sigma_{i+1}, \quad i \in \mathcal{S}_k, \quad \text{and} \quad \gamma_2 \sigma_i \leq \sigma_{i+1}, \quad i \in \{0, \ldots, k\} \setminus \mathcal{S}_k.$$

By induction, we deduce that

$$\sigma_0 \gamma_1^{|\mathcal{S}_k|} \gamma_2^{k-|\mathcal{S}_k|} \leq \gamma_k,$$

since $\sigma_k \leq \sigma_{\max}$, we deduce

$$|\mathcal{S}_k| \log \gamma_1 + (k - |\mathcal{S}_k|) \log \gamma_2 \leq \log \left(\frac{\sigma_{\max}}{\sigma_0}\right).$$

Rearranging the last inequality and using that $\gamma_1 < 1$ yields (4.19). □

This result implies that the overall complexity of the algorithm can be estimated once bounds on σ_k and $|\mathcal{S}_k|$ are known, as we will show in the next section.

We now state three simple relations between the size of the stepsize s_k in all three cases ($\|s_k^{neig}\|$, $\|s_k^{def}\|$, $\|s_k^{curv}\|$), σ_k, $\|g_k\|$ and other algorithmic constants.

Lemma 4.2.2 *For all iterations k where s_k^{neig} is computed, we have that*

$$\|s_k^{neig}\| \leq V_{\max}\|y_k^{neig}\| \tag{4.20}$$

$$\widehat{g}_k = -\left(\widehat{H}_k + (\sqrt{\sigma_k\|g_k\|} + [-\lambda_{\min}(\widehat{H}_k)]_+)I_p\right) y_k^{neig} + r_k^{neig} \tag{4.21}$$

and

$$\|s_k^{neig}\| \leq (1+\kappa_\theta)V_{\max}^{\frac{3}{2}}\sqrt{\frac{\|g_k\|}{\sigma_k}}. \tag{4.22}$$

Similarly, when s_k^{def} is computed,

$$\widehat{g}_k = -\left(\widehat{H}_k + \sqrt{\kappa_a\sigma_k\|g_k\|}I_p\right) y_k^{def} + r_k^{def}, \tag{4.23}$$

and

$$\|s_k^{def}\| \leq V_{\max}\|y_k^{def}\|. \tag{4.24}$$

At last, when s_k^{curv} is computed,

$$\|s_k^{curv}\| \leq V_{\max}\frac{\theta\kappa_C\sqrt{\sigma_k\|g_k\|}}{\sigma_k}. \tag{4.25}$$

Proof. First note from the second part of (4.13) and (4.16),

$$\|s_k^{neig}\| = \|V_p y_k^{neig}\| \leq V_{\max}\|y_k^{neig}\|$$

yielding (4.20). A similar proof can be followed to derive (4.24).

Equation (4.21) results from (4.11), (4.13) and the definition of the residual (4.12). Let us rewrite now (4.21) in function of $\widehat{g}_k$ and $\widehat{H}_k$,

$$\widehat{g}_k = -\left(\widehat{H}_k + (\sqrt{\sigma_k\|g_k\|} + [-\lambda_{\min}(\widehat{H}_k)]_+)I_p\right) y_k^{neig} + r_k^{neig}.$$

From (4.21), the facts that $\widehat{H}_k + (\sqrt{\sigma_k\|g_k\|} + [-\lambda_{\min}(\widehat{H}_k)]_+)I_p$ is a positive definite matrix with

$$\lambda_{\min}(\widehat{H}_k + (\sqrt{\sigma_k\|g_k\|} + [-\lambda_{\min}(\widehat{H}_k)]_+)I_p) \geq \sqrt{\sigma_k\|g_k\|}$$

and that $\|r_k^{neig}\| \leq \kappa_\theta\|\widehat{g}_k\|$ because of (4.12). We thus obtain that

$$\|y_k^{neig}\| \leq (1+\kappa_\theta)\sqrt{\frac{\|\widehat{g}_k\|}{\sigma_k}} \leq (1+\kappa_\theta)\sqrt{\frac{V_{\max}\|g_k\|}{\sigma_k}}, \tag{4.26}$$

where the last inequality follows from (4.16). This last inequality and (4.20) give (4.22).

If $k \in \mathcal{S}_k^{def}$, (4.23) is obtained from (4.6) and the definition of the residual r_k^{def}.

Else if $k \in \mathcal{S}_k^{curv}$, (4.16), the fact that $\|u_k\| = 1$ and (4.15) give (4.25). □

The next lemma gives a lower bound on the decrease of the local quadratic approximation. In standard adaptive regularization algorithms, this decrease automatically results from the minimization of the model (See [29] for instance). In our case, we need to use the properties of s_k^{def}, s_k^{curv} and s_k^{neig} to obtain the desired result.

Lemma 4.2.3 *Let k be a successful an iteration of AN2C. If $k \in \mathcal{S}_k^{def}$, we have that*

$$-\left(g_k^T s_k + \frac{1}{2} s_k^T H_k s_k\right) \geq \frac{1-2\varsigma_2}{2}\sqrt{\kappa_a \sigma_k \|g_k\|}\|y_k^{def}\|^2 \geq \frac{1-2\varsigma_2}{2}\sqrt{\kappa_a \sigma_k \|g_k\|}\frac{\|s_k\|^2}{V_{\max}^2}. \quad (4.27)$$

If $k \in \mathcal{S}_k^{neig}$, then

$$-\left(g_k^T s_k + \frac{1}{2} s_k^T H_k s_k\right) \geq (1-\varsigma_3)\sqrt{\sigma_k \|g_k\|}\|y_k^{neig}\|^2 \geq (1-\varsigma_3)\sqrt{\sigma_k \|g_k\|}\frac{\|s_k\|^2}{V_{\max}^2}. \quad (4.28)$$

Else, if $k \in \mathcal{S}_k^{curv}$,

$$-\left(g_k^T s_k + \frac{1}{2} s_k^T H_k s_k\right) \geq \frac{1}{2}\theta^3 \kappa_C^3 \frac{\|g_k\|^{\frac{3}{2}}}{\sqrt{\sigma_k}} \geq \frac{1}{2}\sigma_k \frac{\|s_k\|^3}{V_{\max}^3}. \quad (4.29)$$

Proof. Suppose first that $k \in \mathcal{S}_k^{def}$. We then obtain from (4.2), (4.10) and (4.8) that

$$\begin{aligned}
g_k^T s_k^{def} + \frac{1}{2}(s_k^{def})^T H_k s_k^{def} &= (V_p^T g_k)^T y_k^{def} + \frac{1}{2}(y_k^{def})^T V_p^T H_k V_p y_k^{def} \\
&= (r_k^{def})^T y_k^{def} - (y_k^{def})^T(\widehat{H}_k + \sqrt{\kappa_a \sigma_k \|g_k\|} I_p) y_k^{def} \\
&\quad + \frac{1}{2}(y_k^{def})^T \widehat{H}_k y_k^{def} \\
&= -\sqrt{\kappa_a \sigma_k \|g_k\|}\|y_k^{def}\|^2 + (r_k^{def})^T y_k^{def} - \frac{1}{2}(y_k^{def})^T \widehat{H}_k y_k^{def} \\
&\leq -\sqrt{\kappa_a \sigma_k \|g_k\|}\|y_k^{def}\|^2 + \varsigma_2\sqrt{\kappa_a \sigma_k \|g_k\|}\|y_k^{def}\|^2 \\
&\quad + \frac{1}{2}\sqrt{\kappa_a \sigma_k \|g_k\|}\|y_k^{def}\|^2.
\end{aligned}$$

Hence (4.27) follows froms (4.24).

Suppose now that $k \in \mathcal{S}_k^{neig}$. By using (4.21) and the fact that $\widehat{H}_k + [-\lambda_{\min}(\widehat{H}_k)]_+ I_p \succeq 0$,

$$\begin{aligned}
g_k^\intercal s_k^{neig} + \frac{1}{2}(s_k^{neig})^\intercal H_k s_k^{neig} &= (V_p^\intercal g_k)^\intercal y_k^{neig} + \frac{1}{2}(y_k^{neig})^\intercal V_p^\intercal H_k V_p y_k^{neig} \\
&= (r_k^{neig})^\intercal y_k^{neig} - (y_k^{neig})^\intercal (\widehat{H}_k + [-\lambda_{\min}(\widehat{H}_k)]_+ I_p) y_k^{neig} \\
&\quad + \frac{1}{2}(y_k^{neig})^\intercal (\widehat{H}_k + [-\lambda_{\min}(\widehat{H}_k)]_+ I_p) y_k^{neig} \\
&\quad - \frac{1}{2}[-\lambda_{\min}(\widehat{H}_k)]_+ \|y_k^{neig}\|^2 - \sqrt{\sigma_k \|g_k\|} \|y_k^{neig}\|^2 \\
&= (r_k^{neig})^\intercal y_k^{neig} - \frac{1}{2}(y_k^{neig})^\intercal (\widehat{H}_k + [-\lambda_{\min}(\widehat{H}_k)]_+ I_p) y_k^{neig} \\
&\quad - \frac{1}{2}[-\lambda_{\min}(\widehat{H}_k)]_+ \|y_k^{neig}\|^2 - \sqrt{\sigma_k \|g_k\|} \|y_k^{neig}\|^2 \\
&\leq \varsigma_3 \sqrt{\sigma_k \|g_k\|} \|y_k^{neig}\|^2 - \frac{1}{2}[-\lambda_{\min}(\widehat{H}_k)]_+ \|y_k^{neig}\|^2 - \sqrt{\sigma_k \|g_k\|} \|y_k^{neig}\|^2,
\end{aligned}$$

where we have used (4.12) to obtain the last inequality. Rearranging, ignoring the $\frac{1}{2}[-\lambda_{\min}(\widehat{H}_k)]_+ \|y_k^{neig}\|^2$ term and using (4.20) yield (4.28).

Suppose finally that $k \in \mathcal{S}_k^{curv}$. As (4.14) and (4.15) hold and that Step 4 of NewtonEigenStep is taken when $\lambda_{\min}(\widehat{H}_k) \leq -\kappa_C \sqrt{\sigma_k \|g_k\|}$, we deduce that

$$\begin{aligned}
g_k^\intercal s_k^{curv} + \frac{1}{2}(s_k^{curv})^\intercal H_k s_k^{curv} &= \frac{\theta \kappa_C \sqrt{\sigma_k \|g_k\|}}{\sigma_k} g_k^\intercal V_p u_k + \frac{1}{2}(s_k^{curv})^\intercal H_k s_k^{\sigma_k} \\
&\leq \frac{1}{2}\frac{\theta^2 \kappa_C^2 \|g_k\|}{\sigma_k} u_k^\intercal \widehat{H}_k u_k \leq \frac{1}{2}\frac{\theta^3 \kappa_C^2 \|g_k\|}{\sigma_k} \lambda_{\min}(\widehat{H}_k) \\
&\leq -\frac{1}{2}\theta^3 \kappa_C^3 \frac{\|g_k\|^{\frac{3}{2}}}{\sqrt{\sigma_k}}, \qquad (4.30)
\end{aligned}$$

yielding the first inequality in (4.29). For the second inequality, remark that from (4.25), we derive that

$$\theta^3 \kappa_C^3 \frac{\|g_k\|^{\frac{3}{2}}}{\sqrt{\sigma_k}} = \sigma_k \theta^3 \kappa_C^3 \frac{\|g_k\|^{\frac{3}{2}}}{\sigma_k^{\frac{3}{2}}} \geq \sigma_k \frac{\|s_k\|^3}{V_{\max}^3},$$

injecting the last bound in (4.30) gives the second inequality in (4.29). □

4.3 Complexity analysis for the AN2C algorithm

We now turn to analyzing the worst-case complexity of the AN2C algorithm. Our analysis is conducted under AS.0 and the following assumptions.

AS.1 The function f is in the $\mathcal{C}^{2,1}(\mathbb{R}^n;\mathbb{R})$ class with L_H the Hessian Lipschitz constant.

AS.2 There exists a constant f_{low} such that $f(x) \geq f_{\text{low}}$ for all $x \in \mathbb{R}^n$.

AS.3 There exists a constant $\kappa_B > 0$ such that

$$\max(0, -\lambda_{\min}(\nabla_x^2 f(x))) \leq \kappa_B \text{ for all } x \in \{y \in \mathbb{R}^n \mid f(y) \leq f(x_0)\}.$$

AS.1-AS.2 are standard assumptions when analyzing algorithms that utilize second-order information [51, 29]. AS.3 is weaker than assuming bounded Hessians, a condition often used when theoretically analyzing second-order methods that combines negative curvature and gradient based directions [184, 71, 145]. The left-hand side of the inequality is sometimes called the "convex deviation", "modulus of nonconvexity" [137] or "weak-convexity" constant [176]. As it turns out, AS.4 is only needed for x being any iterate x_k produced by the algorithm and these iterates all belong to the level associated with the starting point x_0 because the acceptance condition in Step 5 ensures that the sequence $\{f(x_k)\}$ is non-increasing. If this level set is bounded or if the sequence $\{x_k\}$ remains bounded for any other reason, we immediately obtain that

$$\max(0, -\lambda_{\min}(H_k)) \leq \kappa_B \quad \text{for all} \quad k \geq 0 \tag{4.31}$$

for some $\kappa_B \geq 0$, and both AS.2 and AS.3 automatically hold for an open bounded convex set containing all line segments $[x_k, x_k + s_k]$. Having established a lower bound on the decrease ratio in Lemma 4.2.3, we next proceed to derive an upper bound on the regularization parameter. This is a crucial step when analyzing adaptive regularization methods.

Lemma 4.3.1 *Suppose that AS.1 holds. Then, for all $k \geq 0$,*

$$\sigma_k \leq \sigma_{\max} \stackrel{def}{=} \gamma_3 \max\left(\sigma_0,\, \varsigma_{\max} \frac{L_H}{6(1-\eta_2)}\right), \tag{4.32}$$

where

$$\varsigma_{\max} \stackrel{def}{=} \max\left(\frac{(1+\kappa_\theta)V_{\max}^{\frac{7}{2}}}{(1-\varsigma_3)},\, \frac{2(1+\kappa_\theta)V_{\max}^3}{\kappa_a \varsigma_1 (1-2\varsigma_2)},\, 2\,V_{\max}^3\right). \tag{4.33}$$

Proof. Let us compute the ratio ρ_k for $k \in \mathcal{S}_k^{neig}$. By using AS.1 and the standard error bound for Lispschitz approximation of the function (see inequality (1.3.15) of Lemma 1.3.1), that $\varsigma_3 < 1$, (4.28) and (4.22), we obtain that

$$\begin{aligned}
1 - \rho_k &= \frac{f(x_k + s_k) - f(x_k) - g_k^T s_k - \frac{1}{2} s_k^T H_k s_k}{-(g_k^T s_k + \frac{1}{2} s_k^T H_k s_k)} \\
&\leq \frac{L_H V_{\max}^2 \|s_k^{neig}\|^3}{6(1-\varsigma_3)\sqrt{\sigma_k \|g_k\|}\|s_k^{neig}\|^2} \\
&\leq \frac{L_H V_{\max}^2 \|s_k^{neig}\|}{6(1-\varsigma_3)\sqrt{\sigma_k \|g_k\|}} \\
&\leq \frac{L_H (1+\kappa_\theta) V_{\max}^{\frac{7}{2}}}{6(1-\varsigma_3)\sigma_k}.
\end{aligned} \tag{4.34}$$

Hence, if $\sigma_k \geq \frac{L_H(1+\kappa_\theta)V_{\max}^{\frac{7}{2}}}{6(1-\varsigma_3)(1-\eta_2)}$, then $\rho_k \geq \eta_2$, which implies that iteration k is successful and

$\sigma_{k+1} \leq \sigma_k$ because of (4.5). The mechanism of (4.5) in the algorithm then ensures that

$$\sigma_k \leq \gamma_3 \max\left(\sigma_0, \frac{L_H(1+\kappa_\theta)V_{\max}^{\frac{7}{2}}}{6(1-\varsigma_3)(1-\eta_2)}\right). \tag{4.35}$$

Similarly, if $k \in \mathcal{S}_k^{def}$, we use AS.1, the Lipschitz approximation error bound, the fact that $\varsigma_2 < \frac{1}{2}$, (4.27), (4.8) and (4.24) to deduce that

$$1-\rho_k \leq \frac{L_H\|s_k^{def}\|V_{\max}^2}{3(1-2\varsigma_2)\sqrt{\kappa_a\sigma_k\|g_k\|}} \leq \frac{L_H(1+\kappa_\theta)V_{\max}^3}{3\kappa_a\varsigma_1(1-2\varsigma_2)\sigma_k}.$$

Using the same argument as above, we now obtain that

$$\sigma_k \leq \gamma_3 \max\left(\sigma_0, \frac{L_H(1+\kappa_\theta)V_{\max}^3}{3\kappa_a\varsigma_1(1-2\varsigma_2)(1-\eta_2)}\right). \tag{4.36}$$

Consider finally the case where $k \in \mathcal{S}_k^{curv}$. Again using AS.1, the Lipschitz approximation error bound and (4.29) lower-bound, we derive that

$$1-\rho_k = \frac{f(x_k+s_k)-f(x_k)-g_k^Ts_k-\frac{1}{2}s_k^TH_ks_k}{-g_k^Ts_k-\frac{1}{2}s_k^TH_ks_k} \leq \frac{L_H\|s_k^{curv}\|^3V_{\max}^3}{6\frac{1}{2}\sigma_k\|s_k^{curv}\|^3} = \frac{L_HV_{\max}^3}{3\sigma_k},$$

so that

$$\sigma_k \leq \gamma_3 \max\left(\sigma_0, \frac{L_HV_{\max}^3}{3(1-\eta_2)}\right). \tag{4.37}$$

Combining (4.35), (4.36) and (4.37) gives (4.32) with $\varsigma_{\max}$ defined by (4.33). □

We now prove a lower bound on the decrease at a successful iteration k using negative curvature. We will also bound the change in the norm $\|g_{k+1}\|$ in term of $\|g_k\|$, which will be useful later to bound the cardinality of a subset of $\mathcal{S}_k^{neig} \cup \mathcal{S}_k^{curv}$.

Lemma 4.3.2 *Suppose that AS.1 and AS.3 hold and that $k \in \mathcal{S}_k^{curv}$ before termination. Then*

$$f(x_k)-f(x_{k+1}) \geq \frac{\eta_1\theta^3\kappa_C^3}{2\sqrt{\sigma_{\max}}}\epsilon^{\frac{3}{2}}, \tag{4.38}$$

and

$$\|g_{k+1}\| \leq \left(\frac{L_HV_{\max}^2}{2\sigma_k}\kappa_C^2\theta^2 + \frac{\kappa_B\kappa_C}{\sqrt{\epsilon\sigma_k}} + 1\right)\|g_k\|. \tag{4.39}$$

Proof. Let $k \in \mathcal{S}_k^{curv}$. From (4.4) and (4.29), we obtain that

$$f(x_k)-f(x_{k+1}) \geq \eta_1\left(-g_k^Ts_k - \frac{1}{2}s_k^TH_ks_k\right) \geq \frac{\eta_1\theta^3\kappa_C^3}{2\sqrt{\sigma_k}}\|g_k\|^{\frac{3}{2}}.$$

Since $\|g_k\| \geq \epsilon$ before termination and that $\sigma_k \leq \sigma_{\max}$ by Lemma 4.3.1, we obtain (4.38).

Let us now prove (4.39). By using the Lipschitz error bound for the gradient (see (1.3.16) from Lemma 1.3.1), the triangular inequality, the fact that $k \in \mathcal{S}_k^{curv}$, (4.14), (4.15), and (4.25), we obtain that

$$\begin{aligned}
\|g_{k+1}\| &\leq \|g_{k+1} - g_k - H_k s_k\| + \|H_k s_k + g_k\| \\
&\leq \frac{L_H}{2}\|s_k\|^2 + \|g_k\| + \|H_k s_k\| \\
&= \frac{L_H}{2}\|s_k^{curv}\|^2 + \|g_k\| + \|H_k s_k^{curv}\| \\
&\leq \frac{L_H V_{\max}^2}{2\sigma_k}\kappa_C^2\theta^2\|g_k\| + \|g_k\| + \|H_k s_k^{curv}\|. \qquad (4.40)
\end{aligned}$$

Now, using (4.14), (4.15) again,

$$\begin{aligned}
\|H_k s_k^{curv}\| &= \theta\kappa_C\sqrt{\frac{\|g_k\|}{\sigma_k}}\|H_k V_p u_k\| = \theta\kappa_C\sqrt{\frac{\|g_k\|}{\sigma_k}}\sqrt{u_k^T \widehat{H}_k^2 u_k} \\
&\leq \kappa_C\sqrt{\frac{\|g_k\|}{\sigma_k}}|\lambda_{\min}(\widehat{H}_k)| \leq \kappa_C\sqrt{\frac{\|g_k\|}{\sigma_k}}|\lambda_{\min}(H_k)|.
\end{aligned}$$

Hence (4.40) together with AS.4 and the fact $\|g_k\| \geq \epsilon$ before termination, give that

$$\begin{aligned}
\|g_{k+1}\| &\leq \frac{L_H V_{\max}^2}{2\sigma_k}\kappa_C^2\theta^2\|g_k\| + \|g_k\| + \kappa_B\kappa_C\sqrt{\frac{\|g_k\|}{\sigma_k}} \\
&= \left(\frac{L_H V_{\max}^2}{2\sigma_k}\kappa_C^2\theta^2 + \frac{\kappa_B\kappa_C}{\sqrt{\sigma_k\|g_k\|}} + 1\right)\|g_k\| \\
&\leq \left(\frac{L_H V_{\max}^2}{2\sigma_k}\kappa_C^2\theta^2 + \frac{\kappa_B\kappa_C}{\sqrt{\sigma_k\epsilon}} + 1\right)\|g_k\|,
\end{aligned}$$

yielding (4.39). □

This lemma is the only result requiring AS.3 or its weaker formulation (4.31). Note that this assumption is only required along directions of negative curvature, which we expect to occur rarely in practice for suitably large choices of κ_C.

After proving a lower bound on the quadratic's decrease when $k \in \mathcal{S}_k^{def}$, we now exhibit a relationship between both the decrease in objective function and that in gradient norm at iteration k and $k+1$ for $k \in \mathcal{S}_k^{neig} \cup \mathcal{S}_k^{def}$. This is also where the two global conditions (4.13) and (4.9) on the subspace V_p will be useful. Moreover, we also prove an inequality between the norms of the gradient at two successive iterations, similar to (4.39).

Lemma 4.3.3 *Suppose that AS.1 holds and that $k \in \mathcal{S}_k^{neig} \cup \mathcal{S}_k^{def}$ before termination. Then*

$$\|g_{k+1}\| \leq \left(\frac{L_H V_{\max}^3 (1+\kappa_\theta)}{2\varsigma_1^2 \sigma_k} + \frac{2\kappa_b \sqrt{V_{\max}}}{\varsigma_1} + \kappa_b \kappa_C \sqrt{V_{\max}} \right) (1+\kappa_\theta) \|g_k\| \tag{4.41}$$

and

$$f(x_k) - f(x_{k+1}) \geq \eta_1 \, \varsigma_{\min} \sqrt{\sigma_k \|g_k\|} \left(\frac{-(2+\kappa_C)\kappa_b \sqrt{\kappa_a \sigma_k \|g_k\|} + \sqrt{(\kappa_b(2+\kappa_C))^2 \kappa_a \sigma_k \|g_k\| + 2\, V_{\max}^2 L_H \|g_{k+1}\|}}{L_H V_{\max}^2} \right)^2 \tag{4.42}$$

where

$$\varsigma_{\min} \stackrel{def}{=} \min \left(\frac{1-2\varsigma_2}{2}, 1-\varsigma_3 \right). \tag{4.43}$$

Proof. Consider first the case where $k \in \mathcal{S}_k^{neig}$. By using the Lipschitz error bound for the gradient ((1.3.16) from Lemma (1.3.1)), that (4.13) holds, r_k^{neig} expression (4.21), the condition on $\|r_k^{neig}\|$ (4.12) and the fact that $[-\lambda_{\min}(\widehat{H}_k)]_+ \leq \kappa_C \sqrt{\sigma_k \|g_k\|}$ for $k \in \mathcal{S}_k^{neig}$, we deduce that

$$\begin{aligned}
\|g_{k+1}\| &\leq \|g_{k+1} - H_k s_k^{neig} - g_k\| + \|H_k s_k^{neig} + g_k\| \\
&\leq \frac{L_H}{2} \|s_k^{neig}\|^2 + \kappa_b \|\widehat{H}_k y_k^{neig} + \widehat{g}_k\| \\
&\leq \frac{L_H}{2} \|s_k^{neig}\|^2 + \kappa_b (\sqrt{\sigma_k \|g_k\|} + [-\lambda_{\min}(\widehat{H}_k)]_+) \|y_k^{neig}\| + \kappa_b \|r_k^{neig}\| \\
&\leq \frac{L_H}{2} \|s_k^{neig}\|^2 + \kappa_b (1+\kappa_C) \sqrt{\sigma_k \|g_k\|} \|y_k^{neig}\| + \kappa_b \varsigma_3 \sqrt{\sigma_k \|g_k\|} \|y_k^{neig}\|.
\end{aligned} \tag{4.44}$$

Using now (4.26) and (4.22) in the last inequality

$$\|g_{k+1}\| \leq \left(\frac{L_H V_{\max}^3}{2\sigma_k} (1+\kappa_\theta) + \kappa_b (1+\kappa_C) \sqrt{V_{\max}} + \varsigma_3 \kappa_b \sqrt{V_{\max}} \right) (1+\kappa_\theta) \|g_k\|. \tag{4.45}$$

Consider now $k \in \mathcal{S}_k^{def}$. By arguments similar to those used for (4.44), this time with (4.23), (4.9) and (4.10), we obtain that

$$\begin{aligned}
\|g_{k+1}\| &\leq \|g_{k+1} - H_k s_k^{def} - g_k\| + \|H_k s_k^{def} + g_k\| \\
&\leq \|g_{k+1} - H_k s_k^{def} - g_k\| + \kappa_b \|\widehat{H}_k y_k^{def} + \widehat{g}_k\| \\
&\leq \frac{L_H}{2} \|s_k^{def}\|^2 + \kappa_b \sqrt{\kappa_a \sigma_k \|g_k\|} \|y_k^{def}\| + \kappa_b \|r_k^{def}\| \\
&\leq \frac{L_H}{2} \|s_k^{def}\|^2 + \kappa_b \sqrt{\kappa_a \sigma_k \|g_k\|} \|y_k^{def}\| + \kappa_b \varsigma_2 \sqrt{\kappa_a \sigma_k \|g_k\|} \|y_k^{def}\|.
\end{aligned} \tag{4.46}$$

Bounding $\|s_k^{def}\|$ with (4.24) and utilizing (4.8) yields that

$$\|g_{k+1}\| \leq \left(\frac{L_H(1+\kappa_\theta)V_{\max}^2}{2\varsigma_1^2\kappa_a\sigma_k} + \frac{\kappa_b(1+\varsigma_2)}{\varsigma_1} \right)(1+\kappa_\theta)\,\|g_k\|, \tag{4.47}$$

so that taking the larger bound for both (4.45) and (4.47) and using the bounds $\varsigma_1 < 1$, $\varsigma_2 < \frac{1}{2}$, $\varsigma_3 < 1$, $V_{\max} \geq 1$ and $\kappa_b \geq 1$ gives (4.41).

Finally, from (4.46), (4.44), (4.20), (4.24), the bounds $\max(\varsigma_3, \varsigma_2) < 1$ and $\kappa_a \geq 1$, we obtain that, for $k \in \mathcal{S}_k^{def} \cup \mathcal{S}_k^{neig}$,

$$\frac{L_H V_{\max}^2}{2}\|y_k\|^2 + \kappa_b(2+\kappa_C)\sqrt{\kappa_a\sigma_k\|g_k\|}\|y_k\| - \|g_{k+1}\| \geq 0.$$

Hence $\|y_k\|$ is larger than the positive root of this quadratic and therefore

$$\|y_k\| \geq \frac{-\kappa_b(2+\kappa_C)\sqrt{\kappa_a\sigma_k\|g_k\|} + \sqrt{\kappa_b^2(2+\kappa_C)^2\kappa_a\sigma_k\|g_k\| + 2L_H V_{\max}^2\|g_{k+1}\|}}{L_H V_{\max}^2} > 0.$$

We then deduce (4.42) from this inequality, (4.4), the lower bounds on the quadratic decrease for $k \in \mathcal{S}_k^{neig}$ or $k \in \mathcal{S}_k^{def}$ ((4.28) and (4.27) respectively) and the definition of $\varsigma_{\min}$ in (4.43). □

The bound (4.42) is not sufficient for deriving the required $\mathcal{O}\left(\epsilon^{-3/2}\right)$ optimal complexity rate because the decrease depends on both $\|g_{k+1}\|$ and $\|g_k\|$. Indeed, when $\|g_{k+1}\| \ll \|g_k\|$, the right-hand side of (4.42) tends to zero. To circumvent this difficulty, the next lemma borrows some elements of [156, Theorem 1] and partitions $\mathcal{S}_k^{neig} \cup \mathcal{S}_k^{def}$ in two further subsets. The minimum decrease on the objective function is of the required magnitude in the first one while no meaningful information can be derived on the decrease on the function value in the second, albeit the magnitude of the gradient at the next iteration is halved. The bounds (4.41) and (4.39) are then used to bound the cardinality of the latter set.

Lemma 4.3.4 *Suppose that AS.1 and AS.3 hold and that $\mathcal{S}_k^{neig} \cup \mathcal{S}_k^{def}$ is partitioned as*

$$\mathcal{S}_k^{neig} \cup \mathcal{S}_k^{def} = \mathcal{S}_k^{decr} \cup \mathcal{S}_k^{divgrad} \tag{4.48}$$

where

$$\mathcal{S}_k^{decr} \stackrel{def}{=} \{k \in \mathcal{S}_k^{neig} \cup \mathcal{S}_k^{def},\ \sigma_k\|g_k\| \leq \kappa_m 2L_H\|g_{k+1}\|\}, \tag{4.49}$$

$$\mathcal{S}_k^{divgrad} \stackrel{def}{=} \{k \in \mathcal{S}_k^{neig} \cup \mathcal{S}_k^{def},\ \sigma_k\|g_k\| > \kappa_m 2L_H\|g_{k+1}\|\} \tag{4.50}$$

with

$$\kappa_m \stackrel{def}{=} \gamma_3 \max\left(\frac{\sigma_0}{L_H}, \frac{\varsigma_{\max}}{6(1-\eta_2)}\right). \tag{4.51}$$

Then, for all $k \in \mathcal{S}_k^{decr}$,

$$f(x_k) - f(x_{k+1}) \geq \frac{\eta_1 \varsigma_{\min}(\sigma_k\|g_k\|)^{\frac{3}{2}}}{\left(\kappa_m L_H\left(\kappa_b(2+\kappa_C)\sqrt{\kappa_a} + \sqrt{(\kappa_b(2+\kappa_C))^2\kappa_a + \frac{V_{\max}^2}{\kappa_m}}\right)\right)^2}. \tag{4.52}$$

Moreover,

$$|\mathcal{S}_k^{divgrad}| \leq \kappa_n|\mathcal{S}_k^{decr}| + \left(\frac{1}{2\log(2)}|\log(\epsilon)| + \kappa_{curv}\right)|\mathcal{S}_k^{curv}| + \frac{|\log(\epsilon)| + \log(\|g_0\|)}{\log(2)} + 1, \tag{4.53}$$

where

$$\kappa_n \stackrel{def}{=} \frac{1}{\log(2)}\log\left(\frac{L_H(1+\kappa_\theta)V_{\max}^3}{2\varsigma_1^2\sigma_{\min}} + \frac{2\sqrt{V_{\max}}\kappa_b}{\varsigma_1} + \sqrt{V_{\max}}\kappa_C\kappa_b\right) + \frac{\log(1+\kappa_\theta)}{\log(2)}, \tag{4.54}$$

$$\kappa_{curv} \stackrel{def}{=} \frac{1}{\log(2)}\log\left(\frac{L_H V_{\max}^2}{2\sigma_{\min}}\kappa_C^2\theta^2 + \frac{\kappa_B\kappa_C}{\sqrt{\sigma_{\min}}} + 1\right). \tag{4.55}$$

Proof. Let $k \in \mathcal{S}_k^{decr}$. Injecting the definition of $\mathcal{S}_k^{decr}$ (4.49) in (4.42), we obtain that

$$f(x_k) - f(x_{k+1}) \geq \eta_1\varsigma_{\min}(\sigma_k\|g_k\|)^{\frac{3}{2}}\left(\frac{-(2+\kappa_C)\kappa_b\sqrt{\kappa_a} + \sqrt{(2+\kappa_C)^2\kappa_b^2\kappa_a + \frac{V_{\max}^2}{\kappa_m}}}{L_H V_{\max}^2}\right)^2.$$

Taking the conjugate both at the denominator and numerator yields (4.52).

Let $k \in \mathcal{S}_k^{divgrad}$. Using the definition of κ_m in (4.51) and that of $\mathcal{S}_k^{divgrad}$ in (4.50) gives that

$$\|g_{k+1}\| < \frac{\sigma_k}{\kappa_m L_H}\frac{\|g_k\|}{2} \leq \frac{\sigma_k}{\gamma_3\max\left(\frac{\sigma_0}{L_H}, \frac{\varsigma_{\max}}{6(1-\eta_2)}\right)L_H}\frac{\|g_k\|}{2} \leq \frac{\|g_k\|}{2}, \tag{4.56}$$

where the last inequality results from the upper bound on σ_k in (4.32).

Successively using the fact that $\mathcal{S}_k = \mathcal{S}_k^{decr} \cup \mathcal{S}_k^{divgrad} \cup \mathcal{S}_k^{curv}$, the relationship between $\|g_{k+1}\|$ and $\|g_k\|$ in the three cases ((4.56), (4.41) and (4.39)), the fact that $\sigma_k \geq \sigma_{\min}$ in (4.41) and (4.39), we then deduce that

$$\begin{aligned}
\frac{\epsilon}{\|g_0\|} \leq \frac{\|g_k\|}{\|g_0\|} &= \prod_{i \in \mathcal{S}_k \setminus \{k\}} \frac{\|g_{i+1}\|}{\|g_i\|} \\
&= \prod_{i \in \mathcal{S}_k^{decr} \setminus \{k\}} \frac{\|g_{i+1}\|}{\|g_i\|} \prod_{i \in \mathcal{S}_k^{divgrad} \setminus \{k\}} \frac{\|g_{i+1}\|}{\|g_i\|} \prod_{i \in \mathcal{S}_k^{curv} \setminus \{k\}} \frac{\|g_{i+1}\|}{\|g_i\|} \\
&\leq \left[\left(\frac{L_H (1+\kappa_\theta) V_{\max}^3}{2\varsigma_1^2 \sigma_{\min}} + \frac{2\kappa_b \sqrt{V_{\max}}}{\varsigma_1} + \kappa_C \kappa_b \sqrt{V_{\max}} \right) (1+\kappa_\theta) \right]^{|\mathcal{S}_k^{decr} \setminus \{k\}|} \times \\
&\quad \frac{1}{2^{|\mathcal{S}_k^{divgrad} \setminus \{k\}|}} \times \left[\frac{L_H V_{\max}^2}{2\sigma_{\min}} \kappa_C^2 \theta^2 + \frac{\kappa_B \kappa_C}{\sqrt{\epsilon \sigma_{\min}}} + 1 \right]^{|\mathcal{S}_k^{curv} \setminus \{k\}|}.
\end{aligned}$$

Now $\varsigma_1 \leq 1$ and thus both terms in brackets are larger than one. Moreover, obviously, $|\mathcal{S}_k^{decr} \setminus \{k\}| \leq |\mathcal{S}_k^{decr}|$ and $|\mathcal{S}_k^{curv} \setminus \{k\}| \leq |\mathcal{S}_k^{curv}|$, so that

$$\begin{aligned}
\frac{2^{|\mathcal{S}_k^{divgrad} \setminus \{k\}|} \epsilon}{\|g_0\|} &\leq \left[\left(\frac{L_H (1+\kappa_\theta) V_{\max}^3}{2\varsigma_1^2 \sigma_{\min}} + \frac{2\kappa_b \sqrt{V_{\max}}}{\varsigma_1} + \sqrt{V_{\max}} \kappa_C \kappa_b \right) (1+\kappa_\theta) \right]^{|\mathcal{S}_k^{decr}|} \\
&\quad \times \left[\frac{L_H V_{\max}^2}{2\sigma_{\min}} \kappa_C^2 \theta^2 + \frac{\kappa_B \kappa_C}{\sqrt{\epsilon \sigma_{\min}}} + 1 \right]^{|\mathcal{S}_k^{curv}|}.
\end{aligned}$$

Taking logarithms gives that

$$\begin{aligned}
|\mathcal{S}_k^{divgrad} \setminus \{k\}| \log(2) \leq \log &\left[\left(\frac{L_H (1+\kappa_\theta) V_{\max}^3}{2\varsigma_1^2 \sigma_{\min}} + \frac{\kappa_b \sqrt{V_{\max}}}{\varsigma_1} + \kappa_C \kappa_b \sqrt{V_{\max}} \right) (1+\kappa_\theta) \right] |\mathcal{S}_k^{decr}| \\
&+ \log(\|g_0\|) + |\log(\epsilon)| + \log \left[\frac{L_H V_{\max}^2}{2\sigma_{\min}} \kappa_C^2 \theta^2 + \frac{\kappa_B \kappa_C}{\sqrt{\epsilon \sigma_{\min}}} + 1 \right] |\mathcal{S}_k^{curv}|.
\end{aligned}$$

We then obtain (4.53) with the values of κ_n and κ_{curv} stated in (4.54) and (4.55) by dividing this last inequality by $\log(2)$ and using the facts that $|\mathcal{S}_k^{divgrad} \setminus \{k\}| \geq |\mathcal{S}_k^{divgrad}| - 1$ and $\frac{1}{\sqrt{\epsilon}} \geq 1$. □

Combining the previous lemmas, we are now able to state the complexity of the AN2C algorithm. Our theorem statement relies on the observation that the objective function is evaluated once per iteration, and its derivatives once per successful iteration.

Theorem 4.3.5 *Suppose that AS.1–AS.3 hold. Then the AN2C algorithm requires at most*

$$|\mathcal{S}_k| \leq \left(\kappa_\star + \frac{\kappa_{negdecr}}{2\log(2)}|\log(\epsilon)|\right)\epsilon^{-\frac{3}{2}} + \frac{|\log(\epsilon)| + \log(\|g_0\|)}{\log(2)} + 1$$

successful iterations and evaluations of the gradient and the Hessian and at most

$$\left(1 + \frac{|\log\gamma_1|}{\log\gamma_2}\right)\left[\left(\kappa_\star + \frac{\kappa_{negdecr}}{2\log(2)}|\log(\epsilon)|\right)\epsilon^{-\frac{3}{2}} + \frac{|\log(\epsilon)| + \log(\|g_0\|)}{\log(2)} + 1\right] + \frac{1}{\log\gamma_3}\log\left(\frac{\sigma_{\max}}{\sigma_0}\right)$$

evaluations of f to produce a vector x_ϵ such that $\|g(x_\epsilon)\| \leq \epsilon$, where $\kappa_\star$ is defined by

$$\kappa_\star \stackrel{def}{=} \kappa_{decr}(1 + \kappa_n) + \kappa_{negdecr}(1 + \kappa_{curv}), \tag{4.57}$$

with

$$\kappa_{decr} \stackrel{def}{=} \frac{\left(L_H\kappa_m(\sqrt{\kappa_a}\kappa_b(2+\kappa_C) + \sqrt{\kappa_a(\kappa_b(2+\kappa_C))^2 + \frac{V_{\max}^2}{\kappa_m}})\right)^2}{\eta_1\varsigma_{\min}\sigma_{\min}^{\frac{3}{2}}} \tag{4.58}$$

and

$$\kappa_{negdecr} \stackrel{def}{=} \frac{2(f(x_0) - f_{low})\sqrt{\sigma_{\max}}}{\eta_1\kappa_C^3\theta^3}, \tag{4.59}$$

and where κ_n and κ_{curv} are given by (4.54) *and* (4.55).

Proof. First note that we only need to prove an upper bound on $|\mathcal{S}_k^{decr}|$ and $|\mathcal{S}_k^{curv}|$ to derive a bound on $|\mathcal{S}_k|$ since

$$|\mathcal{S}_k| = |\mathcal{S}_k^{decr}| + |\mathcal{S}_k^{curv}| + |\mathcal{S}_k^{divgrad}| \tag{4.60}$$

and a bound on $|\mathcal{S}_k^{divgrad}|$ is given by (4.53). We start by proving an upper bound on $|\mathcal{S}_k^{curv}|$. Using AS.2, the lower bound on the decrease of the function values (4.38) and that $\sigma_k \leq \sigma_{\max}$ as stated in Lemma 4.3.1, we derive that, for $k \in \mathcal{S}_k^{curv}$,

$$f(x_0) - f_{\text{low}} \geq \sum_{i\in\mathcal{S}_k} f(x_i) - f(x_{i+1}) \geq \sum_{i\in\mathcal{S}_k^{curv}} f(x_i) - f(x_{i+1}) \geq |\mathcal{S}_k^{curv}|\,\frac{\eta_1\kappa_C^3\theta^3}{2\sqrt{\sigma_{\max}}}\,\epsilon^{\frac{3}{2}}$$

and hence that

$$|\mathcal{S}_k^{curv}| \leq \frac{2(f(x_0) - f_{\text{low}})\sqrt{\sigma_{\max}}}{\eta_1\kappa_C^3\theta^3}\,\epsilon^{-\frac{3}{2}} = \kappa_{\text{negdecr}}\,\epsilon^{-\frac{3}{2}}. \tag{4.61}$$

Similarly for $k \in \mathcal{S}_k^{decr}$, using AS.2, (4.52), the fact that $\sigma_k \geq \sigma_{\min}$ and $\|g_k\| \geq \epsilon$ before

termination yields that

$$f(x_0)-f_{\low} \geq \sum_{i\in\mathcal{S}_k^{decr}} f(x_i)-f(x_{i+1}) \geq \frac{|\mathcal{S}_k^{decr}|\eta_1 \varsigma_{\min}(\sigma_{\min}\epsilon)^{\frac{3}{2}}}{\left(L_H\kappa_m(\sqrt{\kappa_a}(2+\kappa_C)\kappa_b + \sqrt{\kappa_a(\kappa_b(2+\kappa_C))^2 + \frac{V_{\max}^2}{\kappa_m}})\right)^2}$$

where κ_m is defined in (4.51). Rearranging the last inequality yields that

$$|\mathcal{S}_k^{decr}| \leq \frac{\left(L_H\kappa_m(\sqrt{\kappa_a}\kappa_b(2+\kappa_C) + \sqrt{\kappa_a(\kappa_b(2+\kappa_C))^2 + \frac{V_{\max}^2}{\kappa_m}})\right)^2}{\eta_1\varsigma_{\min}\sigma_{\min}^{\frac{3}{2}}}\epsilon^{-\frac{3}{2}} = \kappa_{\decr}\,\epsilon^{-\frac{3}{2}}. \tag{4.62}$$

Combining now (4.61) and (4.62) with the upper-bound (4.53) on $|\mathcal{S}_k^{divgrad}|$, we deduce that

$$|\mathcal{S}_k^{divgrad}| \leq \kappa_n\kappa_{\decr}\epsilon^{-\frac{3}{2}} + \left(\frac{|\log(\epsilon)|}{2\log(2)} + \kappa_{curv}\right)\kappa_{\negdecr}\epsilon^{-\frac{3}{2}} + \frac{|\log(\epsilon)| + \log(\|g_0\|)}{\log(2)} + 1. \tag{4.63}$$

By summing equations (4.61), (4.62), and (4.63) to bound $|\mathcal{S}_k|$ in (4.61), while also isolating the terms based on their different orders with respect to ϵ, we obtain that

$$|\mathcal{S}_k| \leq \left(\kappa_\star + \frac{\kappa_{\negdecr}}{2\log(2)}|\log(\epsilon)|\right)\epsilon^{-\frac{3}{2}} + \frac{|\log(\epsilon)| + \log(\|g_0\|)}{\log(2)} + 1, \tag{4.64}$$

where $\kappa_\star$ is defined in (4.57), thus proving the first part of the theorem. The second part is then deduced from (4.64) combined with Lemma 4.2.1. □

Regrouping all the problem's dependent constant of the last theorem and keeping the worst dependency w.r.t ϵ, we derive a $\mathcal{O}\left(|\log(\epsilon)|\epsilon^{-3/2}\right)$ complexity order in ϵ that only differs by the factor $|\log(\epsilon)|$ from the optimal order for nonconvex second-order methods [52], a factor which is typically small for practical values of ϵ. The AN2C algorithm thus enjoys a better complexity order than that of past hybrid algorithms [71, 145, 100] for which the order is $\mathcal{O}\left(\epsilon^{-2}\right)$. However, it is marginally worse than that of the more complex second-order linesearch of [184] which attains the optimal order. Moreover, we see in the proof of Theorem 4.3.5 that the $|\log\epsilon|$ term appears because of (4.53) and (4.61) and we may hope that the number of s_k^{curv} iterations is typically much less than its worst-case $\mathcal{O}\left(\epsilon^{-3/2}\right)$ in practice. The trust-region algorithm of [75] has the same total complexity as AN2C although their method requires only $\mathcal{O}\left(\epsilon^{-3/2}\right)$ gradient and Hessian calls whereas our algorithm suffers from an additional $|\log(\epsilon)|$ term.

4.4 Finding second-order critical points

Can the AN2C algorithm be strengthened to ensure it will compute second-order critical points? We show in this section under the same assumptions as that used for its first-order analysis that approximate second order points can be reached.

The resulting modified algorithm, which we call SOAN2C (for Second-Order AN2C) makes extensive use of AN2C, and is detailed on the following page.

Algorithm 4.4.1: Second-Order Adaptive Newton with Negative Curvature (SOAN2C)

Step 0: Initialization Identical to AN2C[Step 0] with $\epsilon \in (0,1]$ now replaced by $\epsilon_1 \in (0,1]$ and $\epsilon_2 \in (0,1]$.

Step 1: Compute current derivatives Evaluate g_k and H_k. Terminate if

$$\|g_k\| \leq \epsilon_1 \ \text{ and } \ \lambda_{\min}(H_k) \geq -\epsilon_2. \tag{4.65}$$

Step 2: Compute subspace derivatives Form $\widehat{g}_k$ and $\widehat{H}_k$ as in AN2C[Step 2].

Step 3: Step calculation If $\|g_k\| > \epsilon_1$,

$$s_k = s_k^{fo} = \mathsf{RegStep}(\, \widehat{g}_k, \widehat{H}_k, V_p, \|g_k\|, \sigma_k, \kappa_a, \kappa_b, \kappa_\theta, \varsigma_1, \varsigma_2\,), \quad \text{(Optional)}. \tag{4.66}$$

If s_k^{fo} has been successfully defined, go to Step 4. Else, compute

$$s_k = s_k^{fo} = \mathsf{NewtonEigenStep}(\, \widehat{g}_k, \widehat{H}_k, V_p, \|g_k\|, \sigma_k, \kappa_C, \kappa_b, \kappa_\theta, \varsigma_3, \theta\,). \tag{4.67}$$

Else ($\|g_k\| \leq \epsilon_1$), compute u_k such that

$$g_k^T u_k \leq 0, \ \|u_k\| = 1 \ \text{ and } \ H_k u_k = \lambda_{\min}(H_k) u_k, \tag{4.68}$$

and set

$$s_k = s_k^{so} \stackrel{\text{def}}{=} \frac{-\lambda_{\min}(H_k)}{\sigma_k} u_k. \tag{4.69}$$

Step 4: Acceptance ratio computation Identical to AN2C[Step 5].

Step 5: Regularization parameter update Identical to AN2C[Step 6].

Prior to reaching an approximate first-order point, we utilize only the RegStep and NewtonEigenStep subroutines to generate tentative steps, hence the 'fo' (first-order) superscripts in (4.66) and (4.67). Similar to Section 4.2, AS.0 is necessary to obtain a valid step when NewtonEigenStep is invoked. Once an approximate first-order point is reached, further progress towards second-order stationarity is obtained by exploiting the negative-curvature direction (4.68)-(4.69), thereby justifying the 'so' (second-order) superscript. Note that we make use of the full Hessian matrix as exact negative curvature is required in order to check if approximate second-order has been reached.

An upper bound on the evaluation complexity of the SOAN2C algorithm is given by the following theorem.

Theorem 4.4.1 *Suppose that AS.1–AS.3 hold. Then the* SOAN2C *algorithm requires at most*

$$|\mathcal{S}_k| \leq \kappa_\star \epsilon_1^{-\frac{3}{2}} + \kappa_{so}\epsilon_2^{-3} + \frac{|\log(\epsilon_1)|}{2\log(2)}\kappa_{negdecr}\epsilon_1^{-\frac{3}{2}} + \left(\frac{|\log(\epsilon_1)| + \log(\kappa_{gpi})}{\log(2)} + 1\right)(\kappa_{so}\epsilon_2^{-3} + 1)$$

successful iterations and evaluations of the gradient and the Hessian and at most

$$\left(1 + \frac{|\log\gamma_1|}{\log\gamma_2}\right)\left[\kappa_\star \epsilon_1^{-\frac{3}{2}} + \kappa_{so}\epsilon_2^{-3} + \frac{|\log(\epsilon_1)|}{2\log(2)}\kappa_{negdecr}\epsilon_1^{-\frac{3}{2}} + \left(\frac{|\log(\epsilon_1)| + \log(\kappa_{gpi})}{\log(2)} + 1\right)(\kappa_{so}\epsilon_2^{-3} + 1)\right] + \frac{1}{\log\gamma_3}\log\left(\frac{\sigma_{\max}}{\sigma_0}\right)$$

evaluations of f to produce a vector x_ϵ such that $\|g(x_\epsilon)\| \leq \epsilon_1$ and $\lambda_{\min}(H_{x_\epsilon}) \geq -\epsilon_2$, where

$$\kappa_{so} \stackrel{def}{=} \frac{2\sigma_{\max}^2(f(x_0) - f_{low})}{\eta_1} \tag{4.70}$$

κ_{gpi} is defined in (4.A.5) and $\kappa_\star$, $\kappa_{negdecr}$ and $\sigma_{\max}$ (defined by (4.57), (4.59) and (4.32), respectively) depend solely on the problem .

As for Theorem 4.3.5, the bound, in which the ϵ_2^{-3} term is likely to dominate, differs from standard one for second-order algorithms seeking second-order points (in $\mathcal{O}(\max(\epsilon_1^{-3/2}, \epsilon_2^{-3}))$ [57, Theorems 3.3.9 and 3.4.6] by a (modest) factor $|\log(\epsilon_1)|$.

To prove Theorem 4.4.1, we need to take two main issues into account. The first is that, because the step may be computed using (4.66), (4.67) but also (4.69), we need to complete the partition of $|\mathcal{S}_k|$ by introducing subsets relevant to this new type of steps. The second is clearly that negative curvature information must be exploited in order to guarantee a sufficient decrease of the objective function when it is discovered close to a first-order critical point. This leads to a development which broadly follows the lines of Section 4.3, extending the proofs when necessary to handle the more complicated situation. The details of this development are given in appendix.

4.5 Choosing the subspace

In practice, the algorithm crucially depends on how one chooses the matrix V_p spanning the iteration-dependent subspace, and we discuss two options. Each of the choices presented below can be included in both AN2C and SOAN2C, defined in Section 4.2 and Section 4.4, respectively. For conciseness, we only consider AN2C.

4.5.1 A full-space variant

A simple choice of V_p is to consider $V_p \stackrel{\text{def}}{=} I_n$, that is the subspace is in fact the whole space. We note that, in this case, conditions (4.9) or (4.13) automatically hold. We define two variants in this context. The first is called AN2CER (for AN2C Exact using RegStep) exploits

the RegStep algorithm in order to limit the need of possibly costly second-order information. The second, potentially more costly, is called AN2CE and does not use the optional RegStep algorithm, therefore making no attempt to avoid eigenvalue computations.

These variants may be useful for problems in which systems (4.6) and (4.11) may effectively be solved (for instance using Cholesky factorizations). As we will see below, they require on average a single such solution/factorization per iteration. AN2CER and AN2CE may thus be attractive in the large class of applications for which off-the-shelf linear solvers are available. The computation of $\lambda_{\min}(H_k)$ also needs to be feasible but, due to Algorithm RegStep, this occurs only rarely in AN2CER.

4.5.2 A Krylov variant

When the dimension of the problem grows and factorizations become impractical, one can turn to exploiting Krylov subspaces, as we now show. The resulting algorithmic variant will be called AN2CK, where K stands here for Krylov, and is obtained by replacing Steps 3 and 4 of the AN2C algorithm by Algorithm AN2CKStep on the next page. In this variant, the subspace generation and step computation are combined in order to best exploit the structure of the resulting subproblem. As is common in Krylov-based methods, we assume the availability of a 'preconditioner', that is a positive-definite matrix M_k approximating the Hessian H_k in the sense that $M_k^{-1}H_k$ is close to the identity. For clarity, we ignore the iteration subscript k in what follows.

Each iteration of the AN2CKStep algorithm has a moderate cost (a few vector assignments, one matrix-vector product and –possibly– the computation of the smallest eigenvalue of a tridiagonal matrix, see [67] and the references therein for details). We observe that (4.71)-(4.72) amounts to using the standard preconditioned Lanczos process for building an orthonormal (in the $\langle \cdot, M\cdot \rangle$ inner product) basis V_p of successive Krylov subspaces generated by the preconditioned gradient and Hessian. We therefore build on existing theory for this process (see [68, Section 5.2], for instance). We note that the use of the full Lanczos basis V_p is only requested at the end of the process (in (4.77) and (4.79)). As a consequence two options are available for its detailed implementation: one can store the Lanczos basis vectors as the iterations proceed and use them at the end of the step computation, or one can forget them but re-run the necessary Lanczos process to re-generate them (as has been done in the GALAHAD library [104] for the GLTR and GLRT algorithms for trust-region and regularization subproblems, respectively). Obviously, V_p and T_p may be updated incrementally in (4.73) and (4.74). When updating T_p, it is also easy to check if it remains positive definite by recurring the pivots of its Cholesky factorization, which are given by

$$\pi_1 = \delta_1 \text{ and } \pi_p = \delta_p - \beta_p^2/\pi_{p-1} \quad (p > 1).$$

As long as π_p stays positive, it is thus unnecessary to compute $\lambda_{\min}(T_p)$ since $[-\lambda_{\min}(T_p)]_+$ is then identically zero in (4.75). Finally, should a preconditioner M be unavailable, setting $M = I_n$ is possible, in which case w_p and z_p can be dispensed of because they are identical to r_p and v_p, respectively.

We now verify that, as stated, Algorithm AN2CK is a correct instantiation of Algorithm AN2C (without the optional Step 3).

Algorithm 4.5.1: **AN2CKStep**$(\, g, H, \sigma, M, \kappa_C, \kappa_b, \theta \,)$

Step 0: Initialization Set $p = 1$, $r_1 = g$, $w_1 = M^{-1} r_1$, $\beta_1 = \sqrt{w_1^T r_1}$ and $z_0 = 0$.

Step 1: Form the orthonormal basis Compute

$$z_p = \frac{r_p}{\beta_p}, \quad v_p = \frac{w_p}{\beta_p}, \quad \delta_p = v_p^T H v_p, \tag{4.71}$$

$$r_{p+1} = H v_p - \delta_p z_p - \beta_p z_{p-1}, \quad w_{p+1} = M^{-1} r_{p+1}, \quad \beta_{p+1} = \sqrt{w_{p+1}^T r_{p+1}}, \tag{4.72}$$

and define

$$V_p = (v_1, v_2, \ldots, v_p) \in \mathbb{R}^{n \times p}. \tag{4.73}$$

Step 2: Newton step computation Form the subspace Hessian

$$T_p \stackrel{\text{def}}{=} V_p^T H V_p = \begin{pmatrix} \delta_1 & \beta_2 & & & \\ \beta_2 & \delta_2 & \beta_3 & & \\ & \ddots & \ddots & \ddots & \\ & & & \delta_{p-1} & \beta_p \\ & & & \beta_p & \delta_p \end{pmatrix} \tag{4.74}$$

and compute its minimum eigenvalue.

If $\lambda_{\min}(T_p) \leq -\kappa_C \sqrt{\sigma \|g\|}$, go to Step 4.

Otherwise, solve

$$\left(T_p + (\sqrt{\sigma \|g\|} + [-\lambda_{\min}(T_p)]_+) I_p \right) y_p = -\beta_1 e_1. \tag{4.75}$$

Step 3: Check global quality of the solution If

$$\sqrt{\beta_{p+1}^2 (e_p^T y_p)^2 + \|T_p y_p + \beta_1 e_1\|^2} \leq \kappa_b \|T_p y_p + \beta_1 e_1\|, \tag{4.76}$$

then return

$$s = s^{neig} = V_p y_p. \tag{4.77}$$

Else increment p by one and go back to Step 1.

Step 4: Eigenvector direction Compute u such that

$$e_1^T u \leq 0, \ \|u\| = 1, \ u^T T_p u \leq \theta \, \lambda_{\min}(T_p) \ \text{ and } \ u^T T_p u \leq \frac{\lambda_{\min}(T_p)^2}{\theta^2} \tag{4.78}$$

Return

$$s = s^{curv} = \theta \kappa_C \sqrt{\frac{\|g\|}{\sigma}} \, V_p u. \tag{4.79}$$

Theorem 4.5.1 *Suppose that*

$$\mu_1 \leq \lambda_{\min}(M) \;\; \textit{and} \;\; \lambda_{\max}(M) \leq \mu_2 \tag{4.80}$$

for some $\mu_2 \geq \mu_1 > 0$. Then the definitions and conditions (4.78), (4.76) *and* (4.75) *of Algorithm* AN2CKStep *are equivalent to* (4.14), (4.13) *(with κ_b redefined as $\max(1, \kappa_b \sqrt{\mu_2})$) and* (4.11) *of Algorithm 4.2.3, respectively. Moreover, AS.0 holds and* (4.74) *is valid.*

Proof. If Z_p is the matrix whose columns are $z_1, \ldots, z_p$, we deduce from (4.71) and (4.72) that

$$HV_p = Z_p T_p + \beta_{p+1} z_{p+1} e_p^T = M V_p T_p + \beta_{p+1} M v_{p+1} e_p^T. \tag{4.81}$$

Using that $V_p^T M v_{p+1} = 0$ yields (4.74). Note also that as $v_1 = \frac{w_1}{\beta_1} = M^{-1} z_1$ from (4.71) and $V_p^T M V_p = I_p$,

$$V_p^T g = \beta_1 V_p^T z_1 = \beta_1 V_p^T M v_1 = \beta_1 e_1. \tag{4.82}$$

The last identity with the fact that $T_p = V_p^T H V_p$ ensures that (4.78) and (4.75) are reformulations of (4.14) and (4.11). We now prove that (4.76) implies (4.13). Using (4.75), (4.82), (4.81), we obtain that

$$\begin{aligned} Hs + g &= HV_p y_p + \beta_1 M v_1 = HV_p y_p + \beta_1 M V_p e_1 \\ &= HV_p y_p - MV_p T_p y_p - (\sqrt{\sigma \|g\|} + [-\lambda_{\min}(T_p)]_+) M V_p y_p \\ &= \beta_{p+1} (e_p^T y_p) M v_{p+1} - (\sqrt{\sigma \|g\|} + [-\lambda_{\min}(T_p)]_+) M V_p y_p. \end{aligned}$$

Since $V_p^T M V_p = I_p$ and $V_p^T M v_{p+1} = 0$, we deduce, using (4.75) and (4.76), that

$$\begin{aligned} \|Hs + g\|^2 &\leq \lambda_{\max}(M) \, (Hs + g)^T M^{-1} (Hs + g) \\ &= \lambda_{\max}(M) \, [\beta_{p+1}^2 (e_p^T y_p)^2 + (\sqrt{\sigma \|g\|} + [-\lambda_{\min}(T_p)]_+)^2 \|y_p\|^2] \\ &= \lambda_{\max}(M) \, [\beta_{p+1}^2 (e_p^T y_p)^2 + \|T_p y_p + \beta_1 e_1\|^2] \\ &\leq \kappa_b^2 \, \lambda_{\max}(M) \|T_p y_p + \beta_1 e_1\|^2, \end{aligned}$$

and (4.13) follows with the redefined κ_b. We finally verify that AS.0 holds. Because

$$1 = \|M^{\frac{1}{2}} V_p\| \geq \lambda_{\min}(M^{\frac{1}{2}}) \|V_p\| = \sqrt{\lambda_{\min}(M)} \|V_p\|$$

(4.16) holds with $V_{\max} = 1/\sqrt{\lambda_{\min}(M)} \leq \mu_1^{-1/2}$, where we again used (4.80) to derive the last inequality. Moreover, given that $\kappa_b \geq 1$, termination necessarily occurs when $p = n$, $V_n^T M V_n = I_n$, V_n spans the whole space and $\beta_{p+1} = 0$ in (4.76). □

The optional Step 3 of Algorithm 4.2.1 is in fact implicitly contained in Algorithm 4.5.1 since convexity along the current step (condition (4.7)) is verified at each step of the Lanczos process by checking the positive-definiteness of T_p.

Returning now to the complete sequence of minimization iterates, we see that, whenever the AN2CK algorithm is used with iteration-dependent preconditioners $M_k \neq I_n$, Theo-

rems 4.3.5 and 4.4.1 remain valid provided (4.80) holds uniformly for all iterations.

4.6 Numerical illustration

We now illustrate the behaviour of our proposed algorithms on three sets of test problems from the freely available OPM collection[2] [110]. The first set contains 119 small-dimensional problems, the second contains 74 medium-size ones, while the third contains 59 "largish" ones. The list of problems and their dimensions are listed in Tables 1, 2 and 3 in the Thesis appendix.

4.6.1 Using the full-space variants

We use Matlab implementations of AN2CE and AN2CER where the involved linear systems are solved by using the Matlab sparse Cholesky factorization, and where we have set

$$\kappa_C = 10^3,\ \kappa_a = 50 \ \ (\text{AN2CE}) \text{ or } \ 100 \ \ (\text{AN2CER}) \ \ , \ \kappa_\theta = 1,\ \varsigma_1 = \tfrac{1}{2},\ \varsigma_2 = \varsigma_3 = 10^{-10}, \theta = 1$$

$$\sigma_0 = 1,\ \sigma_{\min} = 10^{-8},\ \gamma_1 = \tfrac{1}{2},\ \gamma_2 = \gamma_3 = 10,\ \eta_1 = 10^{-4} \ \text{ and } \ \eta_2 = 0.95.$$

The values of κ_C and κ_a were obtained from a hyper-parameter search[3] on the set of small problems. The values of ς_2 and ς_3 are given here for consistency, but are irrelevant since factorizations are used to solve the linear systems. Other parameters values are typical of regularization algorithms.

We compare AN2CE and AN2CER with implementations of the standard adaptive regularization AR2 and trust-region TR2M, two well-regarded methods. All these algorithms use quadratic approximations of the objective function (i.e. gradients and Hessians). The first three also use the same acceptance thresholds η_1 and η_2 and values of γ_1, γ_2 and γ_3. The TR2M methods shrinks the trust-region radius by a factor $\sqrt{10}$ and expands it by a factor 2 (see [57, Section 11.2] for a discussion of the coherence of these factors between trust-region and adaptive regularization methods). The authors are aware that further method-dependent tuning would possibly result in improved performance, but the values chosen here appear to work reasonably well for each method. The step computation is performed in AR2 following [57, page 67] or [29] using an (unpreconditioned) Lanczos approach while a standard Moré-Sorensen method[4] is used in TR2M (see [57, Chapter 9] for details). For AR2, the step computation is terminated as soon as

$$\|g_k + H_k s_k\| \leq \tfrac{1}{2}\theta_{sub}\sigma_k\|s_k\|^2 \tag{4.83}$$

which slightly differs from the test $\|\nabla_x^1 m_k(s_k)\| \leq \tfrac{1}{2}\theta_{sub}\sigma_k\|s_k\|^2$ used in [57, page 65] and [29] while maintaining the desired $\mathcal{O}(\epsilon^{-3/2})$ evaluation complexity bound (see [111] for a justification of (4.83) –including the fact that it more often allows the pure Newton step to be accepted– or [57, page 67]). The Moré-Sorensen iterations in TR2M are terminated as soon as $\|s_k\| \in [(1-\theta_{sub})\Delta_k, (1+\theta_{sub})\Delta_k]$, where, in both cases, $\theta_{sub} = 10^{-3}$ for $n \leq 100$ and 10^{-2} for $n > 100$. All experiments were run on a Dell Precison computer with Matlab 2022b.

[2] This collection is a subset of the CUTEest [106] collection where the test problems are described in Matlab.

[3] Covering the choice $\{10^{30}, 10^8, 10^5, 10^3, 10^2, 10\}$ for κ_C and $\{100, 50, 10\}$ for κ_a.

[4] Given that our version of AN2C uses matrix factorizations, it seems more natural to compare it with a Moré-Sorensen-based trust-region than to one using truncated conjugate gradients.

We discuss our experiments from the efficiency and reliability points of view. Efficiency is measured, in accordance with the complexity theory, in number of iterations (or, equivalently, function and possibly derivatives' evaluations): the fewer the more efficient the algorithm. In addition to presenting the now standard performance profiles [86] for our four algorithms in Figure 1, we follow [179, 115] and consider the derived "global" measure $\pi_{\texttt{algo}}$ to be $\frac{1}{10}$ of the area below the curve corresponding to `algo` in the performance profile, for abscissas in the interval $[1, 10]$. The larger this area and the closer $\pi_{\texttt{algo}}$ to one, the closer the curve to the left and top borders of the plot and the better the global performance.

When reporting reliability, we say that the run of an algorithmic variant on a specific test problem is successful if the gradient norm tolerance $\epsilon = 10^{-6}$ has been achieved in the allotted cpu-time (1h) and before the maximum number of iterations (5000) is reached. The $\rho_{\texttt{algo}}$ statistic denotes the percentage of successful runs taken on all problems in each of the three classes.

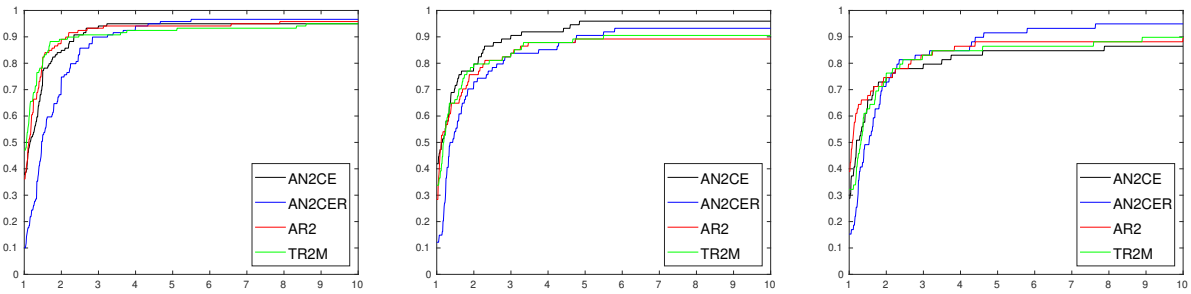

Figure 1: Full-space variants: iteration performance profiles for OPM problems (left: small, center: medium, right: largish). We report on the vertical axis the proportion of problems for which the number of iterations of each algorithm is at most a fraction (given by the horizontal axis) of the smallest across all algorithms (see [86]).

	small pbs.		medium pbs.		largish pbs.	
algo	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$
AN2CER	0.88	96.64	0.85	93.24	0.85	94.92
AN2CE	0.91	96.64	0.91	95.95	0.81	86.44
AR2	0.92	97.48	0.85	93.24	0.84	93.22
TR2M	0.91	94.96	0.86	93.24	0.83	91.53

Table 1: Efficiency and reliability statistics for the OPM problems (full-space variants).

Figure 1 and AN2CER is comparable to that of AR2 and TR2M for all problem sizes. They also indicate that AN2CER is somewhat slower iteration-wise than AR2 and TR2M , but AN2CE is very comparable. The fact that the computationally more expensive AN2CE is often faster than AN2CER in terms of iteration numbers is not surprising. Indeed, the regularization term in (4.11) becomes $\sqrt{\sigma_k \|g_k\|}$ in convex regions, recovering the analysis of [156, 82], whereas AN2CE regularizes the problem more strongly in (4.6) (by a factor 10 in our numerical settings) and therefore may further restricts the steplength. AN2CE may however be computationally

more intensive[5] than AN2CER. Which of the two algorithms is preferable in practice is likely to depend on the CPU cost of calculating the Hessian's smallest eigenvalue.

As expected, the call to NewtonEigenStep in AN2CER is typically performed on very few iterations (for less 6.4% of them for the small-problems testset) and, when used, results in a negative-curvature step (4.15) even more exceptionally (less than 1%). This means in particular that a single linear-system solve was necessary for approximately 93% of all iterations. The AN2CE variant of course called NewtonEigenStep at every iteration, but (4.15) was never actually used.

We also ran the SOAN2CE and SOAN2CER variants with $\epsilon_1 = 10^{-6}$ and $\epsilon_2 = 10^{-4}$, but their results are undistinguishable (for our test sets) from those obtained with AN2CE and AN2CER, except for a final eigenvalue analysis at the found approximate first-order point, which confirmed in all cases that the second-order condition (4.65) did also hold at this point. No step of the form (4.69) was ever taken in our runs, despite the fact that such steps are necessary in theory (think of starting the minimization at a first-order saddle point).

4.6.2 Using the Krylov-based variants

We ran two variants of the AN2CK algorithm on our three problem sets, which differ in how the vector u is chosen in (4.78). In the first, called AN2CKU, u is chosen as the eigenvector associated with the eigenvalue $\lambda_{\min}(T_p)$. In the second, called AN2CKYU, u is chosen as the sum of the current vector y_p plus a multiple of the eigenvector associated with $\lambda_{\min}(T_p)$ chosen to ensure that the last inequality in (4.78) holds as an equality. An hyper-parameter search on a subset of the medium-sized test set yielded the values

$$\kappa_C = 3, \quad \kappa_b = 50 \text{ and } \theta = \tfrac{1}{2}.$$

None of the tested methods used preconditioning (that is the choice $M = I_n$ was made throughout). The matrices V_p were stored explicitly.

We again compared these two variants with AR2 and with TR2K, an implementation of the trust-region close to TR2M, but in which the step is computed by minimizing the quadratic model in the intersection of the trust-region and the successive Krylov spaces until

$$\|g_k + H_k s_k\| \leq \tfrac{1}{10}\|g_k\|. \tag{4.84}$$

The results of our comparison (using the same metrics as in the previous subsection) are given in Figure 2 and Table 2.

	small pbs.		medium pbs.		largish pbs.	
algo	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$	$\pi_{\texttt{algo}}$	$\rho_{\texttt{algo}}$
AN2CKU	0.86	96.64	0.81	93.24	0.77	86.44
AN2CKYU	0.91	96.64	0.90	95.95	0.85	91.53
AR2	0.92	97.48	0.87	93.24	0.89	93.22
TR2K	0.94	96.64	0.85	87.84	0.77	84.75

Table 2: Efficiency and reliability statistics for the OPM problems (Krylov-space variants).

[5] Most failures of this algorithm on large problems occured because the time limit was reached.

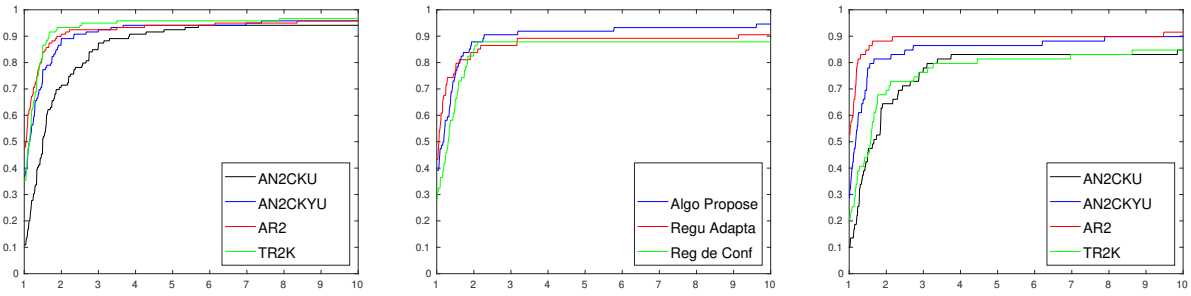

Figure 2: Krylov-space variants: iteration performance profiles for OPM problems (left: small, center: medium, right: largish). We report on the vertical axis the proportion of problems for which the number of iterations of each algorithm is at most a fraction (given by the horizontal axis) of the smallest across all algorithms (see [86]).

We observe that AN2CKU significantly trails the other variants and is in particular both less efficient and less reliable than AN2CKYU, which we explain by the fact that, should a negative curvature step occur, the former strategy does not exploit the decrease of the quadratic model already obtained by the "convex step" y_p. By contrast, AN2CKYU appears to be competitive with both AR2 and TR2K, irrespective of problem size.

For the AN2CKYU variant, the average ratio of the number of matrix-vector products divided by the product of the number of iterations and the problem size (a ratio which is one if every Lanczos process takes n iterations) is below 0.5 for small problems, below 0.15 for medium ones and below 0.03 for large ones. Negative curvature directions (4.79) are also used, for this variant, by 0.25% of the iterations for small problems, 0.23% of iterations for medium ones and never for large ones.

Finally, we also tested SOAN2CKU and SOAN2CKYU, the versions of AN2CKU and AN2CKYU which enforce second-order optimality. As for full-space methods, the results obtained are undistinguable from those for AN2CKU and AN2CKYU, except for a final eigenvalue analysis confirming the approximate second-order optimality of the computed solution.

These early results are encouraging but the authors are aware that only further experiments will allow a proper assessment of the method's true potential, both from the number of function/derivatives evaluations and CPU-usage points of view. Several further algorithmic developments within the new algorithms are also of interest, including a possibly better balance between NewtonEigenStep and RegStep in the full-space version, as well as refinements of the regularization parameter update (4.5), possibly in the spirit of [105].

4.7 Discussions

We have proposed AN2C and AN2CK, two second-order minimization methods for nonconvex problems that alternate, in an iteration dependent subspace, between Newton and negative-curvature directions. These methods differ from the more standard trust-region and adaptive-regularization techniques in that the involved step computation is free of further inner iterative processes and only requires the approximate solution of at most two (but typically one) linear systems per iteration. We have also proved that these algorithms require at most

$\mathcal{O}\left(|\log(\epsilon)|\epsilon^{-3/2}\right)$ iterations to obtain an ϵ-approximate first-order critical point. Our proof builds on some elements of [156, 82] for the convex case and arguments for adaptive regularization [29] and other nonconvex optimization methods [71, 184]. At each iteration, the algorithms either take an explicit Newton step or negative curvature when it is sufficiently large compared to the square root of the gradient. The norm of the residuals of the Newton step are adjusted dynamically and different types of solvers can be used to solve the linear systems, depending on how subspaces are chosen.

An extension of the algorithmic framework ensuring approximate second-order optimality has also been introduced, and we have proved that the resulting methods require at most $\mathcal{O}\left(|\log(\epsilon)|\epsilon^{-3}\right)$ iterations to achieve their goals.

A first set of numerical experiments with full-space variants AN2CE and AN2CER as well as Krylov-subspaces iterative ones AN2CKU and AN2CKYU indicates that they are very reliable and competitive with standard techniques in terms of number of iterations.

The reader may wonder why we haven't considered selecting iteration-dependent low-dimensional random subspaces, as has been advocated in [56, 190] for instance. The main reason is that using the Johnson-Lindenstrauss lemma (the basic tool is such an approach) is possible for defining a probabilistically accurate approximate gradient in the subspace, but, as far we know, this is problematic for the full Hessian matrix unless it is assumed to be of low rank. We could therefore attempt to follow the Cauchy-point-based analysis of [56, 190], and hopefully obtain a probabilistic complexity bound in $\mathcal{O}(\epsilon^{-2})$. However, we do not see at this point how to design a low-dimensional random-subspace algorithm with an $\mathcal{O}(|\log(\epsilon)|\epsilon^{-3/2})$ probabilistic complexity bound for minimizing functions with general (possibly full-rank) Hessians. A promising line is the use of the subspace embeddings proposed in both [66, 155], as they came with stronger guarantees and have already been successfully used to develop subspace ARC algorithm [191]. It would be also be interesting to extend the criteria (4.13) for the standard cubic regularization [50, 51] and higher-order tensor methods [29].

Appendices

4.A Proof of Theorem 4.4.1

As we noted in Section 4.4, the step in the SOAN2C algorithm may be computed using (4.66), (4.67) or (4.69). The notations defining the partition of $|\mathcal{S}_k|$ remain relevant, but we complete them by introducing

$$\mathcal{S}^{so} \stackrel{\rm def}{=} \mathcal{S} \cap \{s_k = s_k^{so}\}, \quad \mathcal{S}_k^{so} \stackrel{\rm def}{=} \mathcal{S}_k \cap \{s_k = s_k^{so}\}, \quad \mathcal{S}^{fo} \stackrel{\rm def}{=} \mathcal{S} \setminus \mathcal{S}^{so} \text{ and } \mathcal{S}_k^{fo} \stackrel{\rm def}{=} \mathcal{S}_k \setminus \mathcal{S}_k^{so}.$$

In addition, for $m \geq \ell \geq 0$, we define

$$\mathcal{S}_{\ell,m} \stackrel{\rm def}{=} \mathcal{S} \cap \{\ell, \ldots, m\}$$

and we naturally extend this notation using superscripts identifying the subsets of $\mathcal{S}_{\ell,m}$ corresponding to the different iteration types identified above. We also introduce two index sequences whose purpose is to keep track of when $s_k = s_k^{fo}$ (4.66)-(4.67) or $s_k = s_k^{so}$ (4.69) are used, in the sense that

$$s_k = s_k^{fo} \text{ for } k \in \bigcup_{i \geq 0, p_i \geq 0} \{p_i, \ldots, q_i - 1\} \text{ and } s_k = s_k^{so} \text{ for } k \in \bigcup_{i \geq 0} \{q_i, \ldots, p_{i+1} - 1\}.$$

Formally,

$$p_0 = \begin{cases} 0 & \text{if } \|g_0\| > \epsilon_1 \\ -1 & \text{if } \|g_0\| \leq \epsilon_1, \end{cases} \quad \text{and} \quad q_0 = \begin{cases} \inf\{k > 0 \mid \|g_k\| \leq \epsilon_1\} & \text{if } \|g_0\| > \epsilon_1 \\ 0 & \text{if } \|g_0\| \leq \epsilon_1. \end{cases} \tag{4.A.1}$$

Then

$$p_i \stackrel{\rm def}{=} \inf\{k > q_{i-1} \mid \|g_k\| > \epsilon_1\} \quad \text{and} \quad q_i \stackrel{\rm def}{=} \inf\{k > p_i \mid \|g_k\| \leq \epsilon_1\} \quad \text{for} \quad i \geq 1. \tag{4.A.2}$$

The following lemma states an important decrease property holding when (4.69) is used. We also verify that the bound on the regularization parameter derived in Section 4.3 still applies.

Lemma 4.A.1 *Suppose that AS.1 and AS.3 hold. Let $k \in \mathcal{S}^{so}$. Then*

$$-g_k^T s_k - \frac{1}{2} s_k^T H_k s_k \geq \frac{1}{2}\sigma_k \|s_k\|^3. \tag{4.A.3}$$

Moreover, the upper bound (4.32) *still holds for all $k \geq 0$.*

Proof. We obtain from (4.68) and (4.69) that

$$g_k^T s_k^{so} + \frac{1}{2}(s_k^{so})^T H_k s_k^{so} \leq \frac{1}{2}\|s_k^{so}\|^2 u_k^T H_k u_k = \frac{1}{2}\|s_k^{so}\|^2 \lambda_{\min}(H_k) \leq -\frac{1}{2}\sigma_k \|s_k^{so}\|^3,$$

which gives (4.A.3). As in Lemma 4.3.1, we now use AS.3, the standard Lipschitz error bound for the function (see [55, Lemma 2.1]) and (4.A.3) to deduce that

$$1 - \rho_k = \frac{f(x_k + s_k) - f(x_k) - g_k^T s_k - \frac{1}{2} s_k^T H_k s_k}{-g_k^T s_k - \frac{1}{2} s_k^T H_k s_k} \leq \frac{L_H \|s_k^{so}\|^3}{6(\frac{1}{2}\sigma_k \|s_k^{so}\|^3)} = \frac{L_H}{3\sigma_k}.$$

Thus, if $\sigma_k \geq \frac{L_H}{3(1-\eta_2)}$, we have that $\rho_k \geq \eta_2$ and k is a successful iteration. We may then use the argument of Lemma 4.3.1 and the fact that $\varsigma_{\max}$ introduced in (4.33) is larger than two as $V_{\max} \geq 1$. Therefore, we deduce that (4.32) also holds for the SOAN2C algorithm. □

We now prove an analogue of Lemma 4.3.1, now using the negative-curvature step as described in (4.68)-(4.69). We also bound the sequence of $\|g_{p_i}\|$.

Lemma 4.A.2 *Suppose that AS.1, AS.3 and AS.4 hold. Then, for $k \in \mathcal{S}^{so}$,*

$$f(x_k) - f(x_{k+1}) \ \geq \frac{\eta_1}{2\sigma_{\max}^2}\epsilon_2^3. \tag{4.A.4}$$

We also have that

$$\|g_{p_i}\| \leq \kappa_{gpi} \stackrel{def}{=} \max\left(\|g_0\|, \left(\frac{L_H \kappa_B^2}{2\sigma_{\min}^2} + \frac{\kappa_B^2}{\sigma_{\min}} + 1\right)\right), \tag{4.A.5}$$

for all $p_i \geq 0$ as defined in (4.A.1)-(4.A.2).

Proof. Let $k \in \mathcal{S}^{so}$. From (4.4) and (4.A.3), we obtain that

$$f(x_k) - f(x_{k+1}) \geq \eta_1 \left(-g_k^T s_k - \frac{1}{2} s_k^T H_k s_k\right) \geq \frac{\eta_1}{2}\sigma_k \|s_k^{so}\|^3.$$

Using now that $\|s_k^{so}\|^3 = \frac{|\lambda_{\min}(H_k)|^3}{\sigma_k^3}$ (see (4.69)) in the previous inequality gives that

$$f(x_k) - f(x_{k+1}) \geq \frac{\eta_1}{2{\sigma_k}^2}|\lambda_{\min}(H_k)|^3.$$

Now $|\lambda_{\min}(H_k)| \geq \epsilon_2$ when s_k^{so} is computed and $\sigma_k \leq \sigma_{\max}$ by Lemma 4.A.1, from which (4.A.4) follows. Observe now that (4.A.5) trivially holds if $p_i = p_0 = 0$. Consider now $p_i > 0$. From the definition of p_i and q_i in (4.A.2), we see that $p_i - 1 \in \mathcal{S}^{so}$. Using the Lipschitz error bound for the gradient ([55, Lemma 2.1]), the triangular inequality, (4.68), (4.69), (4.31) (resulting from AS.4), we obtain that

$$\begin{aligned}
\|g_{p_i}\| &\leq \|g_{p_i} - g_{p_i-1} - H_{p_i-1}s_{p_i-1}^{so}\| + \|H_{p_i-1}s_{p_i-1}^{so} + g_{p_i-1}\| \\
&\leq \frac{L_H}{2}\|s_{p_i-1}^{so}\|^2 + \|g_{p_i-1}\| + \|H_{p_i-1}s_{p_i-1}^{so}\| \\
&\leq \frac{L_H|\lambda_{\min}(H_{p_i-1})|^2}{2\sigma_{p_i-1}^2} + \|g_{p_i-1}\| + \frac{|\lambda_{\min}(H_{p_i-1})|\|H_{p_i-1}u_{p_i-1}\|}{\sigma_{p_i-1}} \\
&\leq \frac{L_H|\lambda_{\min}(H_{p_i-1})|^2}{2\sigma_{p_i-1}^2} + \|g_{p_i-1}\| + \frac{|\lambda_{\min}(H_{p_i-1})|^2}{\sigma_{p_i-1}} \\
&= \frac{L_H(-\lambda_{\min}(H_{p_i-1}))^2}{2\sigma_{p_i-1}^2} + \|g_{p_i-1}\| + \frac{(-\lambda_{\min}(H_{p_i-1}))^2}{\sigma_{p_i-1}} \\
&\leq \frac{L_H\kappa_B^2}{2\sigma_{p_i-1}^2} + \|g_{p_i-1}\| + \frac{\kappa_B^2}{\sigma_{p_i-1}}.
\end{aligned}$$

But $\|g_{p_i-1}\| \leq \epsilon_1 \leq 1$ since $p_i - 1 \in \mathcal{S}^{so}$ and $\sigma_k \geq \sigma_{\min}$ for all $k \geq 0$, which then implies (4.A.5). □

In addition to this lemma, all properties of the different steps derived in Section 3 remain valid because these steps are only computed for $\|g_k\| > \epsilon_1$. In particular, (4.39) still applies with $\epsilon = \epsilon_1$. However, (4.53) in Lemma 4.3.4 may no longer hold because its proof relies on the fact that $\|g_k\| \geq \epsilon_1$, which is no longer true. The purpose of the next lemma is to provide an analogue of (4.53) for the case where SOAN2C is used.

Lemma 4.A.3 *Suppose that AS.1, AS.3 and AS.4 hold and the SOAN2C algorithm is used. Consider the partition of $\mathcal{S}_k^{neig} \cup \mathcal{S}_k^{def}$ into $\mathcal{S}_k^{decr} \cup \mathcal{S}_k^{divgrad}$ defined in Lemma 4.3.4 with the same κ_m (defined in (4.51)). Then (4.52) holds for all $k \in \mathcal{S}_k^{decr}$. Moreover,*

$$\begin{aligned}
|\mathcal{S}_k^{divgrad}| \leq \kappa_n|\mathcal{S}_k^{decr}| &+ \left(\frac{1}{2\log(2)}|\log(\epsilon_1)| + \kappa_{curv}\right)|\mathcal{S}_k^{curv}| \\
&+ \left(\frac{|\log(\epsilon_1)| + \log(\kappa_{gpi})}{\log(2)} + 1\right)(|\mathcal{S}_k^{so}| + 1) \qquad (4.A.6)
\end{aligned}$$

where κ_n and κ_{curv} are defined in (4.54) and (4.55) and κ_{gpi} is given by (4.A.5).

Proof. The proof of (4.52) is identical to that used in Lemma 4.3.4. Moreover, we

still obtain (4.56) for $k \in \mathcal{S}_k^{divgrad}$, because the definition of κ_m in (4.51) is unchanged and Lemma 4.A.1 ensures that (4.32) continues to hold for the SOAN2C algorithm.

We now prove (4.A.6). If $\mathcal{S}_k^{fo}$ is empty, then so is its subset $\mathcal{S}_k^{divgrad}$ and (4.A.6) trivially holds. If $\mathcal{S}_k^{fo}$ is not empty, we see from the definitions (4.A.1)-(4.A.2) that, for some $m \geq 0$ depending on k,

$$\mathcal{S}_k^{fo} = \{0, \ldots, k\} \cap \{\|g_k\| > \epsilon_1\} = \left(\bigcup_{i=0, p_i \geq 0}^{m-1} \{p_i, \ldots, q_i - 1\} \right) \cup \{p_m, \ldots, k\}. \tag{4.A.7}$$

Note that the last set in this union is empty unless $k \in \mathcal{S}^{fo}$, in which case $p_m \geq 0$. Suppose first that the set of indices corresponding to the union in brackets is non-empty and let i be an index in this set. Moreover, suppose also that $p_i < q_i - 1$. Using (4.A.5) and the facts that $\|g_{q_i-1}\| > \epsilon_1$, that the gradient only changes at successful iterations and that $\mathcal{S}_{p_i,q_i-2} = \mathcal{S}_{p_i,q_i-2}^{curv} \cup \mathcal{S}_{p_i,q_i-2}^{divgrad} \cup \mathcal{S}_{p_i,q_i-2}^{decr}$, we now derive that

$$\begin{aligned}
\frac{\epsilon_1}{\kappa_{gpi}} &\leq \frac{\|g_{q_i-1}\|}{\|g_{p_i}\|} = \prod_{j=p_i}^{q_i-2} \frac{\|g_{j+1}\|}{\|g_j\|} = \prod_{j \in \mathcal{S}_{p_i,q_i-2}} \frac{\|g_{j+1}\|}{\|g_j\|} \\
&= \prod_{j \in \mathcal{S}_{p_i,q_i-2}^{decr}} \frac{\|g_{j+1}\|}{\|g_j\|} \prod_{j \in \mathcal{S}_{p_i,q_i-2}^{curv}} \frac{\|g_{j+1}\|}{\|g_j\|} \prod_{j \in \mathcal{S}_{p_i,q_i-2}^{divgrad}} \frac{\|g_{j+1}\|}{\|g_j\|} \\
&\leq \left(\left(\frac{L_H(1+\kappa_\theta)V_{\max}^3}{2\varsigma_1^2 \sigma_{\min}} + \frac{2\kappa_b\sqrt{V_{\max}}}{\varsigma_1} + \kappa_C\kappa_b\sqrt{V_{\max}} \right)(1+\kappa_\theta) \right)^{|\mathcal{S}_{p_i,q_i-2}^{decr}|} \times \\
&\quad \frac{1}{2^{|\mathcal{S}_{p_i,q_i-2}^{divgrad}|}} \times \left(\frac{L_H V_{\max}^2}{2\sigma_{\min}} \kappa_C^2 \theta^2 x + \frac{\theta^2 \kappa_B \kappa_C}{\sqrt{\epsilon_1 \sigma_{\min}}} + 1 \right)^{|\mathcal{S}_{p_i,q_i-2}^{curv}|}
\end{aligned}$$

where we used (4.41), (4.39) and (4.56) to derive the last inequality. Rearranging terms, taking the log, using the inequality $|\mathcal{S}_{p_i,q_i-2}^{divgrad}| \geq |\mathcal{S}_{p_i,q_i-1}^{divgrad}| - 1$ and dividing by $\log(2)$ then gives that

$$(|\mathcal{S}_{p_i,q_i-1}^{divgrad}| - 1) + \frac{\log(\epsilon_1) - \log(\kappa_{gpi})}{\log(2)} \leq \kappa_n |\mathcal{S}_{p_i,q_i-2}^{decr}| + \left(\frac{|\log(\epsilon_1)|}{2\log(2)} + \kappa_{curv} \right) |\mathcal{S}_{p_i,q_i-2}^{curv}|$$

with κ_n and κ_{curv} given by (4.54) and (4.55). Further rearranging this inequality and using the fact that $|\mathcal{S}_{p_i,q_i-2}| \leq |\mathcal{S}_{p_i,q_i-1}|$ for the different types of step, we obtain that

$$|\mathcal{S}_{p_i,q_i-1}^{divgrad}| \leq \kappa_n |\mathcal{S}_{p_i,q_i-1}^{decr}| + \left(\frac{|\log(\epsilon_1)|}{2\log(2)} + \kappa_{curv} \right) |\mathcal{S}_{p_i,q_i-1}^{curv}| + \frac{|\log(\epsilon_1)| + \log(\kappa_{gpi})}{\log(2)} + 1. \tag{4.A.8}$$

If now $p_i = q_i - 1$, then clearly $|\mathcal{S}_{p_i,q_i-1}^{divgrad}| \leq 1$ and (4.A.8) also holds. Using the same reasoning when $\{p_m, \ldots, k\}$ is non-empty, we derive that,

$$|\mathcal{S}_{p_m,k}^{divgrad}| \leq \kappa_n |\mathcal{S}_{p_m,k}^{decr}| + \left(\frac{|\log(\epsilon_1)|}{2\log(2)} + \kappa_{curv} \right) |\mathcal{S}_{p_m,k}^{curv}| + \frac{|\log(\epsilon_1)| + \log(\kappa_{gpi})}{\log(2)} + 1, \tag{4.A.9}$$

and this inequality also holds if $\{p_m, \ldots, k\} = \emptyset$ since $\mathcal{S}_{p_m,k}^{divgrad} \subseteq \{p_m, \ldots, k\}$. Adding now

(4.A.8) for $i \in \{0, \ldots, m\}$ and (4.A.9) to take (4.A.7) into account gives that

$$|\mathcal{S}_k^{divgrad}| \leq \kappa_n |\mathcal{S}_k^{decr}| + \left(\frac{|\log(\epsilon_1)|}{2\log(2)} + \kappa_{curv} \right) |\mathcal{S}_k^{curv}| + \left(\frac{|\log(\epsilon_1)| + \log(\kappa_{gpi})}{\log(2)} + 1 \right) (m+1).$$

As (4.A.7) divides $\mathcal{S}_k^{fo}$ into $m+1$ consecutive sequences, these sequences are then separated by at least a second-order step, so that $m \leq \mathcal{S}_k^{so}$ and (4.A.6) follows. □

Equipped with this last lemma and the results of Sections 4.2 and 4.3, we may finally establish the worst-case iteration/evaluation complexity of the SOAN2C algorithm and prove Theorem 4.4.1 itself.

Proof. Note that the bounds (4.61) and (4.62) derived in the proof of Theorem 4.3.5 are still valid because they only cover steps computed using AN2C, so that we now need to focus on bounding $\mathcal{S}_k^{so}$. Using AS.2 and the lower bound on the decrease of the function values (4.A.4), we derive that, for $k \in \mathcal{S}^{so}$,

$$f(x_0) - f_{\rm low} \geq \sum_{i \in \mathcal{S}_k} f(x_i) - f(x_{i+1}) \geq \sum_{i \in \mathcal{S}_k^{so}} f(x_i) - f(x_{i+1}) \geq |\mathcal{S}_k^{so}| \, \frac{\eta_1}{2\sigma_{\max}^2} \, \epsilon_2^3,$$

and therefore that

$$|\mathcal{S}_k^{so}| \leq \frac{2\sigma_{\max}^2 (f(x_0) - f_{\rm low})}{\eta_1} \, \epsilon_2^{-3} = \kappa_{\rm so} \epsilon_2^{-3}. \tag{4.A.10}$$

Injecting now (4.A.10), (4.62) and (4.61) in the bound (4.A.6) on $\mathcal{S}_k^{divgrad}$ yields that

$$\begin{aligned} |\mathcal{S}_k^{divgrad}| \leq \kappa_n \kappa_{\rm decr} \epsilon_1^{\frac{-3}{2}} &+ \left(\frac{|\log(\epsilon_1)|}{2\log(2)} + \kappa_{curv} \right) \kappa_{\rm negdecr} \epsilon_1^{\frac{-3}{2}} \\ &+ \left(\frac{|\log(\epsilon_1)| + \log(\kappa_{gpi})}{\log(2)} + 1 \right) (\kappa_{\rm so} \epsilon_2^{-3} + 1). \end{aligned}$$

Combining the last inequality with (4.A.10), (4.62) and (4.61) in $|\mathcal{S}_k| = |\mathcal{S}_k^{divgrad}| + |\mathcal{S}_k^{curv}| + |\mathcal{S}_k^{so}| + |\mathcal{S}_k^{decr}|$ and the definition of (4.57) gives that

$$|\mathcal{S}_k| \leq \kappa_\star \epsilon_1^{\frac{-3}{2}} + \kappa_{\rm so} \epsilon_2^{-3} + \frac{|\log(\epsilon_1)|}{2\log(2)} \kappa_{\rm negdecr} \epsilon_1^{\frac{-3}{2}} + \left(\frac{|\log(\epsilon_1)| + \log(\kappa_{gpi})}{\log(2)} + 1 \right) (\kappa_{\rm so} \epsilon_2^{-3} + 1).$$

This proves the first part of the theorem. The second part follows from the last inequality and Lemma 4.2.1. □

The factor $|\log(\epsilon_1)|$ by which the bound of Theorem 4.4.1 differs from $\mathcal{O}(\max(\epsilon_1^{-3/2}, \epsilon_2^{-3})$ occurs as a consequence of (4.A.6), (4.A.10) and (4.61) and one expects that, in practice, (4.A.10) is smaller than $\mathcal{O}\left(\epsilon_2^{-3}\right)$ so that Newton steps are taken most often.

Chapter 5

Hölder Gradient Descent and Adaptive Regularization in Banach Spaces

Chapter Abstract

This chapter considers optimization of nonconvex functionals in smooth infinite dimensional spaces. It is first proved that functionals in a class containing multivariate polynomials augmented with a sufficiently smooth regularization can be minimized by a simple linesearch-based algorithm. Sufficient smoothness depends on gradients satisfying a novel two-terms generalized Lipschitz condition. A first-order adaptive regularization method applicable to functionals with β-Hölder continuous derivatives is then proposed, that uses the linesearch approach to compute a suitable trial step. It is shown to find an ϵ-approximate first-order point in at most $\mathcal{O}(\epsilon^{-\frac{p+\beta}{p+\beta-1}})$ evaluations of the functional and its first p derivatives.

Reference: This chapter is based on a publication in Optimization Methods and Softwares [118].

5.1 Introduction

Evaluation complexity results obtained for AR methods and nonconvex problems have been obtained, to the best of the author's knowledge, in the context of $\mathbb{R}^n$. It is the purpose of this chapter to show that this need not be the case, and that evaluation complexity bounds for computing approximate first-order critical point can be derived in infinite-dimensional Banach spaces.

The main motivation for this generalization is twofold. Our first aim is to cover a number of infinite-dimensional applications in optimal control and variational analysis, and show that adaptive regularization methods do make sense in that context. Indeed, our developement covers optimization problems in $L^p(\mathbb{R}^n)$, ℓ^p and Sobolev spaces $W^{m,p}(\mathbb{R}^n)$ [212] for $p \in (1, \infty)$ as well as in all Hilbert spaces.

Our second aim is to investigate the necessary methodological coherence when optimization algorithms are applied to large-scale discretized problems: it is then important to show that AR methods continue to make sense in the limit, as the discretization mesh converges to zero. This coherence, sometimes called "mesh independence", has long been considered

as an important feature of numerical optimization methods [132, 2, 78, 123, 198]. For trust-region methods, this was studied in [193] in the Hilbert space context, and developed for Hilbert and Banach spaces in [68, Section 8.3]. Considering the question for AR algorithms therefore seems a natural development in this line of research. One might argue that most evaluation complexity results are "dimensionless", making this effort unnecessary. This argument however ignores an important point: problems in infinite dimension (and thus their discretizations) are often defined in spaces whose norms (and inner products when they exist) are not the standard Euclidean one. As a consequence, gradients must be measured in dual norms and thus approximate first-order points detected using these norms. This makes most existing complexity results applicable only though the use of norm-equivalence constants in large-scale finite dimensional approximations, whose value may significantly increase with dimension. The complexity estimates obtained using this approach can thus be severe over-estimates for large-dimensional discretizations of infinite-dimensional variational problems. Considering the norm adapted to the problem may therefore provide substantially more robust evaluation complexity bounds, which is the point of view developed in this chapter.

Our second objective however raises specific technical difficulties. While the outline of adaptive regularization methods is today quite well-known for finite dimensional spaces (see [29], for instance), its simple generalization to infinite dimensions is impossible. Indeed, the existence of a suitable step at a given iteration of the method in finite dimensions typically hinges on approaching a minimizer of a regularized model, which may no longer exist in infinite dimensions. Our analysis circumvents that problem by proposing a specialized optimization technique which guarantees an acceptable step for a class of function that, at variance with existing Lipschitz approaches, includes regularized polynomials.

Contributions. Having set the scene, we now make our contribution more precise.

- We first analyse the convergence of a first-order method for minimizing a regularized differentiable functions on a bounded set, where the first-order approximation error for the objective function's and the regularization's gradients satisfy a two-terms generalized Hölder condition. Significantly, this class includes regularized multivariate polynomials. To our knowledge, no such regularization has been considered before, even in finite dimensional spaces.

- Exploiting this result, we then propose an adaptive regularization algorithm whose step is found by minimizing a regularized polynomial and whose objective is to find first-order points of nonconvex functions having Hölder continuous pth derivative (in the Fréchet sense). We analyze its evaluation complexity and show that the sharp complexity bound known [52] for the finite-dimensional case is recovered, in that the algorithm requires at most $\mathcal{O}\big(\epsilon^{-\frac{p+\beta}{p+\beta-1}}\big)$ evaluations of the function and its first p derivatives to compute such a point.

Outline. The chapter is organized as follows. Section 5.2 considers the minimization of smooth regularized functionals in Banach spaces. Section 5.3 then introduces the class of Banach spaces of interest and details our general adaptive regularization algorithm for first-order minimization in these spaces, while Section 5.4 analyzes its evaluation complexity. We conclude the Chapter in Section 5.5 with a brief discussion of the new results and perspectives.

5.2 Gradient descent with a Hölder regularization

We start by considering the minimization, for x in the Banach space $\mathcal{V}$, of the regularized objective functional

$$\phi(x) \stackrel{\rm def}{=} \psi(x) + h(x), \tag{5.1}$$

where h is a general regularization term. This is motivated by the need to replace the problematic condition that the step of our yet to be defined regularization method is close to a minimizer by some more appropriate condition for infinite dimensional spaces, where ψ will play the role of the regularized model.

The space $\mathcal{V}$ and the functionals ϕ, ψ and h in (5.1) are assumed to satisfy the following properties.

AS.1

(i) There exists $\phi_{\min} \in \mathbb{R}$ such that, for all $x \in \mathcal{V}$, $\phi(x) \geq \phi_{\min}$. Moreover, the set $\mathcal{D} \stackrel{\rm def}{=} \{x \in \mathcal{V}\,,\, \phi(x) \leq \phi(0)\}$ is bounded in the sense that $\sup_{x\in\mathcal{D}} \|x\|_{\mathcal{V}} \leq \omega$ for some $\omega < \infty$.

(ii) ψ is a Fréchet differentiable function that satisfies the local two-terms Hölder condition

$$\forall \delta > 0,\, \forall x \in \mathcal{B}(0,\delta),\, \forall y \in \mathcal{V},\, \|\nabla_x^1 \psi(x) - \nabla_x^1 \psi(y)\|_{\mathcal{V}'} \leq L_{1,\delta} \|x-y\|_{\mathcal{V}}^{\beta_1} + L_{2,\delta} \|x-y\|_{\mathcal{V}}^{\beta_2},$$

where $\beta_1 > 0$ and $\beta_2 > 0$, $L_{1,\delta} > 0$ and $L_{2,\delta} > 0$ are constants, the latter two depending on δ.

(iii) h is a convex Fréchet differentiable function whose gradient satisfies the local two-terms Hölder condition

$$\forall \delta > 0,\, \forall x \in \mathcal{B}(0,\delta),\, \forall y \in \mathcal{V},\, \|\nabla_x^1 h(x) - \nabla_x^1 h(y)\|_{\mathcal{V}'} \leq L_{3,\delta} \|x-y\|_{\mathcal{V}}^{\beta_3} + L_{4,\delta} \|x-y\|_{\mathcal{V}}^{\beta_4},$$

where $\beta_3 > 0$ and $\beta_4 > 0$, $L_{3,\delta} > 0$ and $L_{4,\delta} > 0$ are constants, the latter two depending on δ. Moreover, $\min[\beta_3, \beta_4] \leq 1$

(iv) the space $\mathcal{V}$ is reflexive.

The conditions stated in AS.1(ii) and (iii) are verified by functionals with Hölder continuous gradient as proven in [55, Lemma 2.1] (β_1 is then equal to the Hölder exponent and $L_{1,\delta}$ equal to the Hölder constant). We use the more slightly more general conditions of AS.1(ii) in order to widen the class of allowed functionals and, in particular, to cover multivariate polynomials. Observe also that, should β_3 and β_4 both exceed one, then h must be affine and, since we do not exclude an affine ψ, AS.1(i) could then be violated. This potential contradiction justifies our assumption that $\min[\beta_3, \beta_4] \leq 1$.

The conditions stated in AS.1(ii) (for ψ) and (iii) (for h) are identical, and they obvioulsy combine to yield that

$$\forall \delta > 0,\, \forall x \in \mathcal{B}(0,\delta),\, \forall y \in \mathcal{V},\, \|\nabla_x^1 \phi(x) - \nabla_x^1 \phi(y)\|_{\mathcal{V}'} \leq L'_{1,\delta} \|x-y\|_{\mathcal{V}}^{\delta_1} + L'_{2,\delta} \|x-y\|_{\mathcal{V}}^{\delta_2}, \tag{5.2}$$

where $\delta_1 = \min(\beta_1, \beta_2, \beta_3, \beta_4) \leq 1$, $\delta_2 = \max(\beta_1, \beta_2, \beta_3, \beta_4)$ and $L'_{1,\delta} = L'_{2,\delta} = \sum_{i=1}^4 L_{i,\delta}$. We could clearly have assumed this condition on the gradient of ϕ directly, but we have preferred separate statements because AS.1(ii) and (iii) will be proved separately for the functionals of

interest. We immediately verify that multivariate polynomials satisfy AS.1(ii). This result is crucial for our purposes, as it will allow us to compute a step in the ARp-BS algorithm defined below.

Lemma 5.2.1 *Consider a multivariate polynomial functional $\psi : \mathcal{V} \to \mathbb{R}$ given, for $x \in \mathcal{V}$, by*

$$\psi(x) = \psi_0 + \sum_{\ell=1}^{p} \frac{1}{l!} S_\ell[x]^\ell, \tag{5.3}$$

where $S_\ell \in \mathcal{L}^\ell_{sym}(\mathcal{V}^{\otimes \ell})$ for $\ell \in \{1, \ldots, p\}$. Then, ψ satisfies AS.1(ii).

Proof. First observe that $\nabla^1_x \psi(x) = \sum_{\ell=1}^{p} \frac{1}{(\ell-1)!} S_\ell[x]^{\ell-1}$. Suppose first that $p = 1$. Then $\|\nabla^1_x \psi(x) - \nabla^1_x \psi(y)\|_{\mathcal{V}'} = 0$ for all x, y and the condition of AS.1(ii) holds for arbitrary positive values of $L_{1,\delta}$, $L_{2,\delta}$, β_1 and β_2. Suppose therefore that $p > 1$. For $x \in \mathcal{B}(0, \delta)$, $y \in \mathcal{V}$ and $u \in \mathcal{V}$, $\|u\|_{\mathcal{V}} = 1$ we then derive that

$$\begin{aligned}
\langle \nabla^1_x \psi(y) - \nabla^1_x \psi(x), u \rangle &= \sum_{\ell=1}^{p} \frac{1}{(\ell-1)!} \langle S_\ell[x + (y-x)]^{\ell-1} - S_\ell[x]^{\ell-1}, u \rangle, \\
&= \sum_{\ell=1}^{p} \frac{1}{(\ell-1)!} \left\langle \sum_{i=0}^{\ell-1} \binom{\ell}{i} S_\ell[x]^{\ell-1-i}[(y-x)]^i - S_\ell[x]^{\ell-1}, u \right\rangle, \\
&= \sum_{\ell=2}^{p} \frac{1}{(\ell-1)!} \left\langle \sum_{i=1}^{\ell-1} \binom{\ell}{i} S_\ell[x]^{\ell-1-i}[(y-x)]^i, u \right\rangle, \\
&\leq \sum_{\ell=2}^{p} \sum_{i=1}^{\ell-1} \frac{1}{(\ell-1)!} \binom{\ell}{i} \|S_\ell\| \|x\|^{l-1-i}_{\mathcal{V}} \|y-x\|^i_{\mathcal{V}}, \\
&\leq \sum_{\ell=2}^{p} \kappa_{\ell,\delta} \|y-x\|^{\ell-1}_{\mathcal{V}},
\end{aligned} \tag{5.4}$$

For $\|y - x\|_{\mathcal{V}} \leq 1$, an upper bound on the right hand side of (5.4) is given by

$$\langle \nabla^1_x \psi(y) - \nabla^1_x \psi(x), u \rangle \leq \sum_{\ell=2}^{p} \kappa_{\ell,\delta} \|y-x\|_{\mathcal{V}}, \tag{5.5}$$

while, for $\|y - x\|_{\mathcal{V}} \geq 1$, it is given by

$$\langle \nabla^1_x \psi(y) - \nabla^1_x \psi(x), u \rangle \leq \sum_{\ell=2}^{p} \kappa_{\ell,\delta} \|y-x\|^{p-1}_{\mathcal{V}}, \tag{5.6}$$

Combining (5.5), (5.6) and the fact that $\|u\|_{\mathcal{V}} = 1$ yields AS.1(ii) with $\beta_1 = 1$, $\beta_2 = p - 1 \geq 1$ and $L_{1,\delta} = L_{2,\delta} = \sum_{\ell=2}^{p} \kappa_{\ell,\delta}$. □

We now analyze the following very simple first-order linesearch-based algorithm on the facing page for the minimization of ϕ.

Algorithm 5.2.1: A Simple First-Order Algorithm for Minimizing Regularized Functionals Satisfying (5.2)

Step 0: Initialization. The constants $0 < c_1 < c_2 < 1$ are given. Set $x_0 = 0$ and $k = 0$.

Step 1: Compute a search direction. Compute $\nabla_x^1 \phi(x_k) \in \mathcal{V}'$. If $\|\nabla_x^1 \phi(x_k)\|_{\mathcal{V}'} = 0$, terminate and return x_k. Otherwise, select a direction d_k such that $\|\nabla_x^1 \phi(x_k)\|_{\mathcal{V}'} = -\langle \nabla_x^1 \phi(x_k), d_k \rangle$ and $\|d_k\| = 1$.

Step 2: Linesearch. Compute α_k a stepsize satisfying

$$\phi(x_k + \alpha_k d_k) \leq \phi(x_k) + \alpha_k c_1 \langle \nabla_x^1 \phi(x_k), d_k \rangle, \tag{5.7}$$

$$\langle \nabla_x^1 \phi(x_k + \alpha_k d_k), d_k \rangle \geq c_2 \langle \nabla_x^1 \phi(x_k), d_k \rangle. \tag{5.8}$$

Step 3: Define the next iterate. Set $x_{k+1} = x_k + \alpha_k d_k$, increment k by one and return to Step 1.

Note that the existence of the direction d_k in Step 1 is guaranteed by AS.1(iv) and James' theorem [126]. The reader has undoubtedly recognized the Wolfe linesearch conditions in (5.7) and (5.8) (see [172]). Unfortunately, the general form of (5.2) prevents extending the standard convergence theory for such algorithms applied to functions with Lipschitz gradients [172, Theorem 3.2] to our case. However, a modest modification of the classical argument allows us to prove the following convergence result.

Theorem 5.2.2 *Suppose that ψ, h and $\mathcal{V}$ verify AS.1 and let $\{x_k\}_{k \geq 0}$ be the sequence generated by Algorithm 5.2.1. Then*

$$\phi(x_{k+1}) < \phi(x_k) \quad \textit{for all} \quad k \geq 0$$

and either the algorithm terminates in a finite number of iterations with an iterate x_k such that $\nabla_x^1 \phi(x_k) = 0$, or

$$\lim_{k \to \infty} \|\nabla_x^1 \phi(x_k)\|_{\mathcal{V}'} = 0.$$

Proof. Because of the first Wolfe condition (5.7), the values $\{\phi(x_k)\}$ produced by Algorithm 5.2.1 are strictly decreasing, proving the theorem's first statement. More guarantees that all x_k lie in the level set $\mathcal{D}$. Using now the second Wolfe condition (5.8), we obtain that

$$\langle \nabla_x^1 \phi(x_{k+1}) - \nabla_x^1 \phi(x_k), d_k \rangle \geq (c_2 - 1) \langle \nabla_x^1 \phi(x_k), d_k \rangle = (1 - c_2) \|\nabla_x^1 \phi(x_k)\|_{\mathcal{V}'},$$

which, together with the fact that both x_k and x_{k+1} belong to $\mathcal{D}$, (5.2) (with $\delta = \omega$) and

$\|d_k\|_{\mathcal{V}} = 1$, ensures that

$$(1-c_2)\|\nabla_x^1\phi(x_k)\|_{\mathcal{V}'} \leq \langle \nabla_x^1\phi(x_{k+1}) - \nabla_x^1\phi(x_k), d_k\rangle \leq L'_{1,\omega}\alpha_k^{\delta_1} + L'_{2,\omega}\alpha_k^{\delta_2},$$

with $\delta_1 < \delta_2$. If $\alpha_k \leq 1$, we obtain from the last inequality that

$$\alpha_k \geq \left(\frac{(1-c_2)\|\nabla_x^1\phi(x_k)\|_{\mathcal{V}'}}{L'_{1,\omega} + L'_{2,\omega}}\right)^{\frac{1}{\delta_1}}. \tag{5.9}$$

Conversely, if $\alpha_k \geq 1$, then

$$\alpha_k \geq \left(\frac{(1-c_2)\|\nabla_x^1\phi(x_k)\|_{\mathcal{V}'}}{L'_{1,\omega} + L'_{2,\omega}}\right)^{\frac{1}{\delta_2}}. \tag{5.10}$$

Therefore,

$$\alpha_k \geq \mu \min\left[\|\nabla_x^1\phi(x_k)\|_{\mathcal{V}'}^{\frac{1}{\delta_1}}, \|\nabla_x^1\phi(x_k)\|_{\mathcal{V}'}^{\frac{1}{\delta_2}}\right],$$

where

$$\mu = \min\left[\left(\frac{(1-c_2)}{L'_{1,\omega} + L'_{2,\omega}}\right)^{\frac{1}{\delta_2}}, \left(\frac{(1-c_2)}{L'_{1,\omega} + L'_{2,\omega}}\right)^{\frac{1}{\delta_1}}\right].$$

Combining this lower bound on α_k with the first Wolfe condition yields that

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\mu \min\left(\|\nabla_x^1\phi(x_k)\|_{\mathcal{V}'}^{\frac{1}{\delta_1}}, \|\nabla_x^1\phi(x_k)\|_{\mathcal{V}'}^{\frac{1}{\delta_2}}\right)\|\nabla_x^1\phi(x_k)\|_{\mathcal{V}'}, \tag{5.11}$$

To prove the second theorem statement, we first note that the definition of the algorithm ensures the identity $\nabla_x^1\phi(x_k) = 0$ whenever termination occurs after a finite number of iterations. Assume therefore that the algorithm generates an infinite sequence of iterates and that

$$\|\nabla_x^1\phi(x_{k_i})\|_{\mathcal{V}'} \geq \epsilon, \tag{5.12}$$

for some $\epsilon > 0$ and some subsequence $\{k_i\}_{i=1}^{\infty}$. Summing over all iterations k_i and using AS.1(i), we obtain that

$$\begin{aligned} +\infty > \phi(0) - \phi_{\min} &\geq \sum_{i=1}^{\infty} c_1\mu \min\left(\|\nabla_x^1\phi(x_{k_i})\|_{\mathcal{V}'}^{\frac{\delta_1+1}{\delta_1}}, \|\nabla_x^1\phi(x_{k_i})\|_{\mathcal{V}'}^{\frac{\delta_2+1}{\delta_2}}\right), \\ &\geq c_1\mu \sum_{i=1}^{\infty} \min[\epsilon^{\frac{\delta_1+1}{\delta_1}}, \epsilon^{\frac{\delta_2+1}{\delta_2}}], \end{aligned} \tag{5.13}$$

which is a contradiction since the right-hand side diverges to $+\infty$. Hence (5.12) cannot hold and the second conclusion of the theorem holds. □

Thus a vanilla linesearch gradient-descent algorithm with the standard Wolfe conditions applied to infinite-dimensional functionals verifying AS.1 yields asymptotic first-order stationarity. This is significant for our purpose of developing an adaptive regularization algorithm using a model defined by a regularized polynomial. Note that the iteration complexity of this

algorithm in terms of $\epsilon \in (0,1]$ can easily be derived from (5.13) since

$$\phi(0) - \phi_{\min} \geq c_1 \mu \sum_{i=1}^{N_\epsilon} \min[\epsilon^{\frac{\delta_1+1}{\delta_1}}, \epsilon^{\frac{\delta_2+1}{\delta_2}}] \geq N_\epsilon c_1 \mu \epsilon^{\frac{\delta_1+1}{\delta_1}}, \tag{5.14}$$

where N_ϵ denotes the total number of iterations to achieve the algorithm's termination condition. The iteration complexity for the linesearch Algorithm 5.2.1 is therefore $\mathcal{O}\left(\epsilon^{-\frac{1+\delta_1}{\delta_1}}\right)$ as a function of the requested accuracy ϵ of the gradient's norm. Note that $\delta_1 \leq 1$ and hence this bound cannot be better than $\mathcal{O}\left(\epsilon^{-2}\right)$. This is reminiscent of Theorem 3.2 in [53], where the evaluation complexity of an adaptive regularization method in $\mathbb{R}^n$ is analyzed for functions with Hölder continuous gradients and a more specific regularization $h(x) = \|x\|_2^r$.

5.3 An adaptive regularization algorithm in Banach spaces

We now consider developing an adaptive regularization method for finding first-order points for the problem

$$\min_{x \in \mathcal{V}} f(x), \tag{5.15}$$

and make our assumptions on the problem more precise.

AS.2 f is a function of the $\mathcal{C}^{p,\beta}(\mathcal{V};\mathbb{R})$ class. To revisit the definition, please refer to Subsection 1.3.3.1 in the Introduction.

AS.3 There exists a constant f_{low} such that $f(x) \geq f_{\text{low}}$ for all $x \in \mathcal{V}$.

The gradient $\nabla_x^1 f(x)$ belongs to the dual space $\mathcal{V}'$ and will be denoted by $g(x)$. Thus, for a requested accuracy $\epsilon \in (0,1]$, we are interested in finding an ϵ-approximate first-order critical point, that is a point x_ϵ such that $\|g(x_\epsilon)\|_{\mathcal{V}'} \leq \epsilon$.

5.3.1 Smooth Banach spaces

In a generic Banach space, we can only ensure "a decrease principle" as stated in [65, Theorem 5.22]. To obtain more conclusive results, we need to introduce additional assumptions. We choose to work with the class of *uniformly q smooth Banach spaces*. For the sake of completeness, we briefly recall the context. Given a Banach space $\mathcal{V}$, we first define its module of smoothness, for $t \geq 0$, by

$$\rho_{\mathcal{V}}(t) \stackrel{\text{def}}{=} \sup_{\|x\|_{\mathcal{V}}=1\,,\|y\|_{\mathcal{V}}=t} \left\{ \frac{\|x+y\|_{\mathcal{V}} + \|x-y\|_{\mathcal{V}}}{2} - 1 \right\}, \tag{5.16}$$

and immediately deduce from the triangular inequality that $\rho_{\mathcal{V}}(t) \leq t$. We now say that $\mathcal{V}$ is a uniformly smooth Banach space if and only if $\lim_{t\to 0} \frac{\rho_{\mathcal{V}}(t)}{t} = 0$. Going one step further, we say that a Banach space $\mathcal{V}$ is *uniformly q smooth* for some $q \in (1,2]$ if and only if

$$\exists \kappa_{\mathcal{V}} > 0,\ \rho_{\mathcal{V}}(t) \leq \kappa_{\mathcal{V}} t^q. \tag{5.17}$$

It is easy to see that, if $\mathcal{V}$ is uniformly q smooth, it is also uniformly q' smooth for all $1 < q' < q$. Indeed, one can easily show[1] that $\rho_{\mathcal{V}}(t) \leq \max(1, \kappa_{\mathcal{V}})t^{q'}$ from definition (5.16) and inequality (5.17).

We motivate our choice of this particular class of Banach spaces by giving a few examples. $L^p(\mathbb{R}^n)$, $1 < p < \infty$, are uniformly smooth Banach spaces. In particular, $L^p(\mathbb{R}^n)$ is uniformly 2 smooth for $p \geq 2$ and uniformly p smooth for $1 < p \leq 2$. The same results apply for ℓ^p and the Sobolev spaces $W^{m,p}(\mathbb{R}^n)$. Moreover, all Hilbert spaces are 2 smooth Banach. See [212] for more details. One might wonder if it is possible for the q smooth order to be strictly superior to 2 in (5.17). We now show that this is impossible. Indeed, for any Banach space $\mathcal{V}$, we have that, $\rho_{\mathcal{V}}(t) \geq \rho_{\mathcal{H}}(t) = \frac{t^2}{\sqrt{1+t^2}+1}$ [212]. Suppose now $\rho_{\mathcal{V}}(t) \leq ct^m$ with $m > 2$. Using the last two inequalities, we obtain that: $ct^{m-2} \geq \frac{1}{\sqrt{1+t^2}+1}$ for all t strictly positive. But this inequality is impossible for small enough t and hence our supposition about m is false and $m \in (1, 2]$.

From here on, we assume that

AS.4 $\mathcal{V}$ is a uniformly q smooth space.

Uniformly smooth Banach spaces are also reflexive (See [212, Proposition 1.e.3, p.61]), so that AS.1(iv) automatically holds. Let us now define the set

$$J_p(x) \stackrel{\text{def}}{=} \left\{ v^* \in \mathcal{V}' \,,\, \langle v^*, x \rangle = \|x\|_{\mathcal{V}}^p \,,\, \|v^*\|_{\mathcal{V}'} = \|x\|_{\mathcal{V}}^{p-1} \right\}. \tag{5.18}$$

It is known [210] that $J_p(x)$ is the subdifferential of the functional $\frac{1}{p}\|\cdot\|_{\mathcal{V}}^p$, $p \geq 1$ at x. We may now introduce another characterization of uniform smoothness.

Theorem 5.3.1 *Let*

$\mathcal{F} \stackrel{def}{=} \{\psi : I\!R \to I\!R \mid \psi(0) = 0, \psi$ *is convex, non decreasing and* $\exists \kappa_{\mathcal{F}} > 0 \mid \psi(t) \leq \kappa_{\mathcal{F}} \rho_{\mathcal{V}}(t)\}$.

Then, for any $1 < p < \infty$*, the following statements are equivalent.*

(i) $\mathcal{V}$ *is a uniformly smooth Banach space.*

(ii) J_p *is single valued and there exists* $\varphi_p(t) = \frac{\psi_p(t)}{t}$ *where* $\psi_p \in \mathcal{F}$ *and such that*

$$\|J_p(x) - J_p(y)\|_{\mathcal{V}'} \leq \max(\|x\|_{\mathcal{V}}, \|y\|_{\mathcal{V}})^{p-1} \varphi_p\left(\frac{\|x - y\|_{\mathcal{V}}}{\max(\|x\|_{\mathcal{V}}, \|y\|_{\mathcal{V}})}\right). \tag{5.19}$$

Proof. [212, Theorem 2]. □

We define $J_p(x)$ as the unique value in the set (5.18). As the subdifferential of $\|.\|_{\mathcal{V}}^p$ reduces to a singleton for $p > 1$ and $\|.\|_{\mathcal{V}}^p$ is a convex function, $\|.\|_{\mathcal{V}}^p$ is Fréchet differentiable for $p > 1$ since it verifies [65, Condition 4.16]. The reader is referred to [210] or [212] for more extensive coverage of characterizations of the norm in uniformly smooth Banach spaces.

[1] If $t \in [0, 1]$ this follows from (5.17) and $q' < q$. If $t > 1$, $\rho_{\mathcal{V}}(t) \leq t \leq t^{q'}$.

For all $\ell > 1$, we now prove an upper bound of the norm of $\|J_\ell(x) - J_\ell(y)\|_{\mathcal{V}'}$ in terms of $\|x - y\|_{\mathcal{V}}$ in a uniform q smooth Banach space. Let us first remind the useful inequality $(x+y)^r \leq \max(1, 2^{r-1})(x^r + y^r)$ for all $x, y \geq 0$ and all $r \geq 0$, before stating the next crucial lemma.

Lemma 5.3.2 *Suppose that $\mathcal{V}$ is a uniformly q smooth Banach space and that $x \in \mathcal{B}(0, \omega)$. Then for all $\ell > 1$, there exist constants $\kappa_\omega, \kappa_\ell > 0$ such that*

$$\|J_\ell(x) - J_\ell(y)\|_{\mathcal{V}'} \leq \kappa_\omega \|x - y\|_{\mathcal{V}}^{\min[q,\ell]-1} + \kappa_\ell \|x - y\|_{\mathcal{V}}^{\ell-1}, \tag{5.20}$$

where κ_ω and κ_ℓ depend only on ω, ℓ, $\kappa_{\mathcal{F}}$ and $\kappa_{\mathcal{V}}$.

Proof. As $\ell > 1$, if $q > \ell$, we can use our remark above and decrease the q smooth order until $q' = \min(q, \ell) \leq \ell$. We now develop the upper bound (ii) of Theorem 5.3.1 and use the definition of the set $\mathcal{F}$ to derive that

$$\begin{aligned}
\|J_l(x) - J_l(y)\|_{\mathcal{V}'} &\leq \max(\|x\|_{\mathcal{V}}, \|y\|_{\mathcal{V}})^{\ell-1} \kappa_{\mathcal{F}} \kappa_{\mathcal{V}} \left(\frac{\|x - y\|_{\mathcal{V}}}{\max(\|x\|_{\mathcal{V}}, \|y\|_{\mathcal{V}})} \right)^{q'-1}, \\
&\leq \max(\|x\|_{\mathcal{V}}, \|y\|_{\mathcal{V}})^{l-q'} \kappa_{\mathcal{F}} \kappa_{\mathcal{V}} \|x - y\|_{\mathcal{V}}^{q'-1}.
\end{aligned}$$

Using now the inequalities $\max(\|x\|_{\mathcal{V}}, \|y\|_{\mathcal{V}}) \leq \|x\|_{\mathcal{V}} + \|x - y\|_{\mathcal{V}}$ and $\ell \geq q'$, we obtain that

$$\begin{aligned}
\|J_l(x) - J_l(y)\|_{\mathcal{V}'} &\leq \kappa_{\mathcal{F}} \kappa_{\mathcal{V}} (\|x\|_{\mathcal{V}} + \|x - y\|_{\mathcal{V}})^{\ell-q'} \|x - y\|_{\mathcal{V}}^{q'-1}, \\
&\leq \kappa_{\mathcal{F}} \kappa_{\mathcal{V}} \max(1, 2^{\ell-q'-1}) (\|x\|_{\mathcal{V}}^{\ell-q'} + \|x - y\|_{\mathcal{V}}^{\ell-q'}) \|x - y\|_{\mathcal{V}}^{q'-1}, \\
&\leq \kappa_{\mathcal{F}} \kappa_{\mathcal{V}} \max(1, 2^{\ell-q'-1})\, \omega^{\ell-q'} \|x - y\|_{\mathcal{V}}^{q'-1} \\
&\qquad + \kappa_{\mathcal{F}} \kappa_{\mathcal{V}} \max(1, 2^{\ell-q'-1}) \|x - y\|_{\mathcal{V}}^{\ell-1}, \\
&\leq \kappa_\omega \|x - y\|_{\mathcal{V}}^{q'-1} + \kappa_\ell \|x - y\|_{\mathcal{V}}^{\ell-1}.
\end{aligned}$$

□

It results from this theorem that the primal representation of the gradient of a regularization term of the form $\|s\|_{\mathcal{V}}^\alpha$ does satisfy the condition of AS.1(iii). This will be crucial as it will allow applying Algorithm 5.2.1 to a model consisting of a multivariate polynomial (satisfying AS.1(ii)) augmented by such a regularization term.

5.3.2 The ARp-BS algorithm

In our uniform q smooth setting, $m_k(s)$ defined in (1.3.18) is Fréchet differentiable but this is unfortunately insufficient to derive results on the Lipschitz continuity of its gradient, which makes the use of more standard gradient-descent methods impossible. We state now our ARp algorithm for smooth Banach spaces.

The ARp-BS algorithm follows the main lines of existing ARp methods [51, 29]. However, as we have already mentioned, the existence of a minimizer of $m_k(s)$ may not be guaranteed

Algorithm 5.3.1: pth order adaptive regularization in a uniform q smooth Banach Space (ARp-BS)

Step 0: Initialization. An initial point $x_0 \in \mathcal{V}$, a regularization parameter σ_0 and a requested final gradient accuracy $\epsilon \in (0, 1]$ are given. The constants η_1, η_2, γ_1, γ_2, γ_3, $\chi \in (0, 1)$, and $\sigma_{\min}$ are also given such that

$$\sigma_{\min} \in (0, \sigma_0], 0 < \eta_1 \leq \eta_2 < 1 \quad \text{and} \quad 0 < \gamma_1 < 1 < \gamma_2 < \gamma_3. \tag{5.21}$$

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

Step 1: Check for termination. Evaluate $g_k = \nabla_x^1 f(x_k)$. Terminate with $x_\epsilon = x_k$ if

$$\|g(x_k)\|_{\mathcal{V}'} \leq \epsilon. \tag{5.22}$$

Step 2: Step calculation. Evaluate $f(x_k)$ and $\{\nabla_x^i f(x_k)\}_{i=2}^p$. Compute a step s_k which sufficiently reduces the model m_k defined in (1.3.18) in the sense that

$$m_k(s_k) < m_k(0), \tag{5.23}$$

and

$$\|\nabla_s^1 m_k(s_k)\|_{\mathcal{V}'} \leq \max\left(\chi\epsilon,\, \theta\|s_k\|_{\mathcal{V}}^{p+\beta-1}\right). \tag{5.24}$$

Step 3: Acceptance of the trial point. Compute $f(x_k + s_k)$ and define

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{T_{f,p}(x_k, 0) - T_{f,p}(x_k, s_k)}. \tag{5.25}$$

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

Step 4: Regularization parameter update. Set

$$\sigma_{k+1} \in \begin{cases} [\max(\sigma_{\min}, \gamma_1\sigma_k), \sigma_k] & \text{if } \rho_k \geq \eta_2, \\ [\sigma_k, \gamma_2\sigma_k] & \text{if } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_2\sigma_k, \gamma_3\sigma_k] & \text{if } \rho_k < \eta_1. \end{cases} \tag{5.26}$$

Increment k by one and go to Step 1.

in infinite dimensions and hence a point $s^\star$ such that $\nabla_s^1 m_k(s^\star) = 0$ may not exist. As a consequence, standard proofs that a step satisfying both (5.23) and (5.24) exists no longer apply. We thus need to check that a suitable step can still de found in our context. This is achieved using Algorithm 5.2.1.

Theorem 5.3.3 *Suppose that AS.2 and AS.4 hold. Suppose also that* $\|g(x_k)\|_{\mathcal{V}'} > 0$. *Then a step satisfying both (5.23) and (5.24) always exists.*

Proof. First note that AS.2 imply that $p + \beta > 1$. In order to apply Algorithm 5.2.1 to the problem of minimizing (1.3.18), we just need to prove that $m_k(s)$ satisfies AS.1 of Section 2. We have that

$$m_k(s) \geq m_k(0) - \sum_{i=1}^{p} \|\nabla_x^i f(x)\| \|s\|_{\mathcal{V}}^i + \frac{\sigma_k}{(p+\beta)!} \|s\|_{\mathcal{V}}^{p+\beta} \rightarrow \infty \text{ as } \|s\|_{\mathcal{V}} \rightarrow \infty,$$

and thus m_k is a coercive functional verifying AS.1(i). Lemma 5.2.1 ensures that the Taylor series term $T_{f,p}(x_k, s)$ satisfies AS.1(ii). Lemma 5.3.2 (applied with $\delta = \omega$, $\ell = p + \beta - 1$, $L_{3,\delta} = \kappa_\ell$, $\beta_3 = \min[q, \ell] - 1 \in (0, 1]$, $L_{4,\delta} = \kappa_\omega$ and $\beta_4 = \ell + \beta - 1 > 0$) then ensures that $\|.\|_{\mathcal{V}}^{p+\beta}$ satisfies AS.1(iii). We already noted that, being uniformly smooth, $\mathcal{V}$ must be reflexive, which ensures that AS.1(iv) holds. All the requirements of AS.1 in Section 2 are therefore met and, since $\nabla_s^1 m_k(0) = g(x_k)$, Theorem 5.2.2 applies to the functional $m_k(s)$. As a consequence, a suitable step s_k such that $m_k(s_k) < m_k(0)$ and $\|\nabla_s^1 m_k(s_k)\|_{\mathcal{V}'} \leq \chi\epsilon$ exists. □

Observe that equation (5.14) and the fact that $\delta_1 = \min[q, p+\beta]-1$ and $\delta_2 = p+\beta-1$ (all the other powers ranging from 2 to p), imply that, for our iterative gradient descent Algorithm 5.2.1,

$$\lim_{i \rightarrow \infty} \min \left[\kappa_A \|\nabla_s^1 m(s_i)\|_{\mathcal{V}'}^{\frac{\min[q,p+\beta]}{\min[q,p+\beta]-1}}, \kappa_B \|\nabla_s^1 m(s_i)\|_{\mathcal{V}'}^{\frac{p+\beta}{p+\beta-1}} \right] = 0.$$

As a consequence, the first term in the minimum indicates that the smoother the space, the faster the convergence for $p \geq 2$.
Following well-established practice, we now define

$$\mathcal{S} \stackrel{\text{def}}{=} \{k \geq 0 \mid x_{k+1} = x_k + s_k\} = \{k \geq 0 \mid \rho_k \geq \eta_1\},$$

the set of indexes of "successful iterations", and

$$\mathcal{S}_k \stackrel{\text{def}}{=} \mathcal{S} \cap \{1, \ldots, k\},$$

the set of indexes of successful iterations up to iteration k. We also recall a well-known result bounding the total number of iterations in terms of the number of successful ones.

Lemma 5.3.4 *Suppose that the ARp-BS algorithm is used and that $\sigma_k \leq \sigma_{\max}$ for some $\sigma_{\max} > 0$. Then*

$$k \leq |\mathcal{S}_k| \left(1 + \frac{|\log \gamma_1|}{\log \gamma_2}\right) + \frac{1}{\log \gamma_2} \log\left(\frac{\sigma_{\max}}{\sigma_0}\right). \tag{5.27}$$

Proof. See proof of Lemma 4.2.1. □

5.4 Evaluation complexity for the ARp-BS algorithm

Before discussing our analysis of evaluation complexity, we remind the reader that the two first inequalities of Lemma 1.3.1 hold.

From now on, the analysis follows that presented in [29] quite closely.

Lemma 5.4.1

$$\Delta T_{f,p}(x_k, s_k) \stackrel{def}{=} T_{f,p}(x_k, 0) - T_{f,p}(x_k, s_k) \geq \frac{\sigma_k}{(p+\beta)!} \|s_k\|_{\mathcal{V}}^{p+\beta}. \tag{5.28}$$

Proof. Direct from (5.23) and (1.3.18). □

Lemma 5.4.2 *Suppose that $f \in \mathcal{C}^{p,\beta}(\mathcal{V}; I\!R)$. Then, for all $k \geq 0$,*

$$\sigma_k \leq \sigma_{\max} \stackrel{def}{=} \gamma_3 \max\left(\sigma_0, \frac{L_p}{(1-\eta_2)}\right). \tag{5.29}$$

Proof. See [29, Lemma 2.2]. Using (5.25), (1.3.15), and (5.28), we obtain that

$$|\rho_k - 1| \leq \frac{(p+\beta)!|f(x_k + s_k) - T_{f,p}(x_k, s_k)|}{\sigma_k \|s_k\|_{\mathcal{V}}^{p+\beta}} \leq \frac{L_p}{\sigma_k}.$$

Thus, if $\sigma_k \geq L_p/(1-\eta_2)$, then $\rho_k \geq \eta_2$ ensures that iteration k is successful and (5.26) implies that $\sigma_{k+1} \leq \sigma_k$. The mechanism of the algorithm then guarantees that (5.29) holds. □

The next lemma remains in the spirit of [29, Lemma 2.3], but now takes the condition (5.24) into account.

Lemma 5.4.3 *Suppose that $f \in \mathcal{C}^{p+\beta}(\mathcal{V};\mathbb{R})$ holds and that $k \in \mathcal{S}$ before termination. Then*

$$\|s_k\|_{\mathcal{V}}^{p-1+\beta} \geq \epsilon \min\left[\frac{(1-\chi)(p+\beta-1)!}{L_p+\sigma_{\max}}, \frac{(p+\beta-1)!}{L_p+\sigma_{\max}+\theta(p+\beta-1)!}\right]. \tag{5.30}$$

Proof. Successively using the fact that termination does not occur at iteration k, (1.3.16) and condition (5.24), we deduce that

$$\begin{aligned}
\epsilon &< \|g(x_{k+1})\|_{\mathcal{V}'},\\
&\leq \|g(x_{k+1}) - \nabla_s^1 T_{f,p}(x_k,s_k)\|_{\mathcal{V}'} + \|\nabla_s^1 m_k(s_k)\|_{\mathcal{V}'} + \frac{\sigma_k}{(p+\beta-1)!}\|J_{p+\beta}(s_k)\|_{\mathcal{V}'},\\
&\leq \frac{L_p}{(p-\beta+1)!}\|s_k\|_{\mathcal{V}}^{p-1+\beta} + \max\left(\chi\epsilon, \theta\|s_k\|_{\mathcal{V}}^{p-\beta+1}\right) + \frac{\sigma_k}{(p+\beta-1)!}\|s_k\|_{\mathcal{V}}^{p+\beta-1}.
\end{aligned}$$

By treating each case in the maximum separately, we obtain that either

$$(1-\chi)\epsilon \leq \left(\frac{L_p}{(p+\beta-1)!} + \frac{\sigma_k}{(p+\beta-1)!}\right)\|s_k\|_{\mathcal{V}}^{p-1+\beta},$$

or

$$\epsilon \leq \left(\frac{L_p}{(p+\beta-1)!} + \frac{\sigma_k}{(p+\beta-1)!} + \theta\right)\|s_k\|_{\mathcal{V}}^{p-1+\beta}.$$

Combining the two last inequalities gives that

$$\|s_k\|_{\mathcal{V}}^{p-1+\beta} \geq \min\left[\frac{(1-\chi)\epsilon(p+\beta-1)!}{L_p+\sigma_{\max}}, \frac{(p+\beta-1)!\epsilon}{L_p+\sigma_{\max}+\theta(p+\beta-1)!}\right].$$

This in turn directly implies (5.30). □

We may now resort to the standard "telescoping sum" argument to obtain the desired evaluation complexity result.

Theorem 5.4.4 *Suppose that AS.2–AS.4 hold. Then the ARp-BS algorithm requires at most*

$$\kappa_{ARpBS} \frac{f(x_0) - f_{\,low}}{\epsilon^{\frac{p+\beta}{p+\beta-1}}},$$

successful iterations and evaluations of $\{\nabla_x^i f\}_{i=1,2,\ldots,p}$ *and at most*

$$\kappa_{ARpBS} \frac{f(x_0) - f_{low}}{\epsilon^{\frac{p+\beta}{p+\beta-1}}} \left(1 + \frac{|\log \gamma_1|}{\log \gamma_2}\right) + \frac{1}{\log \gamma_2} \log\left(\frac{\sigma_{\max}}{\sigma_0}\right),$$

evaluations of f *to produce a vector* $x_\epsilon \in \mathcal{V}$ *such that* $\|g(x_\epsilon)\|_{\mathcal{V}'} \leq \epsilon$*, where*

$$\kappa_{ARpBS} = \frac{(p+\beta-1)!}{\eta_1 \sigma_{\min}} \min\left[\frac{(1-\chi)(p+\beta-1)!}{L_p + \sigma_{\max}}, \frac{(p+\beta-1)!}{L_p + \sigma_{\max} + (p+\beta-1)!\theta}\right]^{\frac{p+\beta}{p+\beta-1}}.$$

Proof. Let k be the index of an iteration before termination. Then, using AS.3, the definition of successful iterations, (5.28) and (5.30), and the fact that computing an appropriate step is of constant order of complexity, we obtain that

$$f(x_0) - f_{\rm low} \geq \sum_{i=0, i\in\mathcal{S}}^{k} f(x_i) - f(x_{i+1}) \geq \eta_1 \sum_{i\in\mathcal{S}_k} \Delta T_{f,2}(x_i, s_i) \geq \frac{|\mathcal{S}_k|}{\kappa_{\rm ARpBS}} \epsilon^{\frac{p+\beta}{p+\beta-1}}.$$

Thus

$$|\mathcal{S}_k| \leq \kappa_{\rm ARpBS} \frac{f(x_0) - f_{\rm low}}{\epsilon^{\frac{p+\beta}{p+\beta-1}}},$$

for any k before termination. The first conclusion follows since the derivatives are only evaluated once per successful iteration. Applying now Lemma 5.3.4 gives the second conclusion.

□

Theorem 5.4.4 extends the result of [29] in the case $\beta = 1$ and some results of [55] to uniform q smooth Banach spaces. We recall that L^p, ℓ^p and $W^{m,p}$ are uniform q smooth spaces for $1 < p < \infty$, and hence that Lemma 5.3.2 and Theorem 5.4.4 apply in these spaces. We may also consider the finite dimensional case where $\mathbb{R}^n$ is equipped with the norm $\|x\|_r = (\sum_{i=1}^n |x_i|^r)^{\frac{1}{r}}$. We know that, for all $1 < r < \infty$, this is a uniform $\min(r, 2)$ smooth space, and therefore Theorem 3.5 again applies. We could of course have obtained convergence of the adaptive regularization algorithm in this case using results for the Euclidean norm and introducing norm-equivalence constants in our proofs and final result, but this is avoided by the approach presented here. This could be significant when the dimension is large and the norm-equivalence constants grow.

Finally note that the evaluation complexity of Algorithm 5.2.1 discussed at the end of Section 5.2 is interesting but irrelevant for the evaluation complexity of the ARp-BS algorithm, because the former only evaluates the model m_k without requiring any evaluations of f or its derivatives beyond those already performed in ARp-BS.

5.5 Discussions

We have proposed a generalized Hölder condition and a gradient-descent algorithm for minimizing polynomial functionals with a general convex regularization term in Banach spaces, and have applied this result to show the existence of a suitable step in an adaptive regularization method for unconstrained minimization in q smooth Banach spaces. We have also analyzed the evaluation complexity of this latter algorithm and have shown that, under standard assumptions, it will find an ϵ-approximate first-order critical point in at most $\mathcal{O}\big(\epsilon^{-\frac{p+\beta}{p+\beta-1}}\big)$ evaluations of the functional and its first p derivatives, which is identical to the bound known for minimization in (finite-dimensional) Euclidean spaces. Since these bounds are known to be sharp [52], so is ours.

It would be interesting to consider convergence to second-order points, but the infinite dimensional framework causes more difficulties. Indeed, considering second-order derivatives as in [55] is impossible since we do not know if a power of the norm is twice differentiable. As an example, consider $L^r([0,1])$ for $p > 1$, where

$$\nabla_f^1 \left(\frac{\|f\|^p_{L^r([0,1])}}{p} \right) = \|f\|^{p-r}_{L^r([0,1])} f|f|^{r-2}.$$

The right-hand side of the last equation involves an absolute value which is only differentiable for specific values of r. It is interesting to study the case of $r = 2$ with the objective of extending our analysis to the second order.

Chapter 6

Conclusions

6.1 Summary

In this thesis, we have presented several new results on the theory of unconstrained nonconvex minimization. We now give a concise review of these past results and mention perspectives arising from the exposed findings.

In the introduction, we gave an overview of nonlinear optimization methods. We have provided a summary of old and new nonlinear optimization paradigms with a focus on minimization techniques that use second or higher order information. All these methods use the objective function to adjust a specific parameter (linear search step, radius of the confidence region, etc...). While reviewing the state of the art, we noticed that there is room for improvement.

First, we pointed out that optimal second-order methods require involved subroutines and that solving a linear system is not sufficient. To overcome this problem, we proposed an algorithm that alternates between a regularized Newton method and a negative curvature step. Since exact curvature information is required, we have developed efficient numerical variants that avoid this pitfall. Developments related to this algorithm can be found in Chapter 4.

Next, we considered higher-order methods that provide better convergence speed. Although these methods are covered in various nonlinear optimization frameworks, they have not yet been extended to infinite-dimensional Banach spaces. In our manuscript, we propose an extension of derivative tensor methods to the above case. Our algorithm follows the standard line of analysis of adaptive regularized methods and solves the main difficulty by developing a gradient descent based on a linear search suitable for a specific class of minimization problems. For further developments and reasons for studying infinite-dimensional algorithms, we refer the reader to Chapter 5.

We then turned to the study of Machine Learning minimization problems. By analyzing the algorithms developed for the latter class of problems, we realized that the success of the specialized algorithms come from the fact that they don't use the objective function, don't need accurate derivative information such as Lipshitz gradient constant and still achieve convergence. This empirical observation was also confirmed by an analysis of the conditions required in the theory of both stochastic trust-region and probabilistic line search. We proposed a new point-of-view on these methods by including them into the new class of Objective Function Free Optimization techniques. We developed two significant contributions that we

briefly resumes below

- We proposed a novel view on Adagrad [154, 88], a well-known OFFO algorithm as a weighted trust-region method which allows the usage of curvature information. With this new framework, we propose a scaling scheme that uses the iteration counter and previously computed gradients. For both methods, we propose a stepsize rule resulting from our theoretical analysis. More details can be found in Chapter 2.
- Since the arguments of the introduction clearly pointed out the advantages of OFFO optimization techniques for noisy problems, and that no high-order derivative methods exist, we fill the gap by providing the first tensor methods that do not use function value while ensuring convergence. We achieve the same rate as previous standard adaptive methods despite using significantly less information. These novel adaptive regularization methods have been thoroughly analyzed in Chapter 3.

Numerical experiments show that the proposed methods, especially the OFFO algorithms, are suitable for the noisy problems. Furthermore, all developed schemes are competitive with previously well-established methods.

6.2 Perspectives of Further Research

We now discuss potential avenues for further research in the theory and practice of optimization methods.

Better Theory of OFFO

The newly introduced OFFO paradigm for second-order or higher methods is still incomplete, and there are several possible directions for understanding and improving it. Here are some avenues to explore. Obviously, a theoretical framework that justifies the success of second-order OFFO methods in a stochastic setting[1] is crucial to shed light on the advantages and limitations of this approach. As an example for first-order OFFO algorithms, a theory for the stochastic case has been established in several papers, see [77, 207, 141, 204] and the references therein. Therefore, developing a theory of the stochastic case is a crucial milestone for high-order OFFO methods. The development of other variants of adaptive OFFO regularization techniques may also be considered. One that naturally comes to mind is an update scheme where a component-wise update rule of the regularization is performed a la Adagrad as presented in Algorithm 1.4.1.

Beyond Lipschitz Assumption

First, we should also mention that well-known algorithms such as gradient descent have recently been revisited under assumptions that include the usual global Lipschitz continuity of the gradient, see [220, 177]. For example, [220] considered an additional term that depends on the norm of the gradient at the current iteration. It is then natural to extend this new smoothness condition to second-order or pth-order minimization schemes. Another line of development is to extend these high-order methods to a broader case, such as metrizable spaces (using Bregman divergence or Wasserstein distance) to ensure better fidelity to the geometry of the problem. What comes particular to mind is the class of *quartic* problems [87].

[1] See Numerics of Chapter 3

Further Developments of High-Order Tensors Methods

Although the theory of adaptive regularization algorithms is well established, practical and efficient developments lag behind and have only recently been proposed in [47] for the nonconvex case with $p = 3$. Further practical implementations may also require additional theoretical developments to allow efficient use of inexact tensor approximations or better exploitation of modern hardware infrastructure that allows data to be shared across multiple devices.

As a final word, the work presented in this book has developed the theory of nonlinear optimization. We have proposed a new view on old and new methods of nonconvex optimization, and we firmly believe that the above analysis is of interest for the optimization community and paves the way for the development of new algorithms and methods.

Appendix A

Image Classification with Neural Nets

Chapter Abstract

In this chapter, we provide some brief explanations on image classification with deep neural networks.

A.1 Neural Network Classification

Mean-Cross Entropy Loss The mean cross-entropy loss is a widely used loss function in classification tasks, especially in the context of deep learning. Mathematically, given a set of p samples with C classes, the mean cross-entropy loss is calculated as

$$\mathcal{L}(x; A, Y) \stackrel{\text{def}}{=} \frac{-1}{p} \sum_{i=1}^{p} \sum_{j=1}^{C} y_{i,j} \log(f(x, a_i)). \tag{A.1}$$

Where:

- p is the number of samples in the dataset.
- C is the number of classes.
- $y_{i,j}$ is the indicator function that equals 1 if the ith sample belongs to class j, and 0 otherwise.
- $f(x, a_{i,j})$ is the predicted that the ith sample belongs to class j.
- A is the input dataset and Y its target labels.

Appendix B

Datasets

Chapter Abstract

In this chapter, we provide an overview of the various datasets utilized for numerical illustrations across different chapters of the thesis.

B.1 OPM-Datasets

We detail in this section the test nonlinear optimization problems of the freely available OPM collection [110] in MATLAB. The collection comprises three distinct sets of problems: one set with small-dimensional problems, a second set with medium-dimensional ones, and a third set featuring "larger-scale" ones. Below, we present the specifics of each set.

Problem	n	Problem	n	Problem	n	Problem	n	Problem	n	Problem	n
argauss	3	chebyqad	10	dixmaanl	12	heart8ls	8	msqrtals	16	scurly10	10
arglina	10	cliff	2	dixon	10	helix	3	msqrtbls	16	scosine	10
arglinb	10	clplatea	16	dqartic	10	hilbert	10	morebv	12	sisser	2
arglinc	10	clplateb	16	edensch	10	himln3	2	nlminsurf	16	spmsqrt	10
argtrig	10	clustr	2	eg2	10	himm25	2	nondquar	10	tcontact	49
arwhead	10	cosine	10	eg2s	10	himm27	2	nzf1	13	tquartic	10
bard	3	crglvy	4	eigenals	12	himm28	2	osbornea	5	trigger	7
bdarwhd	10	cube	2	eigenbls	12	himm29	2	osborneb	11	tridia	10
beale	2	curly10	10	eigencls	12	himm30	3	penalty1	10	tlminsurfx	16
biggs5	5	dixmaana	12	engval1	10	himm32	4	penalty2	10	tnlminsurfx	16
biggs6	6	dixmaanb	12	engval2	3	himm33	2	penalty3	10	vardim	10
brownden	4	dixmaanc	12	expfit	2	hypcir	2	powellbs	2	vibrbeam	8
booth	2	dixmaand	12	extrosnb	10	indef	10	powellsg	12	watson	12
box3	3	dixmaane	12	fminsurf	16	integreq	10	powellsq	2	wmsqrtals	16
brkmcc	2	dixmaanf	12	freuroth	4	jensmp	2	powr	10	wmsqrtbls	16
brownal	10	dixmaang	12	genhumps	5	kowosb	4	recipe	2	woods	12
brownbs	2	dixmaanh	12	gottfr	2	lminsurf	16	rosenbr	10	yfitu	3
broyden3d	10	dixmaani	12	gulf	4	mancino	10	s308	2	zangwill2	2
broydenbd	10	dixmaanj	12	hairy	2	mexhat	2	sensors	10	zangwill3	3
chandheu	10	dixmaank	12	heart6ls	6	meyer3	3	schmvett	3		

Table 1: The OPM small test problems and their dimension

B.2 Deep Learning Datasets

Problem	n	Problem	n	Problem	n	Problem	n	Problem	n	Problem	n
arglina	400	crglvy	400	dixmaanj	600	fminsurf	400	ncb20c	500	tcontact	400
arglinb	50	cube	500	dixmaank	600	freuroth	500	nlminsurf	400	tquartic	500
arglinc	50	curly10	500	dixmaanl	600	helix	500	nondquar	500	tridia	500
argtrig	50	deconvu	51	dixon	500	hilbert	500	nzf1	520	tlminsurfx	400
arwhead	500	dixmaana	600	dqrtic	500	hydc20ls	99	penalty1	500	tnlminsurfx	400
bdarwhd	500	dixmaanb	600	edensch	500	indef	500	penalty2	100	vardim	500
brownal	500	dixmaanc	600	eg2	400	integreq	500	penalty3	500	wmsqrtals	400
broyden3d	500	dixmaand	600	eg2s	400	lminsurf	400	powellsg	500	wmsqrtbls	400
broydenbd	500	dixmaane	600	eigenals	110	msqrtals	400	powr	500	woods	500
chandheu	500	dixmaanf	600	eigenbls	110	msqrtbls	400	rosenbr	100		
chebyqad	150	dixmaang	600	eigencls	110	morebv	500	sensors	100		
clplatea	400	dixmaanh	600	engval1	500	ncb20	500	scosine	500		
clplateb	400	dixmaani	600	extrosnb	500	ncb20b	500	spmsqrt	997		

Table 2: The OPM medium-size test problems and their dimension

Problem	n	Problem	n	Problem	n	Problem	n	Problem	n
arwhead	2000	dixmaand	2400	eg2	1600	integreq	2000	powellsg	2000
bdarwhd	2000	dixmaane	2400	eg2s	1600	lminsurf	4900	powr	2000
broyden3d	2000	dixmaanf	2400	eigenals	2550	msqrtals	1600	rosenbr	2000
broydenbd	2000	dixmaang	2400	eigenbls	2550	msqrtbls	1600	spmsqrt	1498
clplatea	4900	dixmaanh	2400	eigencls	2550	morebv	5000	tcontact	4900
clplateb	4800	dixmaani	2400	engval1	2000	ncb20b	2000	tquartic	2000
crglvy	4000	dixmaanj	2400	extrosnb	2000	ncb20c	2000	tridia	2000
cube	2000	dixmaank	2400	fminsurf	4900	nlminsurf	4900	tlminsurfx	4900
curly10	1000	dixmaanl	2400	freuroth	2000	nondquar	2000	tnlminsurfx	4900
dixmaana	2400	dixon	2000	helix	2000	nzf1	2600	vardim	2000
dixmaanb	2400	dqrtic	2000	hilbert	2000	penalty1	2000	woods	2000
dixmaanc	2400	edensch	2000	indef	2000	penalty3	2000		

Table 3: The OPM largish test problems and their dimension

Dataset	Classes	Images	Image Size	Channels	Train Size	Test Size	Comments	Creator
CIFAR-10	10	60,000	32x32	RGB	50,000	10,000	Common benchmark	A. Krizhevsky et al.[1]
CIFAR-100	100	60,000	32x32	RGB	50,000	10,000	Like Cifar-10, more challenging	A. Krizhevsky et al.[2]
SVHN	10	604,388	32x32	RGB	73,257	26,032	House number	[218]
Fashion MNIST	10	70,000	28x28	Grayscale	60,000	10,000	Clothing items	Zalando SE[209]

Table 4: Characteristics of Deep Learning Datasets

Bibliography

[1] N. Agarwal, N. Boumal, B. Bullins, and C. Cartis. Adaptive regularization with cubics on manifolds. *Mathematical Programming*, 188(1):85–134, 2020.

[2] E. L. Allgower, K. Böhmer, F. A. Potra, and W. C. Rheinboldt. A mesh-independence principle for operator equations and their discretizations. *SIAM Journal on Numerical Analysis*, 23(1):160–169, 1986.

[3] V. S. Amaral, R. Andreani, E. G. Birgin, D. S. Marcondes, and J. M. Martínez. On complexity and convergence of high-order coordinate descent algorithms for smooth nonconvex box-constrained minimization. *Journal of Global Optimization*, 84(3):527–561, 2022.

[4] H. Asi, J. Duchi, A. Fallah, O. Javidbakht, and K. Talwar. Private adaptive gradient methods for convex optimization. In *Proceedings in the International Conference on Machine Learning (ICML2021)*, 2021.

[5] A. Attia and T. Koren. SGD with AdaGrad stepsizes: Full adaptivity with high probability to unknown parameters, unbounded gradients and affine variance. arxiv:2302.08783, 2023.

[6] I. Babuschkin, K. Baumli, A. Bell, S. Bhupatiraju, J. Bruce, P. Buchlovsky, D. Budden, T. Cai, A. Clark, I. Danihelka, A. Dedieu, C. Fantacci, J. Godwin, C. Jones, R. Hemsley, T. Hennigan, M. Hessel, S. Hou, S. Kapturowski, T. Keck, I. Kemaev, M. King, M. Kunesch, L. Martens, H. Merzic, V. Mikulik, T. Norman, G. Papamakarios, J. Quan, R. Ring, F. Ruiz, A. Sanchez, R. Schneider, E. Sezener, S. Spencer, S. Srinivasan, W. Stokowiec, L. Wang, G. Zhou, and F. Viola. Optax a gradient processing and optimization library for JAX, 2020. URL `http://github.com/deepmind/optax`.

[7] A. S. Bandeira, K. Scheinberg, and L. N. Vicente. Convergence of trust-region methods based on probabilistic models. *SIAM Journal on Optimization*, 24(3):1238–1264, 2014.

[8] J. Barzilai and J. Borwein. Two-point step size gradient method. *IMA Journal of Numerical Analysis*, 8:141–148, 1988.

[9] A. Beck. *First-order Methods in Optimization*. Number 25 in MOS-SIAM Optimization Series. SIAM, Philadelphia, USA, 2017.

[10] A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

[11] S. Bellavia and G. Gurioli. Stochastic analysis of an adaptive cubic regularization method under inexact gradient evaluations and dynamic hessian accuracy. *Optimization*, 71(1):227–261.

[12] S. Bellavia, G. Gurioli, B. Morini, and P. L. Toint. Adaptive regularization algorithms with inexact evaluations for nonconvex optimization. *SIAM Journal on Optimization*, 29(4):2881–2915, 2019.

[13] S. Bellavia, N. Krejić, and N. Krklec Jerinkić. Subsampled inexact Newton methods for minimizing large sums of convex functions. *IMA Journal of Numerical Analysis*, 40 (4):2309–2341, 2019.

[14] S. Bellavia, N. Krejić, and B. Morini. Inexact restoration with subsampled trust-region methods for finite-sum minimization. *Computational Optimization and Applications*, 76 (3):701–736, 2020.

[15] S. Bellavia, G. Gurioli, B. Morini, and P. Toint. Adaptive regularization for nonconvex optimization using inexact function values and randomly perturbed derivatives. *Journal of Complexity*, 68:91–105, 2022.

[16] S. Bellavia, N. Krejić, B. Morini, and S. Rebegoldi. A stochastic first-order trust-region method with inexact restoration for finite-sum minimization. *Computational Optimization and Applications*, 84(1):53–84, 2022.

[17] S. Bellavia, B. Morini, and S. Rebegoldi. On the convergence properties of a stochastic trust-region method with inexact restoration. *Axioms*, 12(1):38, 2022.

[18] S. Bellavia, G. Gurioli, B. Morini, and P. L. Toint. The impact of noise on evaluation complexity: The deterministic trust-region case. *Journal of Optimization Theory and Applications*, 196(2):700–729, 2023. doi: 10.1007/s10957-022-02153-5.

[19] H. Y. Benson and D. F. Shanno. Cubic regularization in symmetric rank-1 quasi-newton methods. *Mathematical Programming Computation*, 10(4):457–486, 2018.

[20] A. S. Berahas, L. Cao, and K. Scheinberg. Global convergence rate analysis of a generic line search algorithm with noise. *SIAM Journal on Optimization*, 31(2):1489–1518, 2021.

[21] E. Bergou, Y. Diouane, and S. Gratton. On the use of the energy norm in trust-region and adaptive cubic regularization subproblems. *Computational Optimization and Applications*, 68(3):533–554, 2017.

[22] E. H. Bergou, Y. Diouane, and S. Gratton. A line-search algorithm inspired by the adaptive cubic regularization framework and complexity analysis. *Journal of Optimization Theory and Applications*, 178(3):885–913, 2018.

[23] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, Massachussetts, USA, 1995.

[24] T. Bianconcini and M. Sciandrone. A cubic regularization algorithm for unconstrained optimization using line search and nonmonotone techniques. *Optimization Methods and Software*, 31(5):1008–1035, 2016.

[25] E. G. Birgin and J. M. Martínez. *Practical Augmented Lagrangian Methods for Constrained Optimization*. Society for Industrial and Applied Mathematics, May 2014.

[26] E. G. Birgin and J. M. Martínez. The use of quadratic regularization with a cubic descent condition for unconstrained optimization. *SIAM Journal on Optimization*, 27 (2):1049–1074, 2017.

[27] E. G. Birgin and J. M. Martínez. A newton-like method with mixed factorizations and cubic regularization for unconstrained minimization. *Computational Optimization and Applications*, 73(3):707–753, Mar. 2019. doi: 10.1007/s10589-019-00089-7. URL `https://doi.org/10.1007/s10589-019-00089-7`.

[28] E. G. Birgin and J. M. Martínez. Block coordinate descent for smooth nonconvex constrained minimization. *Computational Optimization and Applications*, 83(1):1–27, 2022.

[29] E. G. Birgin, J. L. Gardenghi, J. M. Martínez, S. A. Santos, and P. L. Toint. Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models. *Mathematical Programming*, 163(1-2):359–368, Aug. 2016. doi: 10.1007/s10107-016-1065-8.

[30] E. G. Birgin, N. Krejić, and J. M. Martínez. On the employment of inexact restoration for the minimization of functions whose evaluation is subject to errors. *Mathematics of Computation*, 87(311):1307–1326, 2017.

[31] E. G. Birgin, J. L. Gardenghi, J. M. Martínez, and S. A. Santos. On the use of third-order models with fourth-order regularization for unconstrained optimization. *Optimization Letters*, 14(4):815–838, 2019.

[32] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007.

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

[34] I. Bongartz, A. R. Conn, N. Gould, and P. L. Toint. Cute: Constrained and unconstrained testing environment. *ACM Trans. Math. Softw.*, 21(1):123–160, 1995.

[35] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

[36] L. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1967.

[37] S. Bubeck, Q. Jiang, Y. T. Lee, Y. Li, and A. Sidford. Near-optimal method for highly smooth convex optimization. In *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 492–507, 2019.

[38] B. Bullins and K. A. Lai. Higher-order methods for convex-concave min-max optimization and monotone variational inequalities. *SIAM Journal on Optimization*, 32(3): 2208–2229, 2022.

[39] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[40] L. Calderón, M. A. Diniz-Ehrhardt, and J. M. Martínez. On high-order model regularization for multiobjective optimization. *Optimization Methods and Software*, 37(1): 175–191, 2020.

[41] E. J. Candes, X. Li, and M. Soltanolkotabi. Phase retrieval via wirtinger flow: Theory and algorithms. *IEEE Transactions on Information Theory*, 61(4):1985–2007, 2015.

[42] L. Cao, A. S. Berahas, and K. Scheinberg. First- and second-order high probability complexity bounds for trust-region methods with noisy oracles. *Mathematical Programming*, 2023.

[43] Y. Carmon and J. Duchi. Gradient descent finds the cubic-regularized nonconvex newton step. *SIAM Journal on Optimization*, 29(3):2146–2178, 2019.

[44] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. "Convex until proven guilty": Dimension-free acceleration of gradient descent on non-convex functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 654–663. PMLR, 2017.

[45] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points i. *Mathematical Programming*, 184(1-2):71–120, 2019.

[46] C. Cartis and K. Scheinberg. Global convergence rate analysis of unconstrained optimization methods based on probabilistic models. *Mathematical Programming*, 169(2): 337–375, 2017.

[47] C. Cartis and W. Zhu. Second-order methods for quartically-regularised cubic polynomials, with applications to high-order tensor methods.

[48] C. Cartis, N. I. M. Gould, and P. L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. part II: worst-case function- and derivative-evaluation complexity. *Mathematical Programming*, 130(2):295–319, 2010.

[49] C. Cartis, N. I. M. Gould, and P. L. Toint. On the complexity of steepest descent, Newton's and regularized Newton's methods for nonconvex unconstrained optimization. *SIAM Journal on Optimization*, 20(6):2833–2852, 2010.

[50] C. Cartis, N. I. Gould, and P. L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. part I: Motivation, convergence and numerical results. *Math. Program.*, 127(2):245–295, 2011.

[51] C. Cartis, N. I. M. Gould, and P. L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. part II: worst-case function- and derivative-evaluation complexity. *Math. Program.*, 130(2):295–319, 2011.

[52] C. Cartis, N. I. M. Gould, and P. L. Toint. Worst-case evaluation complexity and optimality of second-order methods for nonconvex smooth optimization. In B. Sirakov, P. de Souza, and M. Viana, editors, *Invited Lectures, Proceedings of the 2018 International Conference of Mathematicians (ICM 2018), vol. 4, Rio de Janeiro*, pages 3729–3768. World Scientific Publishing Co Pte Ltd, 2018.

[53] C. Cartis, N. I. M. Gould, and P. L. Toint. Universal regularization methods: Varying the power, the smoothness and the accuracy. *SIAM Journal on Optimization*, 29(1): 595–615, 2019.

[54] C. Cartis, N. I. M. Gould, and P. L. Toint. A concise second-order complexity analysis for unconstrained optimization using high-order regularized models. *Optimization Methods and Software*, 35(2):243–256, 2019.

[55] C. Cartis, N. I. M. Gould, and P. L. Toint. Sharp worst-case evaluation complexity bounds for arbitrary-order nonconvex optimization with inexpensive constraints. *SIAM Journal on Optimization*, 30(1):513–541, Jan. 2020.

[56] C. Cartis, J. Fowkes, and Z. Shao. Randomised subspace methods for non-convex optimization, with applications to nonlinear least-squares, 2022.

[57] C. Cartis, N. I. M. Gould, and P. L. Toint. *Evaluation complexity of algorithms for nonconvex optimization*. MOS-SIAM Series on Optimization. Society for Industrial & Applied Mathematics, New York, NY, Apr. 2022.

[58] K. Chakrabarti and N. Chopra. Generalized AdaGrad (G-AdaGrad) and Adam: A state-space perspective. arXiv:2106.00092, 2021.

[59] I. Chatzigeorgiou. Bounds on the Lambert function and their application to the outage analysis of user cooperation. *IEEE Communications Letters*, 17(8):1505–1508, 2013.

[60] E. M. Chayti, N. Doikov, and M. Jaggi. Unified convergence theory of stochastic and variance-reduced cubic newton methods, 2023.

[61] J. Chen, D. Zhou, Y. Tang, Z. Yang, Y. Cao, and Q. Gu. Closing the generalization gap of adaptive gradient methods in training deep neural networks, 7 2020.

[62] R. Chen, M. Menickelly, and K. Scheinberg. Stochastic optimization using a trust-region method and random models. *Mathematical Programming*, 169(2):447–487, 2017.

[63] X. Chen and P. L. Toint. High-order evaluation complexity for convexly-constrained optimization with non-lipschitzian group sparsity terms. *Mathematical Programming*, 187(1-2):47–78, 2020.

[64] X. Chen, B. Jiang, T. Lin, and S. Zhang. Accelerating adaptive cubic regularization of newton's method via random sampling. *Journal of Machine Learning Research*, 23(90): 1–38, 2022.

[65] F. Clarke. *Functional Analysis, Calculus of Variations and Optimal Control*. Springer London, 2013.

[66] K. L. Clarkson and D. P. Woodruff. Low-rank approximation and regression in input sparsity time. *J. ACM*, 63(6), jan 2017.

[67] E. S. Coakley and V. Rokhlin. A fast divide-and-conquer algorithm for computing the spectra of real symmetric tridiagonal matrices. *Applied and Computational Harmonic Analysis*, 34(3):379–414, May 2013.

[68] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust-Region Methods*. Number 1 in MOS-SIAM Optimization Series. SIAM, Philadelphia, USA, 2000.

[69] A. R. Conn, K. Scheinberg, and L. N. Vicente. Global convergence of general derivative-free trust-region algorithms to first- and second-order critical points. *SIAM Journal on Optimization*, 20(1):387–415, Jan. 2009.

[70] R. M. Corless, G. H. Gonnet, D. E. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. *Advances in Computational Mathematics*, 5:329—359, 1996.

[71] F. E. Curtis and D. P. Robinson. Exploiting negative curvature in deterministic and stochastic optimization. *Mathematical Programming*, 176(1-2):69–94, 2018.

[72] F. E. Curtis and Q. Wang. Worst-case complexity of trace with inexact subproblem solutions for nonconvex smooth optimization. *SIAM Journal on Optimization*, 33(3): 2191–2221, Aug. 2023.

[73] F. E. Curtis, Z. Lubberts, and D. P. Robinson. Concise complexity analyses for trust region methods. *Optimization Letters*, 12(8):1713–1724, 2018.

[74] F. E. Curtis, D. P. Robinson, and M. Samadi. An inexact regularized newton framework with a worst-case iteration complexity of $ {\mathscr o}(\varepsilon^{-3/2}) $ for nonconvex optimization. *IMA Journal of Numerical Analysis*, 39(3):1296–1327, 2018.

[75] F. E. Curtis, D. P. Robinson, C. W. Royer, and S. J. Wright. Trust-region newton-CG with strong second-order complexity guarantees for nonconvex optimization. *SIAM Journal on Optimization*, 31(1):518–544, 2021.

[76] T. A. Davis. *Direct Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Jan. 2006.

[77] A. Défossez, L. Bottou, F. Bach, and N. Usunier. A simple convergence proof for Adam and Adagrad. *Transactions on Machine Learning Research*, October 2022.

[78] P. Deuflhard and F. A. Potra. Asymptotic mesh independence of newton–galerkin methods via a refined mysovskii theorem. *SIAM Journal on Numerical Analysis*, 29(5): 1395–1412, 1992.

[79] N. Doikov. Minimizing quasi-self-concordant functions by gradient regularization of newton method, 2023.

[80] N. Doikov and Y. Nesterov. Inexact tensor methods with dynamic accuracies. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 2577–2586. PMLR, 2020.

[81] N. Doikov and Y. Nesterov. High-order optimization methods for fully composite problems. *SIAM Journal on Optimization*, 32(3):2402–2427, 2022.

[82] N. Doikov and Y. Nesterov. Gradient regularization of newton method with bregman distances. *Mathematical Programming*, Mar. 2023.

[83] N. Doikov, P. Richtarik, and U. of Edinburgh. Randomized block cubic Newton method. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1290–1298, 10–15 Jul 2018.

[84] N. Doikov, K. Mishchenko, and Y. Nesterov. Super-universal regularized newton method, 2022.

[85] N. Doikov, E. M. Chayti, and M. Jaggi. Second-order optimization with lazy hessians. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 8138–8161. PMLR, 23–29 Jul 2023.

[86] E. D. Dolan, J. J. Moré, and T. S. Munson. Optimality measures for performance profiles. *SIAM Journal on Optimization*, 16(3):891–909, 2006.

[87] R.-A. Dragomir, A. d'Aspremont, and J. Bolte. Quartic first-order methods for low-rank minimization. *Journal of Optimization Theory and Applications*, 189(2):341–363, Mar. 2021.

[88] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12, July 2011.

[89] J. Duchi, M. I. Jordan, and B. Brendan. Estimation, optimization, and parallelism when data is sparse. In *Advances in Neural Information Processing Systems (Neurips2013)*, 2013.

[90] J.-P. Dussault. ARCq: a new adaptive regularization by cubics. *Optimization Methods and Software*, 33(2):322–335, 2017.

[91] K. S. F. E. Curtis and R. Shi. A stochastic trust region algorithm based on careful step normalization. *INFORMS Journal on Optimization*, 1(3):200–220, 2019.

[92] J. Fan and Y. Yuan. A new trust region algorithm with trust region radius converging to zero. In *Proceedings of the 5th International Conference on Optimization: Techniques and Applications (ICOTA 2001, Hong Kong)*, pages 786–794, 2001.

[93] M. Faw, I. Tziotis, C. Caramanis, A. Mokhtari, S. Shakkottai, and R. Ward. The power of adaptivity in SGD: Self-tuning step sizes with unbounded gradients and affine variance. In *Proceedings of 35th Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 313–355, 2022.

[94] M. Faw, L. Rout, C. Caramanis, and S. Shakkottai. Beyond uniform smoothness: A stopped analysis of adaptive SGD. arxiv:2302.06570, 2023.

[95] M. C. Ferris, S. Lucid, and M. Roma. Nonmonotone curvilinear line search methods for unconstrained optimization. *Computational Optimization and Applications*, 6(2): 117–136, 1996.

[96] A. Gasnikov, P. Dvurechensky, E. Gorbunov, E. Vorontsova, D. Selikhanovych, and C. A. Uribe. Optimal tensor methods in smooth convex and uniformly convexoptimization. In *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 1374–1391. PMLR, 2019.

[97] R. Ge, J. D. Lee, and T. Ma. Matrix completion has no spurious local minimum. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, page 2981–2989. Curran Associates Inc., 2016.

[98] S. Ghadimi and G. Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, Jan 2013.

[99] I. Gitman and B. Ginsburg. Comparison of batch normalization and weight normalization algorithms for the large-scale image classification, 2017.

[100] S. M. Goldfeld, R. E. Quandt, and H. F. Trotter. Maximization by quadratic hill-climbing. *Econometrica*, 34(3):541, 1966.

[101] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[102] N. I. M. Gould, S. Lucidi, M. Roma, and P. L. Toint. Solving the trust-region subproblem using the lanczos method. *SIAM Journal on Optimization*, 9(2):504–525, 1999.

[103] N. I. M. Gould, S. Lucidi, M. Roma, and P. L. Toint. Exploiting negative curvature directions in linesearch methods for unconstrained optimization. *Optimization Methods and Software*, 14(1-2):75–98, 2000.

[104] N. I. M. Gould, D. Orban, and P. L. Toint. GALAHAD, a library of thread-safe fortran 90 packages for large-scale nonlinear optimization. *ACM Transactions on Mathematical Software*, 29(4):353–372, Dec. 2003.

[105] N. I. M. Gould, M. Porcelli, and P. L. Toint. Updating the regularization parameter in the adaptive cubic regularization algorithm. *Computational Optimization and Applications*, 53(1):1–22, Dec. 2012.

[106] N. I. M. Gould, D. Orban, and P. L. Toint. CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. *Computational Optimization and Applications*, 60(3):545–557, 2015.

[107] G. N. Grapiglia and Y. Nesterov. Tensor methods for minimizing convex functions with hölder continuous higher-order derivatives. *SIAM Journal on Optimization*, 30(4): 2750–2779, 2020.

[108] G. N. Grapiglia and Y. Nesterov. Adaptive third-order methods for composite convex optimization. *SIAM Journal on Optimization*, 33(3):1855–1883, 2023.

[109] G. N. Grapiglia and G. F. D. Stella. An adaptive trust-region method without function evaluation. *Computational Optimization and Applications*, 82:31–60, 2022.

[110] S. Gratton and P. L. Toint. Opm, a collection of optimization problems in matlab, 2021.

[111] S. Gratton and P. L. Toint. Adaptive regularization minimization algorithms with nonsmooth norms. *IMA Journal of Numerical Analysis*, 03 2022.

[112] S. Gratton, A. Sartenaer, and P. L. Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, 19(1):414–444, 2008.

[113] S. Gratton, C. W. Royer, L. N. Vicente, and Z. Zhang. Complexity and global rates of trust-region methods based on probabilistic models. *IMA Journal of Numerical Analysis*, 38(3):1579–1597, Aug. 2017.

[114] S. Gratton, S. Jerad, and P. L. Toint. Complexity of a class of first-order objective-function-free optimization algorithms. arXiv:2203.01647v3, 2022. 'Under review in OMS'.

[115] S. Gratton, S. Jerad, and P. L. Toint. First-order objective-function-free optimization algorithms and their complexity, 2022.

[116] S. Gratton, S. Jerad, and P. L. Toint. Convergence properties of an objective-function-free optimization regularization algorithm, including an $\mathcal{O}(\epsilon^{-3/2})$ complexity bound. *SIAM Journal on Optimization*, 33(3):1621–1646, 2023.

[117] S. Gratton, S. Jerad, and P. L. Toint. Yet another fast variant of newton's method for nonconvex optimization. arXiv:2302.10065, 2023. 'Under review in IMAJNA'.

[118] S. Gratton, S. Jerad, and P. L. Toint. An adaptive regularization method in banach spaces. *Optimization Methods and Software*, pages 1–17, June 2023.

[119] S. Gratton, A. Kopanicakova, and P. L. Toint. Multilevel objective-function-free optimization with an application to neural networks training, 2023.

[120] A. Griewank. The modification of Newton's method for unconstrained optimization by bounding cubic terms, 1981.

[121] F. Hamad and O. Hinder. A consistently adaptive trust-region method. In *Advances in Neural Information Processing Systems*, volume 35, pages 6640–6653. Curran Associates, Inc., 2022.

[122] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[123] M. Heinkenschloss. Mesh independence for nonlinear least squares problems with norm constraints. *SIAM Journal on Optimization*, 3(1):81–117, 1993.

[124] T. Hennigan, T. Cai, T. Norman, L. Martens, and I. Babuschkin. Haiku: Sonnet for JAX, 2020. URL `http://github.com/deepmind/dm-haiku`.

[125] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289, Oct. 2021.

[126] R. C. James. Reflexivity and the sup of linear functionals. *Israel Journal of Mathematics*, 13(3-4):289–300, 1972.

[127] B. Jiang, H. Wang, and S. Zhang. An optimal high-order tensor method for convex optimization. In *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99, pages 1799–1801. PMLR, 25–28 Jun 2019.

[128] B. Jiang, T. Lin, and S. Zhang. A unified adaptive tensor approximation scheme to accelerate composite convex optimization. *SIAM Journal on Optimization*, 30(4):2897–2926, Jan. 2020.

[129] R. Jiang and A. Mokhtari. Generalized optimistic methods for convex-concave saddle point problems, 2022.

[130] B. Jin, K. Scheinberg, and M. Xie. High probability complexity bounds for line search based on stochastic oracles. In *Advances in Neural Information Processing Systems*, volume 34, pages 9193–9203. Curran Associates, Inc., 2021.

[131] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596 (7873):583–589, July 2021.

[132] C. T. Kelley and E. W. Sachs. Quasi-newton methods and unconstrained optimal control problems. *SIAM Journal on Control and Optimization*, 25(6):1503–1516, 1987.

[133] L. Kfir, Y. Alp, and C. Volkan. Online adaptive methods, universality and acceleration. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[134] A. Khaled and P. Richtárik. Better theory for SGD in the nonconvex world. *Transactions on Machine Learning Research*, 2023.

[135] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings in the International Conference on Learning Representations (ICLR)*, 2015.

[136] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtarik. Federated optimization: Distributed machine learning for on-device intelligence, 2016. URL `https://arxiv.org/abs/1610.02527`.

[137] S. Kong and A. S. Lewis. The cost of nonconvexity in deterministic nonsmooth optimization, 2022.

[138] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

[139] C. Lemarechal. Cauchy and the gradient method. *Doc Math Extra*, pages 251–254, 2012.

[140] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, 1944.

[141] X. Li and F. Orabona. On the convergence of stochastic gradient descent with adaptive stepsizes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, page 983–992, 2019.

[142] T. Lin and M. I. Jordan. Perseus: A simple and optimal high-order method for variational inequalities, 2023.

[143] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming, Series B*, 45(1):503–528, 1989.

[144] H. Liu, T. Yao, R. Li, and Y. Ye. Folded concave penalized sparse linear regression: sparsity, statistical performance, and algorithmic theory for local solutions. *Mathematical Programming*, 166(1-2):207–240, 2017.

[145] M. Liu, Z. Li, X. Wang, J. Yi, and T. Yang. Adaptive negative curvature descent with applications in non-convex optimization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[146] Y. Liu and F. Roosta. A newton-mr algorithm with complexity guarantees for nonconvex smooth unconstrained optimization, 2022.

[147] N. Loizou, S. Vaswani, I. Hadj Laradji, and S. Lacoste-Julien. Stochastic polyak stepsize for sgd: An adaptive learning rate for fast convergence. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130, pages 1306–1314, 13–15 Apr 2021.

[148] I. Loshchilov and F. Hutter. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations*, 2017.

[149] S. Lucidi, F. Rochetich, and M. Roma. Curvilinear stabilization techniques for truncated newton methods in large scale unconstrained optimization. *SIAM Journal on Optimization*, 8(4):916–939, 1998.

[150] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[151] J. M. Martínez. On high-order model regularization for constrained optimization. *SIAM Journal on Optimization*, 27(4):2447–2458, 2017.

[152] J. M. Martínez and M. Raydan. Cubic-regularization counterpart of a variable-norm trust-region method for unconstrained minimization. *Journal of Global Optimization*, 68(2):367–385, 2016.

[153] G. P. McCormick. A modification of armijos step-size rule for negative curvature. *Mathematical Programming*, 13(1):111–115, 1977.

[154] B. McMahan and M. Streeter. Adaptive bound optimization for online convex optimization. In *Conference on Learning Theory*, page 244sq, 2010.

[155] X. Meng and M. W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, page 91–100. Association for Computing Machinery, 2013.

[156] K. Mishchenko. Regularized newton method with global $o(1/k^2)$ convergence. *SIAM Journal on Optimization*, 33(3):1440–1462, July 2023.

[157] J. J. Moré and D. C. Sorensen. On the use of directions of negative curvature in a modified newton method. *Mathematical Programming*, 16(1):1–20, 1979.

[158] J. J. Moré and G. Toraldo. Algorithms for bound constrained quadratic programming problems. *Numerische Mathematik*, 14:14–21, 1989.

[159] J. J. Moré and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1(1):93–113, 1991.

[160] M. C. Mukkamala and M. Hein. Variants of RMSProp and Adagrad with logarithmic regret bounds. In *Proceedings of the 34th International Conference on Machine Learning*, page 2545–2553, 2017.

[161] K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39(2):117–129, 1987.

[162] Y. Nabou and I. Necoara. Efficiency of higher-order algorithms for minimizing general composite optimization, 2022.

[163] Y. Nesterov. *Introductory Lectures on Convex Optimization*. Applied Optimization. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004.

[164] Y. Nesterov. Accelerating the cubic regularization of newton's method on convex problems. *Mathematical Programming*, 112(1):159–181, Jan. 2007.

[165] Y. Nesterov. *Lectures on Convex Optimization*. Springer International Publishing, 2018.

[166] Y. Nesterov. Implementable tensor methods in unconstrained convex optimization. *Mathematical Programming*, 186(1-2):157–183, Nov. 2019.

[167] Y. Nesterov. Inexact basic tensor methods for some classes of convex optimization problems. *Optimization Methods and Software*, 37(3):878–906, 2020.

[168] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, Jan. 1994.

[169] Y. Nesterov and B. Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[170] T. D. Niri, M. Heydari, and M. M. Hosseini. An improvement of adaptive cubic regularization method for unconstrained optimization problems. *International Journal of Computer Mathematics*, 98(2):271–287, 2020.

[171] T. D. Niri, M. Heydari, and M. M. Hosseini. Two modified adaptive cubic regularization algorithms by using the nonmonotone armijo-type line search. *Optimization*, pages 1–24, May 2022. doi: 10.1080/02331934.2022.2075746. URL `https://doi.org/10.1080/02331934.2022.2075746`.

[172] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, 2e edition, 2006.

[173] F. Orabona and D. Pal. Scale-free algorithms for online linear optimization. ALT, 2015.

[174] A. Orvieto, S. Lacoste-Julien, and N. Loizou. Dynamics of SGD with stochastic polyak stepsizes: Truly adaptive variants and convergence to exact solution. In *Advances in Neural Information Processing Systems*, 2022.

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

[176] C. Paquette, H. Lin, D. Drusvyatskiy, J. Mairal, and Z. Harchaoui. Catalyst for gradient-based nonconvex optimization. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 613–622, 2018.

[177] V. Patel and A. S. Berahas. Gradient descent in the absence of global lipschitz continuity of the gradients, 2023.

[178] B. T. Polyak. *Introducton to optimization.* Optimization Software, Inc., New York, 1987.

[179] M. Porcelli and P. L. Toint. A note on using performance and data profiles for training algorithms. *ACM Transactions on Mathematical Software*, 45(2):1–25, 2019.

[180] M. J. Quinn. *Parallel Computing (2nd Ed.): Theory and Practice.* McGraw-Hill, Inc., USA, 1994.

[181] M. Raissi, P. Perdikaris, and G. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, Feb. 2019.

[182] S. Reddi, S. Kale, and S. Kumar. On the convergence of Adam and beyond. In *Proceedings in the International Conference on Learning Representations (ICLR)*, 2018.

[183] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, Sept. 1951.

[184] C. W. Royer and S. J. Wright. Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. *SIAM Journal on Optimization*, 28(2): 1448–1477, 2018.

[185] C. W. Royer, M. O'Neill, and S. J. Wright. A newton-cg algorithm with complexity guarantees for smooth unconstrained optimization. *Mathematical Programming*, (1–2): 451–488, Jan. 2019.

[186] Y. Saad. *Iterative Methods for Sparse Linear Systems.* Society for Industrial and Applied Mathematics, Jan. 2003.

[187] F. Schaipp, R. M. Gower, and M. Ulbrich. A stochastic proximal polyak step size. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

[188] K. Scheinberg and M. Xie. Stochastic Adaptive Regularization Method with Cubics: A High Probability Complexity Bound. In *OPT2022: 14th Annual Workshop on Optimization for Machine Learning*, 2022.

[189] R. M. Schmidt, F. Schneider, and P. Hennig. Descending through a crowded valley - benchmarking deep learning optimizers. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 9367–9376, 2021.

[190] Z. Shao. *On Random Embeddings and Their Application to Optimization*. PhD thesis, University of Oxford, Oxford, UK, 2021.

[191] Z. Shao and C. Cartis. Random-subspace adaptive cubic regularisation method for nonconvex optimisation, 2022.

[192] T. Tieleman and G. Hinton. Lecture 6.5-RMSPROP. COURSERA: Neural Networks for Machine Learning, 2012.

[193] P. L. Toint. Global convergence of a a of trust-region methods for nonconvex minimization in hilbert space. *IMA Journal of Numerical Analysis*, 8(2):231–252, 1988.

[194] C. Traoré and E. Pauwels. Sequential convergence of AdaGrad algorithm for smooth convex optimization. *Operations Research Letters*, 49(4):452–458, 2021.

[195] N. Tripuraneni, M. Stern, C. Jin, J. Regier, and M. I. Jordan. Stochastic cubic regularization for fast nonconvex optimization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 2904–2913, 2018.

[196] K. Ueda and N. Yamashita. Convergence properties of the regularized newton method forhenconstrained nonconvex optimization. *Applied Mathematics and Optimization*, 62 (1):27–46, Dec. 2009.

[197] K. Ueda and N. Yamashita. A regularized newton method without line search for unconstrained optimization. *Computational Optimization and Applications*, 59(1-2): 321–351, 2014.

[198] M. Ulbrich and S. Ulbrich. Superlinear convergence of affine-scaling interior-point newton methods for infinite-dimensional nonlinear problems with pointwise bounds. *SIAM Journal on Control and Optimization*, 38(6):1938–1984, 2000.

[199] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[200] S. Vaswani, I. Laradji, F. Kunstner, S. Y. Meng, M. Schmidt, and S. Lacoste-Julien. Adaptive gradient methods converge faster with over-parameterization (but you should do a line-search). arXiv:2006.06835, 2020.

[201] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2005.

[202] J. Wang, Z. Charles, Z. Xu, G. Joshi, H. B. McMahan, B. A. y Arcas, M. Al-Shedivat, G. Andrew, S. Avestimehr, K. Daly, D. Data, S. Diggavi, H. Eichner, A. Gadhikar, Z. Garrett, A. M. Girgis, F. Hanzely, A. Hard, C. He, S. Horvath, Z. Huo, A. Ingerman, M. Jaggi, T. Javidi, P. Kairouz, S. Kale, S. P. Karimireddy, J. Konecny, S. Koyejo,

T. Li, L. Liu, M. Mohri, H. Qi, S. J. Reddi, P. Richtarik, K. Singhal, V. Smith, M. Soltanolkotabi, W. Song, A. T. Suresh, S. U. Stich, A. Talwalkar, H. Wang, B. Woodworth, S. Wu, F. X. Yu, H. Yuan, M. Zaheer, M. Zhang, T. Zhang, C. Zheng, C. Zhu, and W. Zhu. A field guide to federated optimization, 2021.

[203] Z. Wang, Y. Zhou, Y. Liang, and G. Lan. Stochastic variance-reduced cubic regularization for nonconvex optimization. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 2731–2740, 2019.

[204] R. Ward, X. Wu, and L. Bottou. AdaGrad stepsizes: Sharp convergence over nonconvex landscapes. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 6677–6686, 2019.

[205] P. Wolfe. Convergence conditions for ascent methods. *SIAM Review*, 11(2):226–235, 1969.

[206] P. Wolfe. Convergence conditions for ascent methods. II: Some corrections. *SIAM Review*, 13(2):185–188, 1971.

[207] X. Wu, R. Ward, and L. Bottou. WNGRAD: Learn the learning rate in gradient descent. arXiv:1803.02865, 2018.

[208] Y. xiang Yuan. Recent advances in trust region algorithms. *Mathematical Programming*, 151(1):249–281, Mar. 2015.

[209] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. Comment: Dataset is freely available at https://github.com/zalandoresearch/fashion-mnist Benchmark is available at http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/.

[210] H.-K. Xu. Inequalities in banach spaces with applications. *Nonlinear Analysis: Theory, Methods &amp Applications*, 16(12):1127–1138, 1991.

[211] P. Xu, F. Roosta, and M. W. Mahoney. Newton-type methods for non-convex optimization under inexact hessian information. *Mathematical Programming*, 184(1-2): 35–70, May 2020.

[212] Z.-B. Xu and G. Roach. Characteristic inequalities of uniformly convex and uniformly smooth banach spaces. *Journal of Mathematical Analysis and Applications*, 157(1): 189–210, 1991.

[213] J. Yang, X. Li, and N. He. Nest your adaptive algorithm for parameter-agnostic nonconvex minimax optimization. arXiv:2206.00743, 2022.

[214] Z. Yao, P. Xu, F. Roosta, and M. W. Mahoney. Inexact nonconvex newton-type methods. *INFORMS Journal on Optimization*, 3(2):154–182, 2021.

[215] Z. Yao, P. Xu, F. Roosta, S. J. Wright, and M. W. Mahoney. Inexact newton-CG algorithms with complexity guarantees. *IMA Journal of Numerical Analysis*, 43(3): 1855–1897, 2022.

[216] C. K. Yap. *Fundamental Problems of Algorithmic Algebra*. Oxford University Press, Inc., USA, 1999.

[217] T. J. Ypma. Historical development of the newton–raphson method. *SIAM Review*, 37 (4):531–551, 1995.

[218] A. C. A. B. B. W. Yuval Netzer, Tao Wang and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[219] M. Zeiler. ADADELTA: an adaptive learning rate method. arXiv:1212.5701, 2012.

[220] J. Zhang, T. He, S. Sra, and A. Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. In *International Conference on Learning Representations*, 2020.

[221] D. Zhou, P. Xu, and Q. Gu. Stochastic variance-reduced cubic regularization methods. *Journal of Machine Learning Research*, 20(134):1–47, 2019.

[222] D. Zhou, J. Chen, Y. Tang, Z. Yang, Y. Cao, and Q. Gu. On the convergence of adaptive gradient methods for nonconvex optimization. arXiv:2080.05671, 2020.

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

Université de Toulouse

Titre : Approches du second ordre de d'ordre élevées pour l'optimisation nonconvex avec variantes sans évaluation de la fonction objective

Mots clés : Optimisation nonconvexe, Méthodes d'ordre elevées, Analyse numérique, Méthodes second ordre, calcul hautes performances

Résumé : Même si l'optimisation non linéaire semble (a priori) être un domaine mature, de nouveaux schémas de minimisation sont proposés ou redécouverts pour les problèmes modernes à grande échelle. A titre d'exemple et en rétrospective de la dernière décennie, nous avons vu une vague de méthodes du premier ordre avec différentes analyses, malgré le fait que les limitations théoriques bien connues de ces méthodes ont été discutées en profondeur auparavant. Cette thèse explore deux lignes principales de recherche dans le domaine de l'optimisation non-convexe avec un accent particulier sur les méthodes de second ordre et d'ordre supérieur. Dans la première série de travaux, nous nous concentrons sur les algorithmes qui ne calculent pas les valeurs des fonctions et opèrent sans connaissance d'aucun paramètre, car les méthodes du premier ordre les plus adaptées pour les problèmes modernes apartiennent à cette dernière catégorie. Nous commençons par redéfinir l'algorithme bien connu d'Adagrad dans un cadre de région de confiance et utilisons ce dernier paradigme pour étudier deux classes d'algorithmes OFFO (Objective-Free Function Optimization) déterministes du premier ordre. Pour permettre des algorithmes OFFO exacts plus rapides, nous proposons ensuite une méthode de régularisation adaptative déterministe d'ordre p qui évite le calcul des valeurs de la fonction. Cette approche permet de retrouver la vitesse de convergence bien connu du cadre standard lors de la recherche de points stationnaires, tout en utilisant beaucoup moins d'informations. Dans une deuxième série de travaux, nous analysons les algorithmes adaptatifs dans le cadre plus classique où les valeurs des fonctions sont utilisées pour adapter les paramètres. Nous étendons les méthodes de régularisation adaptatives à une classe spécifique d'espaces de Banach en développant un algorithme de descente du gradient de Hölder. En plus, nous étudions un algorithme de second ordre qui alterne entre la courbure négative et les étapes de Newton avec une vitesse de convergence quasi optimal. Pour traiter les problèmes de grande taille, nous proposons des versions sous-espace de l'algorithme qui montrent des performances numériques prometteuses. Dans l'ensemble, cette recherche couvre un large éventail de techniques d'optimisation et fournit des informations et des contributions précieuses aux algorithmes d'optimisation adaptatifs et sans paramètres pour les fonctions non convexes. Elle ouvre également la voie à des développements théoriques ultérieurs et à l'introduction d'algorithmes numériques plus rapides.

Title: Second Order and High Order Approaches for Nonconvex Optimization with Objective Function-Free Algorithms

Key words: Nonconvex Optimization, High-order methods, Numerical Analysis, Second order methods, High performances calculus

Abstract: Even though nonlinear optimization seems (a priori) to be a mature field, new minimization schemes are proposed or rediscovered for modern large-scale problems. As an example and in retrospect of the last decade, we have seen a surge of first-order methods with different analysis, despite the fact that well-known theoretical limitations of the previous methods have been thoroughly discussed. This thesis explores two main lines of research in the field of nonconvex optimization with a narrow focus on second and higher order methods. In the first series, we focus on algorithms that do not compute function values and operate without knowledge of any parameters, as the most popular currently used first-order methods fall into the latter category. We start by redefining the well-known Adagrad algorithm in a trust-region framework and use the latter paradigm to study two first-order deterministic OFFO (Objective-Free Function Optimization) classes. To enable faster exact OFFO algorithms, we then propose a pth-order deterministic adaptive regularization method that avoids the computation of function values. This approach recovers the well-known convergence rate of the standard framework when searching for stationary points, while using significantly less information. In the second set of papers, we analyze adaptive algorithms in the more classical framework where function values are used to adapt parameters. We extend adaptive regularization methods to a specific class of Banach spaces by developing a Hölder gradient descent algorithm. In addition, we investigate a second-order algorithm that alternates between negative curvature and Newton steps with a near-optimal convergence rate. To handle large problems, we propose subspace versions of the algorithm that show promising numerical performance. Overall, this research covers a wide range of optimization techniques and provides valuable insights and contributions to both parameter-free and adaptive optimization algorithms for nonconvex functions. It also opens the door for subsequent theoretical developments and the introduction of faster numerical algorithms.