

Acceleration Methods

Other titles in Foundations and Trends® in Optimization

Algorithms for Verifying Deep Neural Networks

Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong,
Clark Barrett and Mykel J. Kochenderfer

ISBN: 978-1-68083-786-5

Distributionally Robust Learning

Ruidi Chen and Ioannis Ch. Paschalidis

ISBN: 978-1-68083-772-8

Atomic Decomposition via Polar Alignment: The Geometry of Structured Optimization

Zhenan Fan, Halyun Jeong, Yifan Sun and Michael P. Friedlander

ISBN: 978-1-68083-742-1

Optimization Methods for Financial Index Tracking: From Theory to Practice

Konstantinos Benidis, Yiyong Feng and Daniel P. Palomar

ISBN: 978-1-68083-464-2

The Many Faces of Degeneracy in Conic Optimization

Dmitriy Drusvyatskiy and Henry Wolkowicz

ISBN: 978-1-68083-390-4

Acceleration Methods

Alexandre d'Aspremont

CNRS & Ecole Normale Supérieure, Paris
France
aspremon@ens.fr

Damien Scieur

Samsung SAIT AI Lab & Mila, Montreal
Canada
damien.scieur@gmail.com

Adrien Taylor

INRIA & Ecole Normale Supérieure, Paris
France
adrien.taylor@inria.fr

now

the essence of knowledge

Boston — Delft



Documents

Document Sharing Hub

Foundations and Trends® in Optimization

Published, sold and distributed by:

now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:

now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

A. d'Aspremont et al.. *Acceleration Methods*. Foundations and Trends® in Optimization, vol. 5, no. 1-2, pp. 1–245, 2021.

ISBN: 978-1-68083-929-6

© 2021 A. d'Aspremont et al.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, mechanical, photocopying, recording or otherwise, without prior written permission of the publishers.

Photocopying. In the USA: This journal is registered at the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923. Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by now Publishers Inc for users registered with the Copyright Clearance Center (CCC). The 'services' for users can be found on the internet at: www.copyright.com

For those organizations that have been granted a photocopy license, a separate system of payment has been arranged. Authorization does not extend to other kinds of copying, such as that for general distribution, for advertising or promotional purposes, for creating new collective works, or for resale. In the rest of the world: Permission to photocopy must be obtained from the copyright owner. Please apply to now Publishers Inc., PO Box 1024, Hanover, MA 02339, USA; Tel. +1 781 871 0245; www.nowpublishers.com; sales@nowpublishers.com

now Publishers Inc. has an exclusive license to publish this material worldwide. Permission to use this content must be obtained from the copyright license holder. Please apply to now Publishers, PO Box 179, 2600 AD Delft, The Netherlands, www.nowpublishers.com; e-mail: sales@nowpublishers.com

Foundations and Trends® in Optimization

Volume 5, Issue 1-2, 2021

Editorial Board

Editors-in-Chief

Garud Iyengar

Columbia University, USA

Editors

Dimitris Bertsimas

Massachusetts Institute of Technology

John R. Birge

The University of Chicago

Robert E. Bixby

Rice University

Emmanuel Candes

Stanford University

David Donoho

Stanford University

Laurent El Ghaoui

University of California, Berkeley

Donald Goldfarb

Columbia University

Michael I. Jordan

University of California, Berkeley

Zhi-Quan (Tom) Luo

University of Minnesota, Twin Cities

George L. Nemhauser

Georgia Institute of Technology

Arkadi Nemirovski

Georgia Institute of Technology

Yurii Nesterov

HSE University

Jorge Nocedal

Northwestern University

Pablo A. Parrilo

Massachusetts Institute of Technology

Boris T. Polyak

Institute for Control Science, Moscow

Tamás Terlaky

Lehigh University

Michael J. Todd

Cornell University

Kim-Chuan Toh

National University of Singapore

John N. Tsitsiklis

Massachusetts Institute of Technology

Lieven Vandenbergh

University of California, Los Angeles

Robert J. Vanderbei

Princeton University

Stephen J. Wright

University of Wisconsin

Editorial Scope

Topics

Foundations and Trends® in Optimization publishes survey and tutorial articles in the following topics:

- algorithm design, analysis, and implementation (especially, on modern computing platforms)
- models and modeling systems, new optimization formulations for practical problems
- applications of optimization in machine learning, statistics, and data analysis, signal and image processing, computational economics and finance, engineering design, scheduling and resource allocation, and other areas

Information for Librarians

Foundations and Trends® in Optimization, 2021, Volume 5, 4 issues. ISSN paper version 2167-3888. ISSN online version 2167-3918. Also available as a combined paper and online subscription.

Contents

1	Introduction	2
2	Chebyshev Acceleration	6
2.1	Introduction	6
2.2	Optimal Methods and Minimax Polynomials	9
2.3	The Chebyshev Method	11
2.4	Notes and References	20
3	Nonlinear Acceleration	22
3.1	Introduction	22
3.2	Nonlinear Acceleration for Quadratic Minimization	24
3.3	Regularized Nonlinear Acceleration Beyond Quadratics	34
3.4	Extensions	42
3.5	Globalization Strategies and Speeding-up Heuristics	44
3.6	Notes and References	45
4	Nesterov Acceleration	47
4.1	Introduction	48
4.2	Gradient Method and Potential Functions	51
4.3	Optimized Gradient Method	55
4.4	Nesterov's Acceleration	65
4.5	Acceleration under Strong Convexity	73

4.6	Recent Variants of Accelerated Methods	84
4.7	Practical Extensions	93
4.8	Continuous-time Interpretations	116
4.9	Notes and References	123
5	Proximal Acceleration and Catalysts	128
5.1	Introduction	128
5.2	Proximal Point Algorithm and Acceleration	129
5.3	Güler and Monteiro-Svaiter Acceleration	134
5.4	Exploiting Strong Convexity	139
5.5	Application: Catalyst Acceleration	145
5.6	Notes and References	154
6	Restart Schemes	157
6.1	Introduction	157
6.2	Hölderian Error Bounds	161
6.3	Optimal Restart Schemes	164
6.4	Robustness and Adaptivity	166
6.5	Extensions	167
6.6	Calculus Rules	170
6.7	Restarting Other First-Order Methods	171
6.8	Application: Compressed Sensing	173
6.9	Notes and References	174
	Appendices	175
A	Useful Inequalities	176
A.1	Smoothness and Strong Convexity in Euclidean spaces	176
A.2	Smoothness for General Norms and Restricted Sets	183
B	Variations on Nesterov Acceleration	184
B.1	Relations between Acceleration Methods	184
B.2	Conjugate Gradient Method	189
B.3	Acceleration Without Monotone Backtracking	194

C On Worst-case Analyses for First-order Methods	203
C.1 Principled Approaches to Worst-case Analyses	203
C.2 Worst-case Analysis as Optimization/Feasibility Problems .	204
C.3 Analysis of Gradient Descent via Linear Matrix Inequalities	208
C.4 Accelerated Gradient Descent via Linear Matrix Inequalities	214
C.5 Notes and References	214
Acknowledgements	217
References	219

Acceleration Methods

Alexandre d'Aspremont¹, Damien Scieur² and Adrien Taylor³

¹*CNRS & Ecole Normale Supérieure, Paris, France; aspremon@ens.fr*

²*Samsung SAIT AI Lab & Mila, Montreal, Canada;*

damien.scieur@gmail.com

³*INRIA & Ecole Normale Supérieure, Paris, France;*

adrien.taylor@inria.fr

ABSTRACT

This monograph covers some recent advances in a range of acceleration techniques frequently used in convex optimization. We first use quadratic optimization problems to introduce two key families of methods, namely momentum and nested optimization schemes. They coincide in the quadratic case to form the *Chebyshev method*.

We discuss momentum methods in detail, starting with the seminal work of Nesterov [1] and structure convergence proofs using a few master templates, such as that for *optimized gradient methods*, which provide the key benefit of showing how momentum methods optimize convergence guarantees. We further cover proximal acceleration, at the heart of the *Catalyst* and *Accelerated Hybrid Proximal Extragradient* frameworks, using similar algorithmic patterns.

Common acceleration techniques rely directly on the knowledge of some of the regularity parameters in the problem at hand. We conclude by discussing *restart* schemes, a set of simple techniques for reaching nearly optimal convergence rates while adapting to unobserved regularity parameters.

Alexandre d'Aspremont, Damien Scieur and Adrien Taylor (2021), “Acceleration Methods”, Foundations and Trends® in Optimization: Vol. 5, No. 1-2, pp 1–245. DOI: 10.1561/24000000036.

©2021 A. d'Aspremont et al.

1

Introduction

Optimization methods are a core component of the modern numerical toolkit. In many cases, iterative algorithms for solving convex optimization problems have reached a level of efficiency and reliability comparable to that of advanced linear algebra routines. This is largely true for medium scale-problems where interior point methods reign supreme, but less so for large-scale problems where the complexity of first-order methods is not as well understood and efficiency remains a concern.

The situation has improved markedly in recent years, driven in particular by the emergence of a number of applications in statistics, machine learning, and signal processing. Building on Nesterov's path-breaking algorithm from the 80's, several accelerated methods and numerical schemes have been developed that both improve the efficiency of optimization algorithms and refine their complexity bounds. Our objective in this monograph is to cover these recent developments using a few master templates.

The methods described in this manuscript can be arranged in roughly two categories. The first, stemming from the work of Nesterov [1], produces variants of the gradient method with accelerated worst-case

convergence rates that are provably optimal under classical regularity assumptions. The second uses outer iteration (a.k.a. nested) schemes to speed up convergence. In this second setting, accelerated schemes run both an inner loop and an outer loop, with the inner iterations being solved by classical optimization methods, and the outer loop containing the acceleration mechanism.

Direct acceleration techniques. Ever since the original algorithm by Nesterov [1], the acceleration phenomenon was regarded as somewhat of a mystery. While accelerated gradient methods can be seen as iteratively building a model for the function and using it to guide gradient computations, the argument is essentially algebraic and is simply an effective exploitation of regularity assumptions. This approach of collecting inequalities induced by regularity assumptions and cleverly chaining them to prove convergence was also used in e.g., [2], to produce an optimal proximal gradient method. There too, however, the proof yielded little evidence as to why the method is actually faster.

Fortunately, we are now better equipped to push the proof mechanisms much further. Recent advances in the programmatic design of optimization algorithms allow us to design and analyze algorithms by following a more principled approach. In particular, the *performance estimation approach*, pioneered by Drori and Teboulle [3], can be used to design optimal methods from scratch, selecting algorithmic parameters to optimize worst-case performance guarantees [3], [4]. Primal dual optimality conditions on the design problem then provide a blueprint for the accelerated algorithm structure and for its convergence proof.

Using this framework, acceleration is no longer a mystery: it is the main objective in the design of the algorithm. We recover the usual “soup of regularity inequalities” that forms the template of classical convergence proofs, but the optimality conditions of the design problem explicitly produce a method that optimizes the convergence guarantee. In this monograph, we cover accelerated first-order methods using this systematic template and describe a number of convergence proofs for classical variants of the accelerated gradient method, such as those of Nesterov (1983, 2003), Beck and Teboulle [2] and Tseng [6] as well as more recent ones [4].

Nested acceleration schemes. The second category of acceleration techniques that we cover in this monograph is composed of outer iteration schemes, in which classical optimization algorithms are used as a black-box in the inner loop and acceleration is produced by an argument in the outer loop. We describe three acceleration results of this type.

The first scheme is based on nonlinear acceleration techniques. Based on arguments dating back to [7]–[9], these techniques use a weighted average of iterates to extrapolate a better candidate solution than the last iterate. We begin by describing the Chebyshev method for solving quadratic problems, which interestingly qualifies both as a gradient method and as an outer iteration scheme. It takes its name from the use of Chebyshev polynomial coefficients to approximately minimize the gradient at the extrapolated solution. The argument can be extended to non-quadratic optimization problems provided the extrapolation procedure is regularized.

The second scheme, due to [10]–[12] relies on a conceptual accelerated proximal point algorithm, and uses classical iterative methods to approximate the proximal point in an inner loop. In particular, this framework produces accelerated gradient methods (in the same sense as Nesterov’s acceleration) when the approximate proximal points are computed using linearly converging gradient-based optimization methods, taking advantage of the fact that the inner problems are always strongly convex.

Finally, we describe restart schemes. These techniques exploit regularity properties called Hölderian error bounds, which extend strong convexity properties near the optimum and hold almost generically, to improve the convergence rates of most first-order methods. The parameters of the Hölderian error bounds are usually unknown, but the restart schemes are robust: that is, they are adaptive to the Hölderian parameters and their empirical performance is excellent on problems with reasonable precision targets.

Content and organization. We present a few convergence acceleration techniques that are particularly relevant in the context of (first-order) convex optimization. Our summary includes our own points of view on

the topic and is focused on techniques that have received substantial attention since the early 2000's, although some of the underlying ideas are much older. We do not pretend to be exhaustive, and we are aware that valuable references might not appear below.

The sections can be read nearly independently. However, we believe the insights of some sections can benefit the understanding of others. In particular, Chebyshev acceleration (Section 2) and nonlinear acceleration (Section 3) are clearly complementary readings. Similarly, Chebyshev acceleration (Section 2) and Nesterov acceleration (Section 4), Nesterov acceleration (Section 4) and proximal acceleration (Section 5), as well as Nesterov acceleration (Section 4) and restart schemes (Section 6) certainly belong together.

Prerequisites and complementary readings. This monograph is not meant to be a general-purpose manuscript on convex optimization, for which we refer the reader to the now classical references [13]–[15]. Other directly related references are provided in the text.

We assume the reader to have a working knowledge of base linear algebra and convex analysis (such as of subdifferentials), as we do not detail the corresponding technical details while building on them. Classical references on the latter include [16]–[18].

2

Chebyshev Acceleration

While “Chebyshev polynomials are everywhere dense in numerical analysis,” we would like to argue here that Chebyshev polynomials also provide one of the most direct and intuitive explanations for acceleration arguments in first-order methods. That is, one can form linear combinations of past gradients for optimizing a worst-case guarantee on the distance to an optimal solution. In quadratic optimization, these linear combinations emerge from a Chebyshev minimization problem, whose solution can also be computed iteratively, thereby yielding an algorithm called the *Chebyshev method* [19]. The Chebyshev method traces its roots to at least Flanders and Shortley [20], who credit Tuckey and Grosch. Its recurrence matches asymptotically the one of the heavy-ball method and is detailed below.

2.1 Introduction

In this section, we demonstrate basic acceleration results on quadratic minimization problems. In such problems, optimal points are the solutions of a linear system, and the basic gradient method can be seen as a simple iterative solver for this linear system. In this context, acceleration methods can be obtained using a classical argument involving Chebyshev polynomials.

Analyzing this simple scenario is useful in two ways. First, recursive formulations of the Chebyshev argument yield a basic algorithmic template for designing accelerated methods and provide first approach to their structures, such as the presence of a momentum term. Second, the arguments are robust to perturbations of the quadratic function f and hence apply in more generic contexts. This property enables acceleration in a wider range of applications, which we cover later in the Section 3 and Section 4.

For now, consider the following unconstrained quadratic minimization problem

$$\text{minimize } \left\{ f(x) \triangleq \frac{1}{2} \langle x; \mathbf{H}x \rangle - \langle b; x \rangle \right\} \quad (2.1)$$

in the variable $x \in \mathbb{R}^d$, where $\mathbf{H} \in \mathbf{S}_d$ (the set of symmetric matrices of size $d \times d$) is the Hessian of f . We further assume that f is both smooth and strongly convex, i.e., that there exist some $L > \mu > 0$ such that $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$. The reasoning of this section readily extends to the case where μ is the smallest *nonzero* eigenvalue of $\mathbf{H}$. We start by analyzing the convergence of the fixed step gradient method (Algorithm 1) for solving (2.1).

Algorithm 1 Gradient method

Input: A differentiable convex function f , initial point x_0 , step size $\gamma > 0$, budget N .

- 1: **for** $k = 1, \dots, N$ **do**
- 2: $x_k = x_{k-1} - \gamma \nabla f(x_{k-1})$
- 3: **end for**

Output: Approximate solution x_N .

For problem (2.1), the iteration reads

$$x_{k+1} = (\mathbf{I} - \gamma \mathbf{H})x_k + \gamma b,$$

and calling $x_\star$ the optimum of problem (2.1) (satisfying $\mathbf{H}x_\star = b$) yields

$$x_{k+1} - x_\star = (\mathbf{I} - \gamma \mathbf{H})(x_k - x_\star). \quad (2.2)$$

This means the iterates of gradient descent $x_k - x_*$ can be computed from $x_0 - x_*$ via $x_k - x_* = P_k^{\text{Grad}}(\mathbf{H})(x_0 - x_*)$ using the matrix polynomial

$$P_k^{\text{Grad}}(\mathbf{H}) = (\mathbf{I} - \gamma \mathbf{H})^k. \quad (2.3)$$

Suppose we set the step size γ to ensure

$$\|\mathbf{I} - \gamma \mathbf{H}\|_2 < 1,$$

where $\|\cdot\|_2$ stands for the operator ℓ_2 norm. Then, (2.2) controls the convergence with

$$\|x_k - x_*\|_2 \leq \|\mathbf{I} - \gamma \mathbf{H}\|_2^k \|x_0 - x_*\|_2, \quad \text{for } k \geq 0. \quad (2.4)$$

Because the matrix $\mathbf{H}$ is symmetric and hence diagonalizable in an orthogonal basis, given $\gamma > 0$, we obtain

$$\begin{aligned} \|\mathbf{I} - \gamma \mathbf{H}\|_2 &\leq \max_{\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}} \|\mathbf{I} - \gamma \mathbf{H}\|_2 \\ &\leq \max_{\mu \leq \lambda \leq L} |1 - \gamma \lambda| \\ &\leq \max_{\mu \leq \lambda \leq L} \max \{\gamma \lambda - 1; 1 - \gamma \lambda\} \\ &\leq \max \{\gamma L - 1; 1 - \gamma \mu\}. \end{aligned}$$

To get the best possible worst-case convergence rate, we now minimize this quantity in γ by solving

$$\min_{\gamma} \max \{\gamma L - 1, 1 - \gamma \mu\} = \frac{L - \mu}{L + \mu}. \quad (2.5)$$

The optimal step size is obtained when both terms in the max are equal, reaching:

$$\gamma = \frac{2}{L + \mu}. \quad (2.6)$$

Denoting by $\kappa \triangleq \frac{L}{\mu} \geq 1$ the *condition number* of the function f , the bound in (2.4) finally becomes

$$\|x_k - x_*\|_2 \leq \left(\frac{\kappa - 1}{\kappa + 1} \right)^k \|x_0 - x_*\|_2, \quad \text{for } k \geq 0, \quad (2.7)$$

which is a worst-case guarantee for the gradient method when minimizing smooth strongly convex quadratic functions.

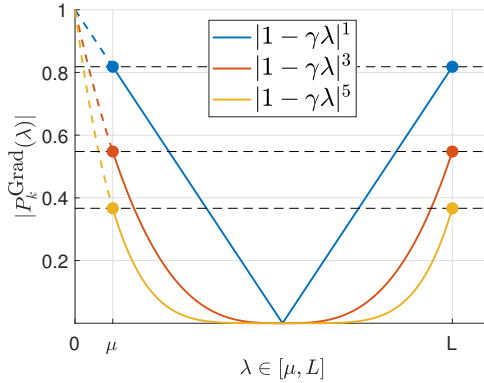


Figure 2.1: We plot $|P_k^{\text{Grad}}(\lambda)|$ (for the optimal γ in (2.6)) for $k \in \{1, 3, 5\}$, $\mu = 1$, $L = 10$. Note that the polynomials satisfy $P_k^{\text{Grad}}(0) = 1$. The rate is equal to the largest value of $|P_k^{\text{Grad}}(\lambda)|$ on the interval, which is achieved at the boundaries (where λ is either equal to μ or to L).

2.2 Optimal Methods and Minimax Polynomials

In Equation (2.4) above, we saw that the worst-case convergence rate of the gradient method on quadratic functions can be controlled by the spectral norm of a matrix polynomial. Figure 2.1 plots the polynomial P_k^{Grad} for several degrees k . We can extend this reasoning further to produce methods with accelerated worst-case convergence guarantees.

2.2.1 First-Order Methods and Matrix Polynomials

The bounds derived above for gradient descent can be extended to a broader class of first-order methods for quadratic optimization. We consider first-order algorithms in which each iterate belongs to the span of previous gradients, i.e.

$$x_{k+1} \in x_0 + \text{span} \{ \nabla f(x_0), \nabla f(x_1), \dots, \nabla f(x_k) \}, \quad (2.8)$$

and show that the iterates can be written using matrix polynomials as in (2.3) above.

Proposition 2.1. Let $x_0 \in \mathbb{R}^d$ and f be a quadratic function defined as in (2.1) with $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$ for some $L > \mu > 0$. The sequence $\{x_k\}_{k=0,1,\dots}$ satisfies

$$x_{k+1} \in x_0 + \text{span} \{ \nabla f(x_0), \nabla f(x_1), \dots, \nabla f(x_k) \}, \quad (2.9)$$

for all $k = 0, 1, \dots$, if and only if the errors $\{x_k - x_\star\}_{k=0,1,\dots}$ can be written as

$$x_k - x_\star = P_k(\mathbf{H})(x_0 - x_\star), \quad (2.10)$$

for all $k = 0, 1, \dots$, for some sequence of polynomials $\{P_k\}_{k=0,1,\dots}$ with P_k of degree at most k and $P_k(0) = 1$.

Proof. Since $\nabla f(x)$ is the gradient of a quadratic function, it reads

$$\nabla f(x) = \mathbf{H}x - b = \mathbf{H}(x - x_\star)$$

for any $x_\star$ satisfying $\mathbf{H}x_\star = b$, where $\mathbf{H}$ is symmetric. We have

$$\begin{aligned} x_0 - x_\star &= 1 \cdot (x_0 - x_\star) \\ &= P_0(\mathbf{H})(x_0 - x_\star). \end{aligned}$$

We now show recursively that $x_k - x_\star = P_k(\mathbf{H})(x_0 - x_\star)$, where P_k is a residual polynomial of degree at most k . Our assumption about the iterates (2.9) implies that, for some sequence of coefficients $\{\alpha_i^{(k+1)}\}_{i=0,\dots,k}$,

$$x_{k+1} - x_\star = x_0 - x_\star + \sum_{i=0}^k \alpha_i^{(k+1)} \nabla f(x_i).$$

Assuming recursively that (2.10) holds for all indices $i \leq k$,

$$\begin{aligned} x_{k+1} - x_\star &= x_0 - x_\star + \sum_{i=0}^k \alpha_i^{(k+1)} \mathbf{H} P_i(\mathbf{H})(x_0 - x_\star) \\ &= \left(\mathbf{I} + \mathbf{H} \sum_{i=0}^k \alpha_i^{(k+1)} P_i(\mathbf{H}) \right) (x_0 - x_\star). \end{aligned}$$

Then, by writing $P_{k+1}(x) = 1 + x \sum_{i=0}^k \alpha_i^{(k+1)} P_i(x)$, we have

$$x_{k+1} - x_\star = P_{k+1}(\mathbf{H})(x_0 - x_\star)$$

with $P_{k+1}(0) = 1$ and $\deg(P_{k+1}) \leq k + 1$. Since the proof is a sequence of equalities, the equivalence readily follows. ■

Given a class $\mathcal{M}$ of problem matrices $\mathbf{H}$, Proposition 2.1 provides a way to design algorithms. Indeed, we can extract a first-order method from a sequence of polynomials $\{P_k\}_{k=0,\dots,N}$. We can therefore use tools from approximation theory to find optimal polynomials and extract corresponding methods from them. Given a matrix class $\mathcal{M}$, this involves minimizing the worst-case convergence bound over $\mathbf{H} \in \mathcal{M}$ by solving

$$P_k^* = \operatorname{argmin}_{\substack{P \in \mathcal{P}_k, \\ P(0)=1}} \max_{\mathbf{H} \in \mathcal{M}} \|P(\mathbf{H})\|_2 \quad (2.11)$$

where P_k is the set of polynomials of degree at most k . The polynomial P_k^* is an optimal polynomial for $\mathcal{M}$ and yields a (worst-case) optimal algorithm for the class $\mathcal{M}$. In terms of the notation used in the proof of Proposition 2.1, we are by construction looking for coefficients $\{\alpha_i^{(j)}\}_i$ depending only on the problem class $\mathcal{M}$, but *not* on a specific instance of $\mathbf{H}$.

2.3 The Chebyshev Method

In the case where $\mathcal{M}$ is the set of positive definite matrices with a bounded spectrum, namely

$$\mathcal{M} = \{\mathbf{H} \in \mathbf{S}_d : 0 \prec \mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}\},$$

the optimal polynomial can be found by solving

$$P_k^* = \operatorname{argmin}_{\substack{P \in \mathcal{P}_k, \\ P(0)=1}} \max_{\lambda \in [\mu, L]} |P(\lambda)| \quad (2.12)$$

Polynomials that solve (2.12) are derived from *Chebyshev polynomials of the first kind* in approximation theory and can be formed explicitly to produce an optimal algorithm called the *Chebyshev method*. This section describes this method and provides its corresponding worst-case convergence guarantees.

2.3.1 Shifted Chebyshev Polynomials

We now explicitly introduce the Chebyshev polynomials. A more complete treatment of these polynomials is available in, e.g., Mason and

Handscorn [21]. Chebyshev polynomials of the first kind are defined recursively as follows

$$\begin{aligned}\mathcal{T}_0(x) &= 1, \\ \mathcal{T}_1(x) &= x, \\ \mathcal{T}_k(x) &= 2x\mathcal{T}_{k-1}(x) - \mathcal{T}_{k-2}(x), \quad \text{for } k \geq 2.\end{aligned}\tag{2.13}$$

There exists a compact explicit solution for Chebyshev polynomials that involves trigonometric functions:

$$\mathcal{T}_k(x) = \begin{cases} \cos(k \arccos(x)) & x \in [-1, 1], \\ \cosh(k \operatorname{acosh}(x)) & x > 1, \\ (-1)^k \cosh(k \operatorname{acosh}(-x)) & x < -1. \end{cases}\tag{2.14}$$

It is possible to show that Chebyshev polynomials satisfy the minimax property

$$\frac{1}{2^{k-1}} \mathcal{T}_k = \operatorname{argmin}_{\substack{\deg(P) \leq k \\ P \text{ is monic}}} \max_{x \in [-1, 1]} |P(x)|,$$

where a monic polynomial is a polynomial whose coefficient associated with the highest power is equal to one. From this minimax definition, that defines the minimal polynomial over $[-1, 1]$, we can transform it by *shifting* it to the interval $[\mu, L]$, then *rescaling* it to obtain a polynomial such that $P(0) = 1$. More precisely, using a simple linear mapping from $[\mu, L]$ to $[-1, 1]$,

$$x \rightarrow t^{[\mu, L]}(x) = \frac{2x - (L + \mu)}{L - \mu},$$

we obtain *shifted Chebyshev polynomials*:

$$C_k^{[\mu, L]}(x) = \frac{\mathcal{T}_k(t^{[\mu, L]}(x))}{\mathcal{T}_k(t^{[\mu, L]}(0))}.\tag{2.15}$$

where we have enforced the normalization constraint $C_k^{[\mu, L]}(0) = 1$. Under these transformations, the shifted Chebyshev polynomials keep some minimax property, and can be shown to be solutions to (2.12).

More formally, Golub and Varga [22] characterize the ℓ_∞ optimality on the interval $[\mu, L]$ using an *equi-oscillation* argument (see, e.g.,

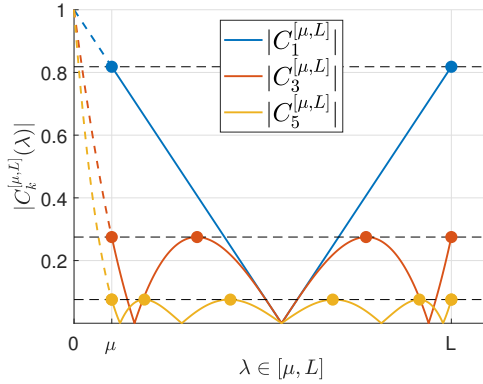


Figure 2.2: We plot the absolute value of $C_1^{[\mu,L]}(x)$, $C_3^{[\mu,L]}(x)$ and $C_5^{[\mu,L]}(x)$ for $\lambda \in [\mu, L]$, where $\mu = 1$ and $L = 10$. Note that the polynomials satisfy $C_k^{[\mu,L]}(0) = 1$. The maximum value of the image of $[\mu, L]$ by $C_k^{[\mu,L]}$ decreases rapidly as k grows, implying an accelerated rate of convergence.

[23]); i.e., they show that the solution of (2.12) is $P_k^* = C_k^{[\mu,L]}$. The equioscillation property of the shifted Chebyshev polynomial is clearly visible on Figure 2.2, where the polynomial hits its maximum value $k + 1$ times on the interval $[\mu, L]$.

2.3.2 Chebyshev Algorithm

The following recursion follows from (2.13) together with (2.15) and a few simplifications:

$$\begin{aligned}
 C_0^{[\mu,L]}(x) &= 1, \\
 C_1^{[\mu,L]}(x) &= 1 - \frac{2}{L + \mu}x, \\
 C_k^{[\mu,L]}(x) &= \frac{2\delta_k}{L - \mu}(L + \mu - 2x)C_{k-1}^{[\mu,L]}(x) \\
 &\quad + \left(1 - \frac{2\delta_k(L + \mu)}{L - \mu}\right)C_{k-2}^{[\mu,L]}(x), \quad \text{for } k \geq 2,
 \end{aligned} \tag{2.16}$$

where $\delta_1 = \frac{L - \mu}{L + \mu}$ and

$$\delta_k = -\frac{\mathcal{T}_{k-1}(t^{[\mu,L]}(0))}{\mathcal{T}_k(t^{[\mu,L]}(0))} = \frac{1}{2\frac{L + \mu}{L - \mu} - \delta_{k-1}}, \quad \text{for } k \geq 2.$$

The sequence δ_k ensures $C_k^{[\mu,L]}(0) = 1$. We present this recursion for simplicity, but one should note that it might present some numerical stability issues in practice. There exist numerically more stable first-order methods based on Chebyshev polynomials, see, e.g., [24, Algorithm 1]. We plot $C_1^{[\mu,L]}(x)$, $C_3^{[\mu,L]}(x)$ and $C_5^{[\mu,L]}(x)$ in Figure 2.2 for illustration.

Computational details for the shifted Chebyshev. Let us quickly detail how to arrive to (2.16) using (2.13) and (2.15). We first expand $\mathcal{T}_k(t^{[\mu,L]}(x))$ using (2.13),

$$\begin{aligned}\mathcal{T}_k(t^{[\mu,L]}(x)) &= 2t^{[\mu,L]}(x)\mathcal{T}_{k-1}(t^{[\mu,L]}(x)) - \mathcal{T}_{k-2}(t^{[\mu,L]}(x)), \\ &= \frac{2(2x - L - \mu)}{L - \mu}\mathcal{T}_{k-1}(t^{[\mu,L]}(x)) - \mathcal{T}_{k-2}(t^{[\mu,L]}(x)).\end{aligned}$$

Since $C_k^{[\mu,L]}(x) = \frac{\mathcal{T}_k(t^{[\mu,L]}(x))}{\mathcal{T}_k(t^{[\mu,L]}(0))}$, we substitute $\mathcal{T}_k(t^{[\mu,L]}(x))$ by $\mathcal{T}_k(t^{[\mu,L]}(0)) \cdot C_k^{[\mu,L]}(x)$.

$C_k^{[\mu,L]}(x)$ in the equation above and obtain

$$\begin{aligned}\mathcal{T}_k(t^{[\mu,L]}(0)) \cdot C_k^{[\mu,L]}(x) &= \frac{2(2x - L - \mu)}{L - \mu}\mathcal{T}_{k-1}(t^{[\mu,L]}(0)) \cdot C_{k-1}^{[\mu,L]}(x) \\ &\quad - \mathcal{T}_{k-2}(t^{[\mu,L]}(0)) \cdot C_{k-2}^{[\mu,L]}(x), \\ C_k^{[\mu,L]}(x) &= \frac{2(2x - L - \mu)}{L - \mu} \frac{\mathcal{T}_{k-1}(t^{[\mu,L]}(0))}{\mathcal{T}_k(t^{[\mu,L]}(0))} \cdot C_{k-1}^{[\mu,L]}(x) \\ &\quad - \frac{\mathcal{T}_{k-2}(t^{[\mu,L]}(0))}{\mathcal{T}_k(t^{[\mu,L]}(0))} \cdot C_{k-2}^{[\mu,L]}(x), \\ C_k^{[\mu,L]}(x) &= -\frac{2(2x - L - \mu)}{L - \mu}\delta_k C_{k-1}^{[\mu,L]}(x) \\ &\quad - \delta_{k-1}\delta_k C_{k-2}^{[\mu,L]}(x).\end{aligned}$$

For obtaining a simple recursion on δ_k , note that $C_k^{[\mu,L]}(0) = 1$ for all $k \geq 0$ by construction. It follows that

$$C_k^{[\mu,L]}(0) = \frac{2(L + \mu)}{L - \mu}\delta_k \underbrace{C_{k-1}^{[\mu,L]}(0)}_{=1} - \delta_{k-1}\delta_k \underbrace{C_{k-2}^{[\mu,L]}(0)}_{=1} = 1,$$

and thereby

$$\delta_{k-1}\delta_k = 1 - 2\delta_k \frac{L + \mu}{L - \mu} \quad \text{and} \quad \delta_k = \frac{1}{2\frac{L + \mu}{L - \mu} - \delta_{k-1}}.$$

2.3.3 Chebyshev and Polyak's Heavy-Ball Methods

We now present the resulting algorithm, called *Chebyshev semi-iterative method* [25]. We define iterates using $C_k^{[\mu, L]}(x)$ as follows,

$$x_k - x_\star = C_k^{[\mu, L]}(\mathbf{H})(x_0 - x_\star).$$

The recursion in (2.16) then yields

$$\begin{aligned} x_k - x_\star &= \frac{2\delta_k}{L - \mu} ((L + \mu)\mathbf{I} - 2\mathbf{H})(x_{k-1} - x_\star) \\ &\quad + \left(1 - 2\delta_k \frac{L + \mu}{L - \mu}\right) (x_{k-2} - x_\star). \end{aligned}$$

Since the gradient of the function reads

$$\nabla f(x_k) = \mathbf{H}(x_k - x_\star),$$

we can simplify away $x_\star$ to get the following recursion:

$$x_k = \frac{2\delta_k}{L - \mu} ((L + \mu)x_{k-1} - 2\nabla f(x_{k-1})) + \left(1 - 2\delta_k \frac{L + \mu}{L - \mu}\right) x_{k-2},$$

which describes iterates of the Chebyshev method. We summarize it as Algorithm 2. By construction, the Chebyshev method is a worst-case optimal first-order method for minimizing quadratics whose spectrum lies in $[\mu, L]$. Surprisingly, its iteration structure is simple and somewhat intuitive: it involves a gradient step with variable step size $4\delta_k/(L - \mu)$, combined with a variable momentum term.

Algorithm 2 Chebyshev's method

Input: An L -smooth μ -strongly convex quadratic f , initial point x_0 and budget N .

- 1: Set $\delta_1 = \frac{L-\mu}{L+\mu}$, $x_1 = x_0 - \frac{2}{L+\mu} \nabla f(x_0)$.
- 2: **for** $k = 2, \dots, N$ **do**
- 3: Set $\delta_k = \frac{1}{2 \frac{L+\mu}{L-\mu} - \delta_{k-1}}$,
- 4: $x_k = x_{k-1} - \frac{4\delta_k}{L-\mu} \nabla f(x_{k-1}) + \left(1 - 2\delta_k \frac{L+\mu}{L-\mu}\right) (x_{k-2} - x_{k-1})$.
- 5: **end for**

Output: Approximate solution x_N .

Perhaps more surprisingly, the Chebyshev method has a *stationary regime* that is even simpler. Indeed, when $k \rightarrow \infty$, the coefficients of the recursion from Algorithm 2 converge to the ones of *Polyak's heavy-ball method*,

$$x_k = x_{k-1} - \frac{4}{(\sqrt{L} + \sqrt{\mu})^2} \nabla f(x_{k-1}) + \frac{(\sqrt{L} - \sqrt{\mu})^2}{(\sqrt{L} + \sqrt{\mu})^2} (x_{k-1} - x_{k-2}).$$

To see this, it suffices to compute the limit of δ_k , written as δ_∞ , by solving

$$\delta_\infty = \frac{1}{2^{\frac{L+\mu}{L-\mu}} - \delta_\infty},$$

reaching

$$\delta_\infty = \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}. \quad (2.17)$$

We obtain Polyak's heavy-ball method by replacing δ_k with δ_∞ in Algorithm 2.

2.3.4 Worst-case Convergence Bounds

The *shifted* Chebyshev polynomials are solutions to (2.12). Therefore, using the same trick as for gradient descent, we can obtain the following worst-case bound for the Chebyshev method

$$\|x_k - x_\star\|_2 \leq \|C_k^{[\mu, L]}(\mathbf{H})(x_0 - x_\star)\|_2 \leq \|x_0 - x_\star\|_2 \max_{x \in [\mu, L]} |C_k^{[\mu, L]}(x)|. \quad (2.18)$$

The maximum value is determined by evaluating the polynomial at one of the extremities of the interval [21, Chapter 2] (see also Figure 2.2), i.e.,

$$\max_{x \in [\mu, L]} |C_k^{[\mu, L]}(x)| = C_k^{[\mu, L]}(L).$$

Using (2.15) and (2.14) successively,

$$|C_k^{[\mu, L]}(L)| = \frac{1}{|\mathcal{T}_k(t^{[\mu, L]}(0))|} = \frac{1}{\cosh\left(k \operatorname{acosh}\left(\frac{L+\mu}{L-\mu}\right)\right)}.$$

We obtain the worst-case convergence guarantee of the Chebyshev method, as stated in the following theorem.

Theorem 2.1. Let $x_0 \in \mathbb{R}^d$ and f be a quadratic function defined as in (2.1) with $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$ for some $L > \mu > 0$. For any $N \in \mathbb{N}$, the iterates of the Chebyshev method (Algorithm 2) satisfy

$$\|x_N - x_\star\|_2 \leq \frac{2}{\xi^N + \xi^{-N}} \|x_0 - x_\star\|_2 \quad \text{where } \xi = \frac{\sqrt{\frac{L}{\mu}} + 1}{\sqrt{\frac{L}{\mu}} - 1}. \quad (2.19)$$

Proof. We bound (2.18) as follows,

$$\begin{aligned} \|x_N - x_\star\|_2 &\leq \|x_0 - x_\star\|_2 \max_{x \in [\mu, L]} |C_N^{[\mu, L]}(x)| \\ &= \|x_0 - x_\star\|_2 \frac{1}{\cosh\left(N \operatorname{acosh}\left(\frac{L+\mu}{L-\mu}\right)\right)}. \end{aligned}$$

First, we evaluate the acosh term,

$$\begin{aligned} \operatorname{acosh}\left(\frac{L+\mu}{L-\mu}\right) &= \ln\left(\frac{L+\mu}{L-\mu} + \sqrt{\left(\frac{L+\mu}{L-\mu}\right)^2 - 1}\right), \\ &= \ln(\xi), \quad \text{where } \xi = \frac{\sqrt{\frac{L}{\mu}} + 1}{\sqrt{\frac{L}{\mu}} - 1}. \end{aligned}$$

After plugging this result into the $\cosh$, we get

$$\frac{1}{\cosh(N \ln(\xi))} = \frac{2}{e^{N \ln(\xi)} + e^{-N \ln(\xi)}} = \frac{2}{\xi^N + \xi^{-N}},$$

thereby reaching the desired result. ■

It may be difficult to compare the convergence rate of the Chebyshev method with that of gradient descent, due to its more complex expression. However, by neglecting the denominator term ξ^{-N} , we obtain the following upper bound:

$$\|x_N - x_\star\|_2 \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^N \|x_0 - x_\star\|_2.$$

Note that the convergence rate of Polyak's heavy-ball method matches (up to a multiplicative factor) that of Chebyshev's method

asymptotically, which is better than that of gradient descent in (2.7), which reads

$$\|x_N - x_\star\|_2 \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^N \|x_0 - x_\star\|_2.$$

We summarize this result in the following corollary, which compares the number of iterations required to reach a target accuracy ϵ .

Corollary 2.2. Let $x_0 \in \mathbb{R}^d$ and f be a quadratic function defined as in (2.1) with $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$ for some $L > \mu > 0$. After respectively

- $N \geq \frac{L}{2\mu} \log\left(\frac{\|x_0 - x_\star\|_2}{\epsilon}\right)$ iterations of gradient descent (Algorithm 1 with (2.6)), or
- $N \geq \sqrt{\frac{L}{2\mu}} \log\left(\frac{\|x_0 - x_\star\|_2}{\epsilon}\right)$ iterations of Chebyshev's method (Algorithm 2),

we have that

$$\|x_N - x_\star\|_2 \leq \epsilon.$$

Proof. For gradient descent, a sufficient condition on the number of iterations required to reach an accuracy of ϵ reads

$$\|x_N - x_\star\|_2 \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^N \|x_0 - x_\star\|_2 \leq \epsilon.$$

Taking the log on both sides, we get

$$N \geq \frac{\log\left(\frac{\|x_0 - x_\star\|_2}{\epsilon}\right)}{\log\left(\frac{\kappa+1}{\kappa-1}\right)}.$$

Using the bound $\log\left(\frac{\frac{1}{x}+1}{\frac{1}{x}-1}\right) > 2x$, the above condition can be simplified to the following stronger condition on N

$$N \geq \frac{\kappa}{2} \log\left(\frac{\|x_0 - x_\star\|_2}{\epsilon}\right).$$

This gives the desired result for gradient descent. With the same approach, we also get the result for the Chebyshev algorithm. ■

This corollary shows that the Chebyshev method can be $\sqrt{\kappa}$ faster than gradient descent. This translate to a speedup factor of 100 in problems with a (reasonable) condition number of 10^4 , which is very significant.

Worst-case optimality of Chebyshev’s method. When the dimension d of the ambient space is sufficiently large, and without further assumptions on the spectrum of $\mathbf{H}$, the worst-case guarantee on Chebyshev’s method is essentially unimprovable. Informally, given a budget N , a problem class $\mathcal{M}$ and some $R > 0$, the best worst-case guarantee on the distance to optimality $\|x_N - x_\star\|_2$ that can be achieved by a first-order method is given by

$$\max_{\substack{\mathbf{H} \in \mathcal{M}, x_\star, x_0 \in \mathbb{R}^d \\ \|x_0 - x_\star\|_2 \leq R}} \min_{\substack{P \in \mathcal{P}_N, \\ P(0)=1}} \|P(\mathbf{H})(x_0 - x_\star)\|_2, \quad (2.20)$$

which corresponds to the worst-case performance of the best performing method on any problem of the class. More precisely, for any first-order method satisfying $x_k \in x_0 + \text{span}\{\nabla f(x_0), \nabla f(x_1), \dots, \nabla f(x_{k-1})\}$ for all $k \geq 1$ (the “span assumption”) applied on the quadratic problem (2.1), it holds that:

$$\begin{aligned} & \text{span}\{\nabla f(x_0), \dots, \nabla f(x_{k-1})\} \\ & \subseteq \text{span}\{H(x_0 - x_\star), \dots, H^k(x_0 - x_\star)\}, \end{aligned}$$

and therefore, the conjugate gradient-like method

$$\begin{aligned} x_k &= \underset{x}{\operatorname{argmin}} \|x - x_\star\|_2 \\ \text{s.t. } x &\in x_0 + \text{span}\{H(x_0 - x_\star), \dots, H^k(x_0 - x_\star)\}, \end{aligned} \quad (2.21)$$

is *instance-optimal*: it achieves the best worst-case performance on any problem instance. It follows that the worst-case performance of any first-order method satisfying the span assumption can only be worse than that of (2.21). The worst-case performance of (2.21) being given by (2.20), we have that for any initialization $x_0 \in \mathbb{R}^d$ any first-order method satisfying the span assumption in Equation (2.8), there exists at least one problem on which

$$\|x_N - x_\star\|_2 \geq \max_{\substack{\mathbf{H} \in \mathcal{M}, x_\star \in \mathbb{R}^d \\ \|x_0 - x_\star\|_2 \leq R}} \min_{\substack{P \in \mathcal{P}_N, \\ P(0)=1}} \|P(\mathbf{H})(x_0 - x_\star)\|_2, \quad (2.22)$$

where $x_N \in \mathbb{R}^d$ is the output of the first-order method under consideration, and $x_\star$ is the optimal point of the problem.

In other words, the max term in (2.22) searches for the “most difficult” quadratic function, while the min term represents the *best* first-order method for a specific quadratic function. Of course, the method (2.21) is much more powerful than the Chebyshev method, since it is optimal *for any specific function*. However, it is possible to show that despite being more powerful, this optimal algorithm has the same worst-case performance as that of Chebyshev’s method when the dimension d of the ambient space is large enough. The lower bound result is summarized by the next theorem.

Theorem 2.3. [26, Proposition 12.3.2] Let $N, d \in \mathbb{N}$ such that $d \geq N+1$, and let $x_0 \in \mathbb{R}^d$. There exists $\mathbf{H} \in \mathbf{S}_d : \mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$ and $x_\star \in \mathbb{R}^d$ such that any sequence $\{x_k\}_{k=0,\dots,N}$ generated by any first-order method satisfying (2.8), and initiated at x_0 for minimizing the quadratic function f in the form (2.1) satisfies

$$\|x_N - x_\star\|_2 \geq \|x_0 - x_\star\|_2 \min_{\substack{P \in \mathcal{P}_N, \\ P(0)=1}} \max_{\lambda \in [\mu, L]} |P(\lambda)| = \frac{2}{\xi^N + \xi^{-N}} \|x_0 - x_\star\|_2,$$

$$\text{with } \xi = \frac{\sqrt{\frac{L}{\mu}+1}}{\sqrt{\frac{L}{\mu}-1}}.$$

More details on this topic can be found in [26, Section 12.3].

2.4 Notes and References

The Chebyshev method presented in this section is *worst-case* optimal for the class of quadratic functions with Hessians $\mathbf{H}$ satisfying $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$. More detailed discussions and developments on the topic of Chebyshev polynomials, for quadratic minimization, are provided in [5], [27], [28], as well as in the lecture notes [26, Chapter 10]. Those references include the treatment of the case where the smallest eigenvalue is $\mu = 0$. Finally, one should note that the optimal convergence bounds achieved by the Chebyshev method requires knowledge of the problem class parameters, μ and L , which might or might not be an issue, depending on the problem at hand.

Probably the most celebrated method for unconstrained quadratic optimization problems is the conjugate gradient (CG) method. Its origin is usually attributed to Stiefel [29] and Straeter [30]. As for the setup of this section, it turns out that CG methods are *instance-optimal*, in the sense that they are the best performing first-order methods on every particular problem instance in the range of unconstrained quadratic minimization problems (in particular, the CG variant presented in (2.21) achieves the lower bound from Theorem 2.3). The classical CG produces iterates $\{x_k\}_{k \geq 0}$ such that

$$\begin{aligned} x_{k+1} &\in \operatorname{argmin}_x f(x) \\ \text{s.t. } x &\in x_0 + \operatorname{span}\{H(x_0 - x_\star), \dots, H^k(x_0 - x_\star)\}, \end{aligned}$$

which admits efficient formulation; see, e.g., [15]. Another variant of CG is often referred to as MINRES, which produces iterates $\{x_k\}_{k \geq 0}$ in the form

$$\begin{aligned} x_{k+1} &\in \operatorname{argmin}_x \|\nabla f(x)\|_2 \\ \text{s.t. } x &\in x_0 + \operatorname{span}\{H(x_0 - x_\star), \dots, H^k(x_0 - x_\star)\}. \end{aligned}$$

Its generalization GMRES [31] is popular for solving linear systems of the form $\mathbf{H}x = b$ when $\mathbf{H}$ is not required to be either symmetric or invertible.

Yet another alternative for dealing with quadratic minimization is to resort on *Anderson*-type acceleration schemes. As for conjugate gradient methods, those schemes do not readily extend beyond quadratic minimization with the same nice theoretical guarantees. This is the topic of the next section.

Beyond quadratic optimization, properties of Chebyshev polynomials is the focus of [21]. The use of Chebyshev polynomials in the context of solving linear systems is covered at length in [32]. In particular, the theory of [32] can be instantiated for the convex quadratic minimization in average-case analyses, where Chebyshev polynomials (along with their heavy-ball limits) also naturally appear [33]–[35].

3

Nonlinear Acceleration

In this section, we see that the main argument used in the Chebyshev method can be adapted beyond quadratic problems. The extension that we present here, called nonlinear acceleration, follows a pattern that is known in numerical analysis as *vector extrapolation methods*: it seeks to accelerate the convergence of sequences by extrapolation using nonlinear averages. Different such strategies are known under various names, starting with Aitken’s Δ^2 [7], Wynn’s epsilon algorithm [8], and Anderson acceleration [36]; a survey of these techniques can be found in [37]. The vector extrapolation techniques, generic by nature, can be applied to optimization, as explained in what follows.

3.1 Introduction

This section focuses on the convex minimization problem:

$$\text{minimize } f(x) \tag{3.1}$$

in the variable $x \in \mathbb{R}^d$. We assume f to be twice continuously differentiable in a neighborhood of its minimizer $x_\star$, and denote by $f_\star = f(x_\star)$ the minimum of f .

We aim at adapting some of the ideas behind Chebyshev's acceleration (see Section 2) to a broader class of convex minimization problems beyond quadratic minimization. These adaptations stems from a local quadratic approximation of the objective:

$$f(x) = f_\star + \frac{1}{2} \langle x - x_\star; \mathbf{H}(x - x_\star) \rangle + o(\|x - x_\star\|_2^2), \quad (3.2)$$

where $\mathbf{H} = \nabla^2 f(x_\star) \in \mathbf{S}_d$ (the set of symmetric $d \times d$ matrices) is the Hessian of f at $x_\star$, which we assume to satisfy $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$ for some $0 < \mu < L$. Of course, neglecting the second-order term in (3.2) allows recovering a quadratic minimization problem for which one could apply Chebyshev's method as is.

We recall that the Chebyshev method is the first-order method associated with the *best worst-case polynomial*. In short, the k th iteration of Chebyshev's method consists in combining previous gradients for minimizing a worst-case convergence bound over all μ -strongly convex L -smooth quadratic problems in $\mathbb{R}^d$ (in the form (2.1) and with $d \geq k+1$):

$$\begin{aligned} \alpha_k = \operatorname{argmin}_{\{\alpha^{(i)}\}_i} \quad & \max_{\substack{\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I} \\ x_0, \dots, x_k, x_\star, b \in \mathbb{R}^{k+1}}} \frac{\|x_k - x_\star\|_2^2}{\|x_0 - x_\star\|_2^2} \\ \text{s.t. } & x_k = x_0 - \sum_{i=0}^{k-1} \alpha^{(i)} \nabla f(x_i), \\ & \mathbf{H}x_\star = b, \\ & x_k = x_0 - \sum_{i=0}^{k-1} \alpha_k^{(i)} \nabla f(x_i), \end{aligned} \quad (3.3)$$

so that α_k does not depend on the particular problem instance $(\mathbf{H}, x_\star)$ and on the initialization x_0 , but only on the problem class described by μ and L (we note that in Section 2, (3.3) was expressed in terms of optimizing a polynomial (2.12)). A natural alternative to Chebyshev's method consists in choosing those weights *adaptively*. That is, depending on the particular instance of the problem at hand. For doing so, we have to choose another way to measure performance (because minimizing $\|x_k - x_\star\|_2$ would require knowledge of $x_\star$); one such possibility is to rely on function values or gradient norms. One could then rely on conjugate gradient-type methods which are very attractive for unconstrained

quadratic minimization (see, e.g., discussions in Section 2.4). In this section, we consider the case where a first-order optimization method provided us with a sequence of pairs $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$ satisfying (for $i = 1, \dots, k$)

$$x_i \in x_0 + \text{span} \{\nabla f(x_0), \nabla f(x_1), \dots, \nabla f(x_{i-1})\}, \quad (3.4)$$

and we study methods producing approximations of x_* as linear combinations $\sum_{i=0}^k c_i x_i$ of the previous iterates $\{x_i\}_{i=0,\dots,k}$. For choosing the corresponding weights, we minimize the norm of the gradient at the approximated point. In unconstrained convex quadratic minimization problems, this approach is closely related to the so-called MINRES [38] and GMRES [31] methods (conjugate gradient-type methods minimizing gradient norms; see discussions by Walker and Ni [39] and Section 2.4). That is, when f is quadratic, we choose the weights $\{c_i\}_{i=0,\dots,k}$ by solving

$$c_\star = \underset{c}{\operatorname{argmin}} \left\{ \left\| \nabla f \left(\sum_{i=0}^k c_i x_i \right) \right\|_2^2 : \mathbf{1}^T c = 1 \right\}. \quad (3.5)$$

Whereas the new approximation is a linear combination of previous iterates $\{x_i\}_{i=0,\dots,k}$, the coefficients $\{c_i\}_{i=0,\dots,k}$ depend *nonlinearly* on both ∇f and on $\{x_i\}_{i=0,\dots,k}$. This technique is known under a few different names including those of *Anderson acceleration* and *minimal polynomial extrapolation* (see discussions and references in Section 3.6 for more details). In this section, we refer to all these methods as “nonlinear acceleration” techniques.

3.2 Nonlinear Acceleration for Quadratic Minimization

In this section, we present the simplest form of nonlinear acceleration, which is often referred to as the *offline* nonlinear acceleration mechanism. We start with the main arguments underlying the technique, and present a few variants later in this section.

The core idea of the mechanism is to use a sequence of iterates $\{x_i\}_{i=0,\dots,k}$ provided by a first-order method for solving (3.1). On this basis, we generate a new approximation of a solution to (3.1) as a linear combination of past iterates, in the form $x_{\text{extr}} = \sum_{i=0}^k c_i x_i$. The point x_{extr} is commonly referred to as the *extrapolation* and can be

chosen in different ways. In classical nonlinear acceleration mechanisms, it is chosen for making $\|\nabla f(x_{\text{extr}})\|_2$ small, as in (3.5). In general, solving (3.5) is just as costly as solving (3.1), but the mechanism turns out to have an efficient formulation when minimizing quadratic functions of the form

$$f(x) = \frac{1}{2}\langle x - x_\star; \mathbf{H}(x - x_\star) \rangle + f_\star. \quad (3.6)$$

In this case, it is possible to find an explicit formula for (3.5). Indeed, the gradient of f is then a linear function, and because the coefficients $\{c_i\}_{i=0,\dots,k}$ sum to one, the gradient of the linear combination is equal to a linear combination of gradients:

$$\nabla f \left(\sum_{i=0}^k c_i x_i \right) = \mathbf{H} \left(\sum_{i=0}^k c_i x_i - x_\star \right) = \sum_{i=0}^k c_i \mathbf{H} (x_i - x_\star) = \sum_{i=0}^k c_i \nabla f(x_i). \quad (3.7)$$

It follows that (3.5) reduces to a simple quadratic program involving gradients of the past iterates. It can be formulated as

$$c_\star = \underset{c}{\operatorname{argmin}} \left\{ \left\| \sum_{i=0}^k c_i \nabla f(x_i) \right\|_2^2 : c^T \mathbf{1} = 1 \right\}. \quad (3.8)$$

For convenience, we use the following more compact form in the sequel

$$c_\star = \underset{c^T \mathbf{1}=1}{\operatorname{argmin}} \|\mathbf{G}c\|_2^2, \quad (3.9)$$

where $\mathbf{G} = [\nabla f(x_0), \dots, \nabla f(x_k)]$ is the matrix formed by concatenating past gradients. This quadratic subproblem requires solving a small linear system of $(k+1)$ equations. When $\mathbf{G}^T \mathbf{G}$ is invertible, an explicit solution is provided by

$$c_\star = \frac{z}{\sum_{i=0}^k z_i}, \quad \text{where } z = (\mathbf{G}^T \mathbf{G})^{-1} \mathbf{1}.$$

This mechanism is summarized in Algorithm 3.

3.2.1 Worst-case Convergence Bounds

In this section, we quantify the accuracy of nonlinear acceleration. In particular, we show that it is instance-optimal and achieves the same worst-case convergence rate as that Chebyshev's method (see Theorem 2.1 and Theorem 2.3) in the worst-case, as soon as the sequence

Algorithm 3 Nonlinear acceleration (offline version)

Input: Sequence of pairs $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$.

- 1: Form the matrix $\mathbf{G} = [\nabla f(x_0), \dots, \nabla f(x_k)]$, and compute $\mathbf{G}^T \mathbf{G}$.
- 2: Solve the linear system $(\mathbf{G}^T \mathbf{G})z = \mathbf{1}$, and set $c = \frac{z}{z^T \mathbf{1}}$.
- 3: Form the extrapolated point $x_{\text{extr}} = \sum_{i=0}^k c_i x_i$.

Output: Approximate solution x_{extr} .

$\{x_i\}_{i=0,\dots,k}$ is generated by a reasonable first-order method. Before going into the analysis, we introduce a few technical ingredients specifying what is a *reasonable* first-order method. In short, we require that $\{x_i\}_{i=0,\dots,k}$ is obtained from a “nondegenerate” first-order method. That is, we assume that the method uses $\nabla f(x_i)$ non-trivially for generating x_{i+1} (for all $i = 0, \dots, k-1$).

Definition 3.1 (Nondegenerate first-order method). Let $x_0 \in \mathbb{R}^d$ be an initial point, and $f : \mathbb{R}^d \rightarrow \mathbb{R}$ be a continuously differentiable convex function. A first-order method generates sequences of iterates $\{x_i\}_{i=0,1,\dots}$ such that for all $i = 1, 2, \dots$

$$x_i \in x_0 + \text{span}\{\nabla f(x_0), \dots, \nabla f(x_{i-1})\}.$$

A first-order method is *nondegenerate* if for all continuously differentiable convex function $f : \mathbb{R}^d \rightarrow \mathbb{R}$, all $i = 0, 1, \dots$, and all $x_0 \in \mathbb{R}^d$ there exists some $\{\alpha_j^{(i)}\}_{j=0,\dots,i} \subset \mathbb{R}$ with $\alpha_i^{(i)} \neq 0$ such that

$$x_{i+1} = x_0 + \sum_{j=0}^i \alpha_j^{(i)} \nabla f(x_j). \quad (3.10)$$

The proposition below shows that when the sequence $\{x_i\}_{i=0,1,\dots}$ is generated by a nondegenerate first-order method, the gradient of any iterate x_i can be written using a polynomial of degree *exactly* i .

Proposition 3.1. Let $x_0 \in \mathbb{R}^d$ be an initial point, $f : \mathbb{R}^d \rightarrow \mathbb{R}$ be a quadratic function in the form (3.6) with $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$, and let $\{x_i\}_{i=0,1,\dots}$ be generated by a nondegenerate first-order method. Then, for all $i = 0, 1, \dots$, there exists a polynomial P_i of degree *exactly* i such that $P_i(0) = 1$ and

$$\nabla f(x_i) = P_i(\mathbf{H}) \nabla f(x_0).$$

Proof. We proceed by induction. First, we have $P_0 = 1$ (P_0 has degree 0 and $P_0(0) = 1$) and hence

$$\nabla f(x_0) = P_0(\mathbf{H})\nabla f(x_0),$$

thereby trivially reaching the desired conclusion for $i = 0$.

We proceed with the induction hypothesis, assuming that the desired result holds for x_i . By Definition 3.1 (nondegenerate first-order method), and because f is a quadratic function (3.6), we have

$$\begin{aligned} \nabla f(x_{i+1}) &= \nabla f \left(x_0 + \sum_{j=0}^i \alpha_j^{(i)} \nabla f(x_j) \right) \\ &= \mathbf{H} \left(x_0 - x_\star + \sum_{j=0}^i \alpha_j^{(i)} \nabla f(x_j) \right) \\ &= \mathbf{H}(x_0 - x_\star) + \sum_{j=0}^i \alpha_j^{(i)} \mathbf{H} \nabla f(x_j) \\ &= \nabla f(x_0) + \sum_{j=0}^i \alpha_j^{(i)} \mathbf{H} \nabla f(x_j). \end{aligned}$$

Thanks to the induction hypothesis, we have $\nabla f(x_j) = P_j(\mathbf{H})\nabla f(x_0)$ with $P_j(0) = 1$ and $\deg(P_j) = j$ for $j \leq i$. For showing that P_{i+1} satisfies the desired claim, we start by expressing $\nabla f(x_{i+1})$ in terms of a polynomial:

$$\nabla f(x_{i+1}) = \underbrace{\left(P_0(\mathbf{H}) + \sum_{j=0}^i \alpha_j^{(i)} \mathbf{H} P_j(\mathbf{H}) \right)}_{=P_{i+1}(\mathbf{H})} \nabla f(x_0).$$

It is relatively straightforward to verify $P_{i+1}(0) = 1$ using the previous expression:

$$P_{i+1}(0) = P_0(0) + \sum_{j=0}^i \alpha_j^{(i)} \cdot 0 \cdot P_j(0) = P_0(0) = 1.$$

Finally, a minor reorganization of the expression of P_{i+1} allows writing

$$P_{i+1}(\mathbf{H}) = P_0(\mathbf{H}) + \underbrace{\sum_{j=0}^{i-1} \alpha_j^{(i)} \mathbf{H} P_j(\mathbf{H})}_{\text{degree} \leq i} + \alpha_i^{(i)} \mathbf{H} P_i(\mathbf{H}).$$

Nondegeneracy of the first-order method implies that there exists some $\alpha_i^{(i)} \neq 0$ such that the previous expression holds. Finally it follows from $\deg(P_i) = i$ (induction hypothesis) that $\deg(\mathbf{H} P_i(\mathbf{H})) = i + 1$, thereby reaching the desired claim. $\blacksquare$

Equipped with previous technical ingredients, one can show that Algorithm 3 is “instance-optimal” when applied to a nondegenerate first-order method. This means that the nonlinear acceleration algorithm finds the *best polynomial* given a *specific* quadratic function f —in opposition to the Chebyshev method that finds the best polynomial for a *class* of functions (Theorem 2.1). In other words, nonlinear acceleration *adaptively* looks for the best combination of previous iterates given the information stored in the previous gradients, while Chebyshev’s method uses the same worst-case optimal polynomial in all cases. Moreover, Algorithm 3 does *not* require knowledge of the smoothness or strong convexity parameters.

Theorem 3.1 (Instance-optimality of nonlinear acceleration). Let $x_0 \in \mathbb{R}^d$ be an initial point, f be the quadratic function from (3.6) with $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$, and $\{x_i\}_{i=0,\dots,k}$ be generated by a nondegenerate first-order method initiated from x_0 . For any $k \geq 0$, it holds that

$$\begin{aligned} \|\nabla f(x_{\text{extr}})\|_2 &= \min_{c^T \mathbf{1} = 1} \left\| \nabla f \left(\sum_{i=0}^k c_i x_i \right) \right\|_2 \\ &= \min_{\substack{P \in \mathcal{P}_k, \\ P(0)=1}} \|P(\mathbf{H}) \nabla f(x_0)\|_2, \end{aligned} \quad (3.11)$$

where x_{extr} is obtained from Algorithm 3 $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$, and $\mathcal{P}_k$ is the set of polynomials of degree at most k .

Proof. Since the coefficients c_i sum to one, we have the following equalities:

$$\begin{aligned}\nabla f(x_{\text{extr}}) &= \nabla f\left(\sum_{i=0}^k c_i x_i\right) = \mathbf{H}\left(\sum_{i=0}^k c_i (x_i - x_*)\right) \\ &= \left(\sum_{i=0}^k c_i \mathbf{H}(x_i - x_*)\right) = \sum_{i=0}^k c_i \nabla f(x_i).\end{aligned}$$

Therefore,

$$\|\nabla f(x_{\text{extr}})\|_2 = \left\| \sum_{i=0}^k c_i \nabla f(x_i) \right\|_2.$$

We now use the definition of a first-order method, which yields iterates x_i such that

$$x_i \in x_0 + \text{span}\{\nabla f(x_0), \dots, \nabla f(x_{i-1})\}.$$

From Section 2.3, Proposition 2.1, it follows that x_i can be written as

$$x_i = x_* + P_i(\mathbf{H})(x_0 - x_*), \quad P_i \in \mathcal{P}_i, \quad P_i(0) = 1.$$

It also holds that its gradient can be written using the same polynomial:

$$\begin{aligned}\nabla f(x_i) &= \mathbf{H}(x_i - x_*) = \mathbf{H}P_i(\mathbf{H})(x_0 - x_*) = P_i(\mathbf{H})(\mathbf{H}(x_0 - x_*)) \\ &= P_i(\mathbf{H})\nabla f(x_0).\end{aligned}\tag{3.12}$$

By substituting this expression in the objective of (3.8), we obtain

$$\min_{c^T \mathbf{1} = 1} \left\| \sum_{i=0}^k c_i \nabla f(x_i) \right\|_2^2 = \min_{c^T \mathbf{1} = 1} \left\| \left(\sum_{i=0}^k c_i P_i(\mathbf{H}) \right) \nabla f(x_0) \right\|_2^2. \tag{3.13}$$

Since the iterates $\{x_i\}_{i=0,\dots,k}$ are generated by a nondegenerate first-order method, all polynomials P_i have different degrees. Therefore, the polynomials $\{P_i\}_{i=0,\dots,k}$ are linearly independent, and hence $\{P_i\}_{i=0,\dots,k}$ is a basis for the space $\mathcal{P}_k$. Finally, because $P_i(0) = 1$ and $c^T \mathbf{1} = 1$, we can rephrase the objective (3.13) as

$$\min_{c^T \mathbf{1} = 1} \left\| \left(\sum_{i=0}^k c_i P_i(\mathbf{H}) \right) \nabla f(x_0) \right\|_2 = \min_{\substack{P \in \mathcal{P}_k, \\ P(0) = 1}} \|P(\mathbf{H})\nabla f(x_0)\|_2.$$

The convergence rate is thus given by

$$\|\nabla f(x_{\text{extr}})\|_2 = \left\| \sum_{i=0}^k c_i \nabla f(x_i) \right\|_2 = \min_{\substack{P \in \mathcal{P}_k, \\ P(0)=1}} \|P(\mathbf{H}) \nabla f(x_0)\|_2.$$

■

This theorem shows that the worst-case convergence rate is essentially controlled by the optimal value of a minimization problem. The minimum in (3.11) can be bounded using a Chebyshev argument similar to the main argument used in Section 2.

Corollary 3.2. Let $x_0 \in \mathbb{R}^d$ be an initial point, f be the quadratic function from (3.6) with $\mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}$, and $\{x_i\}_{i=0,\dots,k}$ be generated by a nondegenerate first-order method initiated from x_0 . Then, for any $k \geq 0$, it holds that

$$\|\nabla f(x_{\text{extr}})\|_2 \leq \frac{2}{\xi^k + \xi^{-k}} \|\nabla f(x_0)\|_2, \quad \xi = \frac{\sqrt{\frac{L}{\mu}} + 1}{\sqrt{\frac{L}{\mu}} - 1}.$$

where x_{extr} is the output of Algorithm 3 applied to $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$.

Proof. We use the shifted Chebyshev polynomial from Theorem 2.1 as a feasible solution of the minimization problem (3.11). ■

As for the Chebyshev method, it is possible to show that the convergence rate cannot be improved as it matches that of the corresponding lower bound, which can be obtained by adapting Theorem 2.3 to gradient norms; see e.g., [26, Proposition 12.3.2].

Remark 3.1 (Finite-time convergence). When $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$ is generated by a nondegenerate first-order method and when k is large enough, nonlinear acceleration eventually converges *exactly* to the minimizer of the quadratic function f (this follows easily from analogies with conjugate gradient-type methods; see, e.g., Section 2.4). More formally, for all $k \geq d$ it holds that

$$x_{\text{extr}} = x_\star, \tag{3.14}$$

where x_{extr} is the output of Algorithm 3 applied to $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$. This is a natural consequence of the fact that x_{extr} is the best iterate over the set $x_0 + \text{span}\{\nabla f(x_0), \dots, \nabla f(x_{k-1})\}$, as provided by (3.8).

3.2.2 Computational Complexity

The computational complexity of Algorithm 3 is $O(dk^2 + k^3)$, where k is the length of the sequence of gradients and d is the dimension of the ambient space. The first term originates from the matrix-matrix multiplication $\mathbf{G}^T \mathbf{G}$ in step 1, and the second one comes from solving a $(k+1) \times (k+1)$ matrix in step 2. The length k being often much smaller than d , the resulting complexity is typically $O(dk^2)$.

Low-rank updates. When using the nonlinear acceleration method in parallel with a first-order method generating a growing sequence of iterates $\{x_i\}_{i=0,\dots,k}$, an extrapolation step can be computed each time a new iterate is produced. we can reduce the per iteration complexity up to $O(dk)$ by computing the matrix $\mathbf{G}^T \mathbf{G}$ and the coefficients $c_\star$ using low-rank updates [40].

Limited-memory. Because the iteration complexity of nonlinear acceleration grows with the length of the input sequence, it might become costly to compute an extrapolation. It is therefore common to use nonlinear acceleration only with the last few iterates produced by the first-order method. More formally, if we impose a maximum memory of m pairs, we compute the extrapolation as follow when $k \geq m$. More formally, if we impose a maximum memory of m pairs, one can use Algorithm 3 with input $\{(x_{k-m+i}, \nabla f(x_{k-m+i}))\}_{i=1,\dots,m}$.

3.2.3 Online Nonlinear Acceleration

So far, we have seen a post-processing procedure that generates an extrapolated point x_{extr} from a sequence of pairs $\{x_i, \nabla f(x_i)\}_{i=0,\dots,k}$. If this sequence is generated by a nondegenerate first-order method, then Corollary 3.2 shows that the gradient of the extrapolated point x_{extr} converges to zero at an optimal worst-case convergence rate, without any hyper-parameters. Perhaps surprisingly, Algorithm 3 is itself not a nondegenerate first-order method, and can therefore not be used recursively as is.

In what follows, we introduce a *mixing* parameter. This parameter transforms the nonlinear acceleration mechanism to a nondegenerate

first-order method without hurting its worst-case performance. This enables using nonlinear acceleration recursively for generating the whole sequence $\{x_i\}_{i=1,2,\dots}$. This technique is often referred to as *online nonlinear acceleration*.

Mixing parameter. The idea underlying the mixing parameter is fairly simple: instead of combining previous iterates, we combine *gradient steps* as follows:

$$x_{\text{extr}}^{\text{mixing}} = \sum_{i=0}^k c_i (x_i - h \nabla f(x_i)), \quad (3.15)$$

as provided by Algorithm 4. Furthermore, the use of an appropriate step size h can even slightly improve the worst-case convergence speed of Algorithm 3.

Algorithm 4 Nonlinear acceleration (with mixing)

Input: Sequence of pairs $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$, mixing parameter h .

- 1: Form the matrix $\mathbf{G} = [\nabla f(x_0), \dots, \nabla f(x_k)]$, and compute $\mathbf{G}^T \mathbf{G}$.
- 2: Solve the linear system $(\mathbf{G}^T \mathbf{G})z = \mathbf{1}$, and compute $c = \frac{z}{z^T \mathbf{1}}$.
- 3: Form the extrapolated point $x_{\text{extr}} = \sum_{i=0}^k c_i (x_i - h \nabla f(x_i))$.

Output: Approximate solution x_{extr} .

Intuitively, this *mixing* between iterates and gradients emulates a gradient step on the extrapolated point from Algorithm 3, that we call here $x_{\text{extr}}^{\text{offline}} = \sum_{i=0}^k c_i x_i$,

$$\begin{aligned} x_{\text{extr}}^{\text{mixing}} &= \sum_{i=0}^k c_i x_i - h \sum_{i=0}^k c_i \nabla f(x_i) \\ &= \sum_{i=0}^k c_i x_i - h \nabla f \left(\sum_{i=0}^k c_i x_i \right) \\ &= x_{\text{extr}}^{\text{offline}} - h \nabla f(x_{\text{extr}}^{\text{offline}}), \end{aligned}$$

where the second equality comes from the fact that f is a quadratic function, and that $\sum_{i=0}^k c_i = 1$.

This mixing parameter requires tuning one hyper-parameter h , which can be chosen in various ways. Proposition 3.2 shows that the mixing

strategy slightly improves the performance of nonlinear acceleration if h is set properly.

Proposition 3.2. Let $x_0 \in \mathbb{R}^d$ be an initial point, f be the quadratic function from (3.6) with $\mu\mathbf{I} \preceq \mathbf{H} \preceq L\mathbf{I}$, and $\{x_i\}_{i=0,\dots,k}$ be generated by a nondegenerate first-order method initiated from x_0 . Then, for any $k \geq 0$, it holds that

$$\|\nabla f(x_{\text{extr}}^{\text{mixing}})\|_2 \leq C_h \|\nabla f(x_{\text{extr}}^{\text{offline}})\|_2,$$

where $x_{\text{extr}}^{\text{offline}}$ is obtained from Algorithm 3 applied to $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$, $x_{\text{extr}}^{\text{mixing}}$ is the extrapolation with mixing from (3.15), and

$$C_h = \max \{1 - \mu h ; Lh - 1\}.$$

Moreover, the factor C_h is guaranteed to be smaller than one if $h \in (0, \frac{2}{L})$, and takes its minimal value at $h = \frac{2}{L+\mu}$.

Proof. We start with the identity

$$x_{\text{extr}}^{\text{mixing}} = x_{\text{extr}}^{\text{offline}} - h \nabla f(x_{\text{extr}}^{\text{offline}}).$$

Because f is the quadratic function (3.6), the gradient of $x_{\text{extr}}^{\text{mixing}}$ reads

$$\begin{aligned} \nabla f(x_{\text{extr}}^{\text{mixing}}) &= \mathbf{H}(x_{\text{extr}}^{\text{mixing}} - x_\star) \\ &= \mathbf{H}(x_{\text{extr}}^{\text{offline}} - h \nabla f(x_{\text{extr}}^{\text{offline}}) - x_\star) \\ &= \nabla f(x_{\text{extr}}^{\text{offline}}) - \mathbf{H}h \nabla f(x_{\text{extr}}^{\text{offline}}) \\ &= (I - \mathbf{H}h) \nabla f(x_{\text{extr}}^{\text{offline}}). \end{aligned}$$

Therefore, we have the bound

$$\|\nabla f(x_{\text{extr}}^{\text{mixing}})\|_2 \leq \|I - \mathbf{H}h\|_2 \|\nabla f(x_{\text{extr}}^{\text{offline}})\|_2,$$

and the desired result follows from $\mu\mathbf{I} \preceq \mathbf{H} \preceq L\mathbf{I}$. ■

Online nonlinear acceleration method. As previously underlined, the mixing parameter transforms the nonlinear acceleration method into a nondegenerate first-order method. We can therefore use it recursively. The online variant of the nonlinear acceleration technique, with limited memory, is provided in Algorithm 5, using Algorithm 4 as a subroutine.

One should note that when $m = \infty$ (no memory restriction), the worst-case performance of offline version of nonlinear acceleration with mixing (Algorithm 4) is also valid for its online variant (Algorithm 5). It follows from Proposition 3.2 that the worst-case performance of Algorithm 4 and Algorithm 5 is no worse than that of offline version of nonlinear acceleration (Algorithm 3), provided by Corollary 3.2.

Algorithm 5 Nonlinear acceleration (online version, limited memory)

Input: A differentiable function f , initial point x_0 , mixing parameter h , maximum memory parameter m (optional, $m = \infty$ by default).

- 1: **Initialize** Empty sequence $\mathcal{S}$ of pairs iterate/gradient.
- 2: **for** $k = 0, \dots$ **do**
- 3: Compute $\nabla f(x_k)$; append the pair $\mathcal{S} \leftarrow \mathcal{S} \cup \{(x_k, \nabla f(x_k))\}$.
- 4: **if** $k \geq m$ **then**
- 5: Discard the oldest pair from $\mathcal{S}$.
- 6: **end if**
- 7: Compute the extrapolation $x_{k+1} = [\text{Algorithm 4}](\mathcal{S}, h)$.
- 8: **end for**

Output: Approximate solution x_{k+1} .

In the next section, we see that nonlinear acceleration technique might suffer from serious instability issues when applied beyond quadratic minimization. Perhaps luckily, a simple regularization technique allows stabilizing the procedure beyond quadratics.

3.3 Regularized Nonlinear Acceleration Beyond Quadratics

Nonlinear acceleration suffers some serious drawbacks when used outside the restricted setting of quadratic functions. In fact, Algorithm 3 and Algorithm 4 are numerically highly unstable. This problem originates from the conditioning of the matrix $\mathbf{G}^T \mathbf{G}$, used to compute the coefficients $\{c_i\}_{i=0,\dots,k}$. To illustrate this statement, assume we run Algorithm 3 with a noisy sequence of gradients $\{(x_i, \nabla f(x_i) + e_i)\}_{i=0,\dots,k}$ where the sequence $\{e_i\}_{i=0,\dots,k}$ is such that $\|e_i\|_2 \leq \epsilon$ for some $\epsilon > 0$. Scieur, d’Aspremont, and Bach [41, Proposition 3.1] show that the relative distance between $\tilde{c}$ (the coefficients computed using the noisy sequence

above) and its noise-free version c satisfies

$$\frac{\|c - \tilde{c}\|_2}{\|c\|_2} = O\left(\|\mathbf{E}\|_2 \|((\mathbf{G} + \mathbf{E})^T(\mathbf{G} + \mathbf{E}))^{-1}\|_2\right). \quad (3.16)$$

where $\mathbf{E} \triangleq [e_0, \dots, e_k]$ is the noise matrix. In this bound, the perturbation impacts the solution proportionally to its norm and to the conditioning of $\mathbf{G} + \mathbf{E}$.

Unfortunately, even for small perturbations, the condition number of $\mathbf{G} + \mathbf{E}$ and the norm of the vector c are usually *huge*. In fact, $\mathbf{G}$ has a *Krylov* matrix structure, which is notoriously poorly conditioned [42]. Thereby, even a very small perturbation $\mathbf{E}$ might have a significant impact on performance. For illustrating this, let us briefly illustrate the link between $\mathbf{G}$ and Krylov matrices: consider using gradient descent with step size γ on a quadratic function; the iterates follow the rule

$$x_{k+1} - x_\star = (1 - \gamma\mathbf{H})^k(x_0 - x_\star) \quad \Leftrightarrow \quad \nabla f(x_{k+1}) = (1 - \gamma\mathbf{H})^k \nabla f(x_0).$$

Thereby, a matrix $\mathbf{G}$ formed by these expressions has the form

$$\mathbf{G} = \left[\nabla f(x_0), (\mathbf{I} - \gamma\mathbf{H})\nabla f(x_0), (\mathbf{I} - \gamma\mathbf{H})^2\nabla f(x_0), \dots \right],$$

which shows that $\mathbf{G}$ is in fact a *Krylov* matrix—by definition, a Krylov matrix K associated with a matrix A and vector v is defined as $K = [v, Av, A^2v, \dots]$.

In Figure 3.1, we show the norm of c and the condition number of the matrix $\mathbf{G}^T\mathbf{G}$ when it is formed from iterates of gradient descent, accelerated gradient descent (see Section 4), and nonlinear acceleration (in the online setting, see Algorithm 5) for minimizing some randomly generated quadratic function. Figure 3.1a shows that even after 3 iterations, the system can already be considered singular (i.e., the condition number exceeds 10^{16}).

For stabilizing the method, it is common to regularize the linear system. The resulting algorithm is often referred to as regularized nonlinear acceleration (RNA) [41]. The following section is devoted to some theoretical properties of this method.

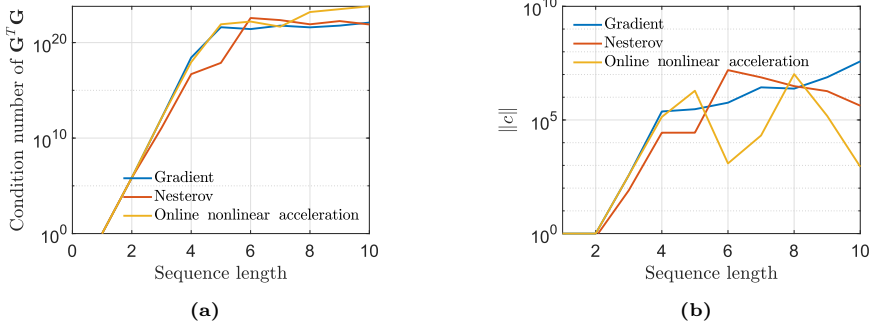


Figure 3.1: Illustration of the sensitivity of nonlinear acceleration when applying nonlinear acceleration to gradient descent, to Nesterov’s method (see Section 4), and using its online variant (Algorithm 5) to minimize some random quadratic function. Figure 3.1a: the condition number of the matrix $\mathbf{G}^T \mathbf{G}$, which grows exponentially with its size (the plateau on the right is caused by numerical errors). Figure 3.1b: the norm of the vector of coefficients c .

3.3.1 Regularized Nonlinear Acceleration

Regularized nonlinear acceleration (RNA) consists of using Algorithm 3 with a regularization, thereby rendering the method less sensitive to noise. In short, the base operation underlying RNA is to solve

$$\operatorname{argmin}_{c^T \mathbf{1} = 1} \frac{\|\mathbf{G}c\|_2^2}{\|\mathbf{G}\|_2^2} + \lambda \|c - c_{\text{ref}}\|_2^2 \quad (3.17)$$

in the variable $c \in \mathbb{R}^k$, where c_{ref} is some reference vector. The effect of regularization is therefore to force c to be close to c_{ref} . Of course, it makes more sense to pick c_{ref} summing to one. A common choice is to pick $c_{\text{ref}} = \mathbf{1}/k$, which would enforce the procedure to be “not too far” from a simple averaging of the iterates. Another possibility is $c_{\text{ref}} = [\mathbf{0}_{k-1}, 1]$, which puts more weight on the last iterate. The division by $\|\mathbf{G}\|_2^2$ is for scaling purposes, as it makes λ dimensionless. The resulting method is slightly more complicated than its previous version without regularization and is provided in Algorithm 6. When $c_{\text{ref}} = \mathbf{1}/k$, the procedure simplifies to Algorithm 7.

Online regularized nonlinear acceleration. As in the quadratic case, Algorithm 6 and Algorithm 7 could be used as subroutines in the

Algorithm 6 Regularized nonlinear acceleration (RNA)

Input: Sequence of pairs $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$, mixing parameter h , regularization term $\lambda > 0$, reference vector c_{ref} .

- 1: Form $\mathbf{G} = [\nabla f(x_0), \dots, \nabla f(x_k)]$, compute $\mathcal{G} = \frac{\mathbf{G}^T \mathbf{G}}{\|\mathbf{G}^T \mathbf{G}\|_2}$.
- 2: Solve the linear system $(\mathcal{G} + \lambda \mathbf{I})w = \lambda c_{\text{ref}}$.
- 3: Solve the linear system $(\mathcal{G} + \lambda \mathbf{I})z = \mathbf{1}$.
- 4: Compute the coefficients $c = w + z \frac{(1-w^T \mathbf{1})}{z^T \mathbf{1}}$.
- 5: Form the extrapolated point $x_{\text{extr}} = \sum_{i=0}^k c_i (x_i - h \nabla f(x_i))$.

Output: Approximate solution x_{extr} .

Algorithm 7 Regularized nonlinear acceleration (with $c_{\text{ref}} = \mathbf{1}/k$)

Input: Sequence of pairs $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$, mixing parameter h , regularization term $\lambda > 0$.

- 1: Form $\mathbf{G} = [\nabla f(x_0), \dots, \nabla f(x_k)]$ and compute $\mathcal{G} = \frac{\mathbf{G}^T \mathbf{G}}{\|\mathbf{G}^T \mathbf{G}\|_2}$.
- 2: Solve the linear system $(\mathcal{G} + \lambda \mathbf{I})z = \mathbf{1}/k$.
- 3: Compute the coefficients $c = \frac{z}{z^T \mathbf{1}}$.
- 4: Form the extrapolated point $x_{\text{extr}} = \sum_{i=0}^k c_i (x_i - h \nabla f(x_i))$.

Output: Approximate solution x_{extr} .

online nonlinear acceleration method (Algorithm 5), thereby forming the regularized version of the online acceleration algorithm.

3.3.2 Perturbed Linear Gradients

In this section, we consider the problem of minimizing a twice continuously differentiable convex function f , as in (3.1), beyond quadratic problems. For doing that, we introduce *perturbed linear gradients*. As before, we consider iterates $\{x_i\}_{i=0,1,\dots}$ originating from a first-order method satisfying

$$x_{k+1} \in x_0 + \text{span}\{\nabla f(x_0), \nabla f(x_1), \dots, \nabla f(x_k)\}.$$

However, $\nabla f(x)$ is now no longer the gradient of a quadratic function. Instead, gradients of f can be written as a sum of the gradients of a quadratic function with a perturbation term $e(x)$, as follows:

$$\nabla f(x) = \mathbf{H}(x - x_\star) + e(x), \quad \text{with } \mathbf{H} : \mathbf{0} \prec \mu \mathbf{I} \preceq \mathbf{H} \preceq L \mathbf{I}. \quad (3.18)$$

Indeed, it follows from twice continuous differentiability of f that

$$f(x) = f_\star + \underbrace{\langle \nabla f(x_\star); x - x_\star \rangle}_{=0} + \frac{1}{2} \langle x - x_\star; \nabla^2 f(x_\star)(x - x_\star) \rangle + O(\|x - x_\star\|_2^3).$$

Therefore, $f(x)$ can be approximated by the quadratic function (3.18) with $\mathbf{H} = \nabla^2 f(x_\star)$. Similarly, its gradient reads

$$\nabla f(x) = \underbrace{\nabla f(x_\star)}_{=0} + \nabla^2 f(x_\star)(x - x_\star) + e(x) = \mathbf{H}(x - x_\star) + e(x),$$

where $e(x)$ is the first-order Taylor remainder of the gradient. Thus, minimizing a non-quadratic function is equivalent to minimizing a perturbed quadratic one with a second-order error on its gradient:

$$e(x_k) = \nabla f(x_k) - \mathbf{H}(x_k - x_\star) \quad \left(\Rightarrow \quad \|e(x_k)\|_2 = O(\|x_k - x_\star\|_2^2) \right), \quad (3.19)$$

where $\mathbf{H} = \nabla^2 f(x_\star)$.

3.3.3 Convergence Bound

Using a perturbation argument, it is possible to derive a convergence guarantee for RNA. We state here a simplified version of [43, Theorem 3.2], which describes how regularization balances acceleration and stability in Algorithm 7. We discuss convergence rates in greater detail in what follows.

Theorem 3.3. Let $x_0 \in \mathbb{R}^d$, f be a twice continuously differentiable function with minimum $x_\star$ and whose gradient ∇f follows (3.18) with $\|e(x_i)\|_2 \leq \epsilon$. Let $\{x_i\}_{i=0,\dots,k}$ be generated by a nondegenerate first-order method initiated from x_0 . Then, for any $k \geq 0$, it holds that

$$\begin{aligned} & \|\mathbf{H}(x_{\text{extr}} - x_\star)\|_2 \\ & \leq \|\mathbf{I} - h\mathbf{H}\|_2 \left(\underbrace{V_k^{[\mu,L]}(\lambda) \|\mathbf{H}(x_0 - x_\star)\|_2}_{\text{acceleration}} + \underbrace{O\left(\sqrt{1 + \frac{1}{\lambda}\epsilon}\right)}_{\text{stability}} \right), \end{aligned}$$

where x_{extr} is the output of Algorithm 7 applied to $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$ with parameters $h, \lambda > 0$, and $V_k^{[\mu,L]}(\lambda)$ is a constant that corresponds to the maximum value on the interval $[\mu, L]$ of the *regularized Chebyshev polynomial*, i.e.,

$$\begin{aligned} V_k^{[\mu,L]}(\lambda) &= \max_{x \in [\mu, L]} |C_k^{[\mu,L],\lambda}(x)|, \\ \text{with } C_k^{[\mu,L],\lambda} &= \underset{\substack{P \in \mathcal{P}_k, \\ P(0)=1}}{\operatorname{argmin}} \max_{x \in [\mu, L]} P^2(x) + \lambda \|\mathbf{G}^T \mathbf{G}\|_2 \|P\|_2^2, \end{aligned} \quad (3.20)$$

where $\|P\|_2$ is the norm of the vector of coefficients of the polynomial P .

This theorem states that regularization helps stabilizing the algorithm while slowing down the convergence rate. The regularized Chebyshev polynomial is somehow a mid-point between the classical shifted Chebyshev polynomial $C_k^{[\mu,L]}$ (from (2.15)) and the polynomial whose coefficients are defined by $1/k$ (the polynomial that averages the iterates $\{x_i\}_{i=0,\dots,k}$). By construction, its maximum value is always larger than that of the Chebyshev polynomial, but the norm of its coefficients is smaller. Unfortunately, there is as yet no known explicit expression of the regularized Chebyshev polynomial. To the best of our knowledge, its value can nevertheless be computed numerically [44].

This mid-point between Chebyshev coefficients and the simple averaging of iterates is also natural in the context of noisy iterations. When the noise is negligible, a small regularization parameter combines the iterates $\{x_i\}_{i=0,\dots,k}$ using nearly the classical Chebyshev weights. When

the noise is more substantial, a larger regularization parameter brings the vector of coefficients c closer to the average $\mathbf{1}/k$, thereby improving the “stability term” while rendering the “acceleration” less effective.

3.3.4 Asymptotic Convergence Rate

We briefly discuss the behavior of RNA when the initial point x_0 approaches the solution $x_\star$. In particular, the next proposition shows that if the perturbation magnitude ϵ decreases faster than $\|\mathbf{H}(x_0 - x_\star)\|_2$, the parameter λ can be adjusted to ensure an asymptotic convergence rate comparable to that of the Chebyshev method on quadratic problems (see Section 2).

Informally, the theorem exploits the fact that as x_0 approaches $x_\star$, f gets closer to its quadratic approximation around $x_\star$. Thereby, an appropriate tuning of RNA allows matching (asymptotically) the convergence rate of nonlinear acceleration on quadratics (see Theorem 3.1).

Proposition 3.3. Let $x_0 \in \mathbb{R}^d$, f be a twice continuously differentiable function with minimum $x_\star$, whose gradient ∇f follows (3.18) with $\|e(x_i)\|_2 \leq \epsilon$. Let $\{x_i\}_{i=0,\dots,k}$ be generated by a nondegenerate first-order method initiated from x_0 . If we have

$$\epsilon = O(\|x_0 - x_\star\|_2^\alpha), \quad \alpha > 1,$$

and if we set $\lambda \propto \|x_0 - x_\star\|_2^s$ (proportional to $\|x_0 - x_\star\|_2^s$), where $0 < s < 2(\alpha - 1)$, then it holds that

$$\lim_{x_0 \rightarrow x_\star} \frac{\|\mathbf{H}(x_{\text{extr}} - x_\star)\|_2}{\|\mathbf{H}(x_0 - x_\star)\|_2} \leq \|\mathbf{I} - h\mathbf{H}\|_2 \frac{2}{\xi^k + \xi^{-k}} \quad \text{where } \xi = \frac{\sqrt{\frac{L}{\mu}} + 1}{\sqrt{\frac{L}{\mu}} - 1},$$

where x_{extr} is the output of Algorithm 7 applied to the sequence $\{(x_i, \nabla f(x_i))\}_{i=0,\dots,k}$ with parameters $h, \lambda > 0$.

Proof. To simplify the notation, set $R \triangleq \|\mathbf{H}(x_0 - x_\star)\|_2$. We start from the result of Theorem 3.3 and divide both sides by R :

$$\frac{\|\mathbf{H}(x_{\text{extr}} - x_\star)\|_2}{R} \leq \|\mathbf{I} - h\mathbf{H}\|_2 \left(V_k^{[\mu, L]}(\lambda) + O\left(\sqrt{1 + \frac{1}{\lambda} \frac{\epsilon}{R}}\right) \right).$$

Since $\lambda \propto R^s$ and $\epsilon = O(R^\alpha)$,

$$\begin{aligned} & \frac{\|\mathbf{H}(x_{\text{extr}} - x_\star)\|_2}{R} \\ & \leq \|\mathbf{I} - h\mathbf{H}\|_2 \left(V_k^{[\mu, L]}(\lambda) + \sqrt{O(R^{2(\alpha-1)}) + O(R^{2(\alpha-1)-s})} \right). \end{aligned}$$

When $x_0 \rightarrow x_\star$, we have $R \rightarrow 0$ and

$$\begin{aligned} R^{2(\alpha-1)} &\rightarrow 0 && (\text{since } \alpha > 1), \\ R^{2(\alpha-1)-s} &\rightarrow 0 && (\text{since } s < 2(\alpha-1)). \end{aligned}$$

Finally, in (3.20) the regularization parameter $\lambda\|\mathbf{G}^T\mathbf{G}\|_2 = O(R^2) \rightarrow 0$. Since the (non-regularized) shifted Chebyshev polynomial $C_k^{[\mu, L]} = C_k^{[\mu, L], 0}$ is a feasible solution of (3.20), we have the following bounds:

$$\begin{aligned} \left(V_k^{[\mu, L]}(\lambda) \right)^2 &= \max_{x \in [\mu, L]} |C_k^{[\mu, L], \lambda}(x)|^2, \\ &\leq \left\{ \max_{x \in [\mu, L]} |C_k^{[\mu, L], \lambda}(x)|^2 \right\} + \lambda\|\mathbf{G}^T\mathbf{G}\|_2 \|C_k^{[\mu, L], \lambda}\|_2^2, \\ &= \min_{\substack{P \in \mathcal{P}_k, \\ P(0)=1}} \max_{x \in [\mu, L]} P^2(x) + \lambda\|\mathbf{G}^T\mathbf{G}\|_2 \|P\|_2^2, \\ &\leq \left\{ \max_{x \in [\mu, L]} |C_k^{[\mu, L]}(x)|^2 \right\} + \lambda\|\mathbf{G}^T\mathbf{G}\|_2 \|C_k^{[\mu, L]}\|_2^2. \end{aligned}$$

As $\lambda \rightarrow 0$ we have that the upper bound on $(V_k^{[\mu, L]}(\lambda))$ converges to the maximum value of the regular (shifted) Chebyshev polynomial, thereby reaching the desired claim. $\blacksquare$

In short, the previous theorem states that the asymptotic convergence rate matches the rate of Chebyshev's method as soon as $\lambda \propto \|x_0 - x_\star\|_2^s$ is decreasing (condition $s > 0$), but not too quickly compared to the perturbation magnitude $\epsilon = O(\|x_0 - x_\star\|_2^\alpha)$ (condition $s < 2(\alpha - 1)$), which is achievable only when $\alpha > 1$. This condition is met, for instance, when accelerating twice continuously differentiable functions with gradient descent: the error decreases as $O(\|x_0 - x_\star\|_2^2)$, see (3.19), and therefore $\alpha = 2 > 1$.