Mathematical Programming (2020) 179:47–83
https://doi.org/10.1007/s10107-018-1319-8

FULL LENGTH PAPER

Series A

Regularized nonlinear acceleration

Damien Scieur[1] · **Alexandre d'Aspremont**[2] · **Francis Bach**[1]

Received: 24 November 2017 / Accepted: 9 August 2018 / Published online: 21 August 2018

Abstract
We describe a convergence acceleration technique for unconstrained optimization problems. Our scheme computes estimates of the optimum from a nonlinear average of the iterates produced by any optimization method. The weights in this average are computed via a simple linear system, whose solution can be updated online. This acceleration scheme runs in parallel to the base algorithm, providing improved estimates of the solution on the fly, while the original optimization method is running. Numerical experiments are detailed on classical classification problems.

Keywords Acceleration · ε-Algorithm · Extrapolation

Mathematics Subject Classification 90C30

1 Introduction

Suppose we seek to solve the following optimization problem

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1}$$

in the variable $x \in \mathbb{R}^n$, where $f(x)$ is strongly convex with parameter μ with respect to the Euclidean norm, and has a Lipschitz continuous gradient with parameter L with

A subset of these results appeared at the 2016 NIPS conference under the same title.

✉ Alexandre d'Aspremont
aspremon@ens.fr

Damien Scieur
damien.scieur@inria.fr

Francis Bach
francis.bach@inria.fr

[1] INRIA & D.I., École Normale Supérieure, Paris, France

[2] CNRS & D.I., UMR 8548, École Normale Supérieure, Paris, France

respect to the same norm. Assume we solve this problem using the *fixed-point iteration*

$$\tilde{x}_{i+1} = g(\tilde{x}_i), \quad \text{for } i = 0, ..., k, \tag{FPI}$$

where $\tilde{x}_i \in \mathbb{R}^n$ and k is the number of iterations. This iteration is typically produced by an optimization algorithm, e.g. the gradient method with fixed step size, written

$$\tilde{x}_{i+1} = \tilde{x}_i - h\nabla f(\tilde{x}_i), \quad \text{for } i = 0, ..., k, \tag{2}$$

with step length $h > 0$. Here, we will focus on improving our estimates of the solution to problem (1) by tracking only the iterate sequence $\tilde{x}_i$ produced by an optimization algorithm, without any further calls to oracles on $g(x)$.

Since the publication of Nesterov's optimal first-order smooth convex minimization algorithm [21], significant efforts have been focused on either providing more interpretable views on current acceleration techniques, or on replicating these complexity gains using different, more intuitive schemes. Early efforts sought to directly extend the original acceleration result in [21] to broader function classes [19], allow for generic metrics, line searches, produce simpler proofs [3,23] or adaptive accelerated algorithms [24], etc. More recently however, several authors [7,15] have started using classical results from control theory to obtain numerical bounds on convergence rates that match the optimal rates. Others have studied the second order ODEs obtained as the limit for small step sizes of classical accelerated schemes, to better understand their convergence [29,31]. Finally, recent results have also shown how to wrap classical algorithms in an outer optimization loop, to accelerate convergence and reach optimal complexity bounds [17] on certain structured problems.

Here, we take a significantly different approach to convergence acceleration stemming from classical results in numerical analysis. We use the iterates produced by any (converging) optimization algorithm, and estimate the solution directly from this sequence, assuming only some regularity conditions on the function to minimize. Our scheme is based on the idea behind Aitken's Δ^2-algorithm [1], generalized as the Shanks transform [26], whose recursive formulation is known as the ε-algorithm [32] (see e.g. [5,27] for a survey). In a nutshell, these methods fit geometrical models to linearly converging sequences, then extrapolate their limit from the fitted model.

In a sense, this approach is more statistical in nature. It assumes an approximately linear model holds for iterations near the optimum, and estimates this model using the iterates. In fact, Wynn's algorithm [32] is directly connected to the Levinson–Durbin algorithm [8,16] used to solve Toeplitz systems recursively and fit autoregressive models (the Shanks transform solves Hankel systems, but this is essentially the same problem [13]). The key difference in these extrapolation techniques is that estimating the autocovariance operator A is not required, as we only focus on the limit. Moreover, the method presents strong links with the conjugate gradient when applied to unconstrained quadratic optimization, but does not further calls to the operator.

We start from a formulation of these techniques known as Anderson Acceleration [2], Mešina's Algorithm [18] or minimal polynomial extrapolation (MPE) [27,28]. They use the minimal polynomial of the linear operator driving iterations to estimate

the optimum by a nonlinear average of the iterates (i.e. computing a weighted average using weights which are nonlinear functions of the iterates).

Our contribution here is to regularize this procedure and produce explicit bounds on the distance to optimality by controlling stability, thus explicitly quantifying acceleration. We show that these extrapolation algorithms reach optimal performance (asymptotically) and describe several numerical examples where these stabilized estimates often speed up convergence by an order of magnitude. So far, for all the techniques cited above, no proofs of convergence of the estimates were given when the estimation process became unstable. Furthermore, the acceleration scheme runs in parallel with the original algorithm, providing improved estimates of the solution on the fly, while the original method is progressing, so its numerical complexity is marginal.

The paper is organized as follows. In Sect. 2 we recall basic results behind the acceleration for linear iterations. Then, in Sect. 3, we generalize these results to nonlinear iterations and show how to fully control the impact of nonlinearity. We use these results to derive explicit bounds on the acceleration performance of our estimates. In Sect. 4 we connect the acceleration methods to the conjugate gradient method and Nesterov's method. Finally, we present numerical results in Sect. 5.

2 Convergence acceleration

We begin by recalling the core arguments behind convergence acceleration. These ideas have taken various forms over time, known for example as *Anderson acceleration* [2], the *Eddy–Mesina method* [9,18] and *minimal polynomial extrapolation* [6,28]. The core idea behind these methods is to use a Taylor expansion of the function g in (FPI) to approximate the fixed point iterations by a vector autoregressive model, then compute a weighted mean of the iterates $\tilde{x}_i$ to produce a better estimate of the limit x^*. In this paper, we assume x^* unique.

Suppose $g(x)$ is differentiable and let G be the Jacobian of g evaluated at x^*. In the rest of the paper, we assume G to be symmetric, positive semi-definite and $G \preceq \sigma I$, with $\sigma < 1$. Equation (FPI) becomes

$$\tilde{x}_{i+1} = g(x^*) + G(\tilde{x}_i - x^*) + O(\|\tilde{x}_i - x^*\|^2), \quad \text{for } i = 1, \dots, k.$$

By neglecting the second order term, and because $g(x^*) = x^*$, we obtain the *linear fixed-point iteration*

$$x_{i+1} - x^* = G(x_i - x^*), \tag{LFPI}$$

where $x_0 = \tilde{x}_0$ and we recognize here a vector autoregressive process. When using the fixed-step gradient method in (2) for example, if $\nabla^2 f$ is the Hessian matrix of $f(x)$, we get

$$x_{i+1} - x^* = \underbrace{(\mathbf{I} - \nabla^2 f(x^*))}_{=G}(x_i - x^*).$$

Because $\|G\|_2 \leq \sigma < 1$, the iterates x_k converges to x^* at a linear rate, with

$$\|x_i - x^*\| \leq \sigma \|x_{i-1} - x^*\| \leq \sigma^i \|x_0 - x^*\|,$$

where $\|\cdot\|$ stands for the Euclidean norm here and throughout the paper. We will now see how to improve convergence rates using a linear combination of the previous iterates.

Suppose we run k iterations of (LFPI), a linear combination of iterates x_i with coefficients c_i reads

$$\begin{aligned}\sum_{i=0}^{k} c_i x_i &= \sum_{i=0}^{k} c_i x^* + \sum_{i=0}^{k} c_i G(x_i - x^*) \\ &= \left(\sum_{i=0}^{k} c_i\right) x^* + \left(\sum_{i=0}^{k} c_i G^i\right)(x_0 - x^*). \end{aligned} \tag{3}$$

Now define the polynomial

$$p(z) \triangleq \sum_{i=0}^{k} c_i z^i, \tag{4}$$

we can write (3) more concisely in terms of the matrix polynomial $p(G)$, setting $p(1) = \sum_{i=0}^{k} c_i = 1$ without loss of generality, to get

$$\sum_{i=0}^{k} c_i x_i = x^* + \underbrace{p(G)(x_0 - x^*)}_{\text{Error term}}.$$

Ideally, we need to find c (or equivalently p) which minimizes the error term $p(G)(x_0 - x^*)$. We will study the error when linearly combining the last $k+1$ iterates x_i, assuming we have an algorithm computing this optimal combination, i.e.

$$\begin{aligned}\left\|\sum_{i=0}^{k} c_i^\star x_i - x^*\right\| &= \min_{\{c \in \mathbb{R}^{k+1} : c^T \mathbf{1} = 1\}} \left\|\sum_{i=0}^{k} c_i G^i (x_0 - x^*)\right\| \\ &= \min_{\{p \in \mathbb{R}_k[x] : p(1) = 1\}} \left\|p(G)(x_0 - x^*)\right\| \end{aligned}$$

where $\mathbb{R}_k[x]$ is the subspace of polynomials of degree at most k and

$$c^\star = \operatorname*{argmin}_{\{c \in \mathbb{R}^{k+1} : c^T \mathbf{1} = 1\}} \left\|\sum_{i=0}^{k} c_i G^i (x_0 - x^*)\right\|.$$

The next proposition produces an uniform bound on the value of this error using Chebyshev polynomials.

Proposition 2.1 *Suppose the iterates x_i for $i = 0, \ldots, k$ are computed using (LFPI), with G the Jacobian of g, assumed to be symmetric, satisfying $0 \preceq G \preceq \sigma I$ for $\sigma < 1$. Let x^* be the fixed point of g. The ℓ_2 norm of the error is bounded, with*

$$\left\| \sum_{i=0}^{k} c_i^\star x_i - x^* \right\| \leq \begin{cases} \dfrac{2\beta^k}{1+\beta^{2k}} \|x_0 - x^*\| & \text{if } k < m \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where m is the number of distinct eigenvalues of G and

$$\beta = \frac{1-\sqrt{1-\sigma}}{1+\sqrt{1-\sigma}} < 1. \tag{6}$$

Proof Because G is symmetric, it admits the eigenvalue decomposition

$$G = Q^* \Lambda Q,$$

where Λ is the diagonal matrix of eigenvalues $\{\lambda_i,\ i = 1, \ldots, m\}$, and Q is a unitary matrix. For any $p \in \mathbb{R}_k[x]$, we have

$$\begin{aligned} \|p(G)(x_0 - x^*)\|_2 &= \|Q^* p(\Lambda) Q (x_0 - x^*)\|_2 \\ &\leq \|p(\Lambda)\|_2 \, \|(x_0 - x^*)\|_2 \\ &= \max_{i=1,\ldots,m} |p(\lambda_i)| \, \|(x_0 - x^*)\|_2. \end{aligned}$$

First, assume $k \geq m$. The Cayley–Hamilton theorem means that if $p(x)$ is the characteristic polynomial of G, then $p(G) = p(\Lambda) = 0$. By assumption none of the λ_i is equal to 1 (we assumed $\lambda_i \in [0, \sigma]$ with $\sigma < 1$) so we can normalize p so that $p(1) = 1$ and $p(\lambda_i) = 0$ for all $i = 1, \ldots, m$.

We now assume $k < m$. We have, for any polynomial p,

$$\max_{i=1,\ldots,m} |p(\lambda_i)| \leq \max_{\lambda \in [\lambda_{\min}, \lambda_{\max}]} |p(\lambda)|$$

Because $0 \preceq G \preceq \sigma$, we have $0 \leq \lambda_i \leq \sigma$, so the error bound becomes

$$\min_{\{p \in \mathbb{R}_k[x]:\, p(1)=1\}} \|p(G)(x_0 - x^*)\|_2 \leq \min_{\{p \in \mathbb{R}_k[x]:\, p(1)=1\}} \max_{\lambda \in [0,\sigma]} |p(\lambda)| \, \|x_0 - x^*\|_2 \tag{7}$$

where the right hand side involves a minmax problem, explicitly solved using Chebyshev's polynomials. Let C_k be the Chebyshev polynomial of degree k. By definition, C_k is a monic polynomial (i.e. a polynomial whose leading coefficient is one) solving

$$C_k(x) \triangleq \operatorname*{argmin}_{\{p \in \mathbb{R}_k[x]:\, p(1)=1\}} \max_{x \in [-1,1]} |p(x)|.$$

Golub and Varga [10] use a variant of $C_k(x)$ to solve the problem in (7), whose solution is a rescaled Chebyshev polynomial given by

$$T_k(x,\sigma) = \frac{C_k(t(x,\sigma))}{C_k(t(1,\sigma))}, \quad \text{where} \quad t(x,\sigma) = \frac{2x-\sigma}{\sigma}, \tag{8}$$

where $t(x,\sigma)$ is simply a linear mapping from interval $[0,\sigma]$ to $[-1,1]$. Moreover, they show

$$\min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \max_{\lambda\in[0,\sigma]} |p(\lambda)| = \max_{\lambda\in[0,\sigma]} |T_k(\lambda,\sigma)| = |T_k(\sigma,\sigma)| = \frac{2\beta^k}{1+\beta^{2k}}, \tag{9}$$

where β is given by

$$\beta = \frac{1-\sqrt{1-\sigma}}{1+\sqrt{1-\sigma}} < \sigma < 1.$$

Injecting the result of (9) in (7) yields the desired result. □

Corollary 2.2 *In the case of the gradient method applied on quadratic function with eigenvalues bounded in the interval* $[\mu, L]$ *(this correspond also to a* L*-smooth and* μ*-strongly convex function), we have*

$$\sigma = 1 - \frac{\mu}{L} < 1.$$

By consequence, the bound (5) *becomes*

$$\left\| \sum_{i=0}^{k} c_i^\star x_i - x^* \right\| \leq \begin{cases} \dfrac{2\beta^k}{1+\beta^{2k}} \|x_0 - x^*\| & \text{if } k < m \\ 0 & \text{otherwise} \end{cases}$$

with

$$\beta = \frac{1-\sqrt{\mu/L}}{1+\sqrt{\mu/L}}.$$

The proof of this proposition suggests $T_k(x,\sigma)$ is a good universal solution for the convergence acceleration problem and we plot both $T_3(x,\sigma)$ and $T_5(x,\sigma)$ for $x \in [0,1]$ and $\sigma = 0.85$ in Fig. 1. This solution is called the *Chebyshev semi-iterative method* in Golub and Varga, [10] and was further studied by e.g. [20]. Combining the k iterates x_i using the coefficients c_i in $T_k(x,\sigma)$, we ensure

$$\left\| \sum_{i=0}^{k} c_i x_i - x^* \right\| \lesssim (1-\sqrt{1-\sigma})^k \|x_0 - x^*\| \ll \sigma^k \|x_0 - x^*\|$$

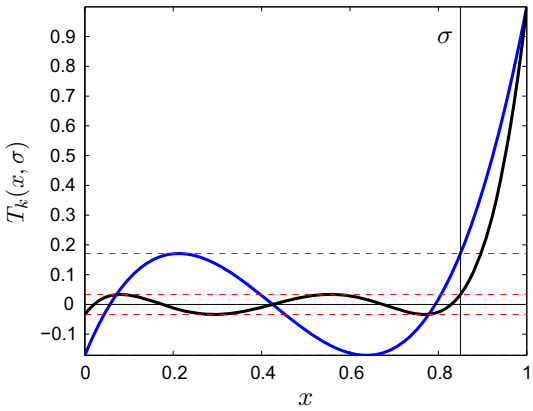

Fig. 1 We plot both $T_3(x,\sigma)$ (blue) and $T_5(x,\sigma)$ (black) for $x \in [0,1]$ and $\sigma = 0.85$. The maximum value of the image of $[0,\sigma]$ by T_k is clearly smaller when k grows, implying a better rate of convergence

which means convergence is indeed accelerated. However, this method has some key drawbacks. First, we need to know σ to form $T_k(x,\sigma)$, which is not always the case. For example, in the case of the gradient method, σ depends on the smoothness constant L and the strong convexity constant μ. In the general non-linear case, σ depends on the spectrum of the Jacobian at the optimum, which is clearly not observed. Second, the algorithm does not allow us to control the magnitude of the coefficients in the polynomial $T(x,\sigma)$, which has a strong impact on the stability of the algorithm in the presence of numerical errors, or when the iterates are generated by a non-linear function g.

Because of stability issues with Chebyshev acceleration, we focus now on a method which will approximately minimize the error $\|p(G)(x_0 - x^*)\|_2$. Since we of course do not observe G and x^* we will work with the residuals

$$\tilde{r}_i = \tilde{x}_{i+1} - \tilde{x}_i = g(\tilde{x}_i) - \tilde{x}_i, \tag{10}$$

when g is a linear function (LFPI) this becomes

$$r_i = x_{i+1} - x_i = (G - I)(x_i - x^*). \tag{11}$$

A linear combination of residuals r_i with coefficients c_i is written

$$\sum_{i=0}^{k} c_i r_i = (G - I) \sum_{i=0}^{k} c_i (x_i - x^*) = (G - I) p(G)(x_0 - x^*).$$

We recognize the error term we wanted to minimize, multiplied by the matrix $(G - I)$. Using the coefficients which minimize this alternative quantity will approximately minimize the error, as stated in the following proposition.

Proposition 2.3 *Let $p^*(x)$ be the polynomial solving*

$$p^*(x) = \underset{\{p \in \mathbb{R}_k[x]:\, p(1)=1\}}{\operatorname{argmin}} \|(G - I) p(G)(x_0 - x^*)\|_2,$$

whose coefficients, written c^*, satisfy

$$c^* = \underset{\{c\in\mathbb{R}^{k+1}:\, c^T\mathbf{1}=1\}}{\operatorname{argmin}} \left\| \sum_{i=0}^{k} c_i r_i \right\|_2. \tag{12}$$

The iterates x_i defined in (LFPI) averaged with coefficients c^* satisfy

$$\left\| \sum_{i=0}^{k} c_i^* x_i - x^* \right\| \leq \frac{1}{1-\sigma} \min_{\{c\in\mathbb{R}^{k+1}:\, c^T\mathbf{1}=1\}} \left\| \sum_{i=0}^{k} c_i G^i (x_0 - x^*) \right\|, \tag{13}$$

where we have assumed $0 \preceq G \preceq \sigma I$, with $\sigma < 1$.

Proof By definition of c^*, and using (LFPI),

$$\begin{aligned} \left\| \sum_{i=0}^{k} c_i^* x_i - x^* \right\| &= \|p^*(G)(x_0 - x^*)\|, \\ &= \|(G-I)^{-1}(G-I)p^*(G)(x_0 - x^*)\|, \\ &\leq \|(G-I)^{-1}\|\, \|(G-I)p^*(G)(x_0 - x^*)\|. \end{aligned}$$

By using the definition of p^*,

$$\|(G-I)p^*(G)(x_0 - x^*)\| = \min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \|(G-I)p(G)(x_0 - x^*)\|_2.$$

We can bound this last error term because $\|G - I\| \leq 1$,

$$\begin{aligned} \min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \|(G-I)p(G)(x_0 - x^*)\| &\leq \min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \|(G-I)\| \|p(G)(x_0 - x^*)\|, \\ &\leq \min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \|p(G)(x_0 - x^*)\|. \end{aligned}$$

Using the fact that $\|(G-I)^{-1}\| \leq \frac{1}{1-\sigma}$ yields the desired result. □

This leads to the following acceleration algorithm.

Algorithm 1 Nonlinear Acceleration of Convergence

Input: Iterates $x_0, x_1, \ldots, x_{k+1} \in \mathbb{R}^d$.
1: Form $R = [r_0, ..., r_k]$
2: Solve

$$c^* = \underset{\{c\in\mathbb{R}^{k+1}:\, c^T 1=1\}}{\operatorname{argmin}} \|Rc\|$$

Output: Approximation of x^* ensuring (13), computed as $\sum_{i=0}^{k} c_i^* x_i$

This acceleration algorithm is called nonlinear because the coefficients c_i vary with of x_i. This method is also known as Anderson acceleration [2], the Eddy–Mesina algorithm [9,18], Minimal Polynomial Extrapolation [6], or Reduced Rank Extrapolation [27,28]. There are small variations between all these methods, which lie in the way they solve the minimization problem in (12). The next proposition gives us an explicit solution, involving the inversion of a k-by-k matrix.

Proposition 2.4 *The explicit solution of the problem*

$$c^* = \operatorname*{argmin}_{c^T 1=1} \|Rc\| \tag{14}$$

in the variable $c \in \mathbb{R}^k$, where R is a $d \times k$ matrix assumed to be of rank k, is given by

$$c^* = \frac{(R^T R)^{-1}\mathbf{1}}{\mathbf{1}^T (R^T R)^{-1}\mathbf{1}}. \tag{15}$$

Proof Let μ be the dual variable of the equality constraint. Both c^* and μ^* should satisfy the KKT system

$$\begin{bmatrix} 2R^T R & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} c^* \\ \mu^* \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{16}$$

This block matrix can be inverted explicitly, with

$$\begin{bmatrix} 2R^T R & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix}^{-1} = \frac{1}{\mathbf{1}^T (R^T R)^{-1}\mathbf{1}} \begin{bmatrix} \frac{1}{2}(R^T R)^{-1}\left(\mathbf{1}^T (R^T R)^{-1}\mathbf{1} I - \mathbf{1}\mathbf{1}^T (R^T R)^{-1}\right) & (R^T R)^{-1}\mathbf{1} \\ \mathbf{1}^T (R^T R)^{-1} & -2 \end{bmatrix}.$$

Using this inverse we easily solve the linear system, which gives the result in (15). □

In practice of course, instead of computing the inverse of the matrix $R^T R$, we solve the linear system

$$R^T R z = \mathbf{1},$$

then set $c^* = z/(\mathbf{1}^T z)$. This formula is used in Anderson Acceleration algorithm and Mesina method. Other algorithms usually force the coefficient c_k to be equal to one, solve the remaining linear system, then normalize the vector. However, these alternative strategies are harder to analyze when the iterates are generated by a nonlinear function g. We will now apply this acceleration algorithm on gradient method for nonlinear functions and compute its rate of convergence.

3 Regularized nonlinear acceleration of convergence

So far, we have only considered linear functions g in (LFPI), without perturbations, when computing the iterates x_i. In general, the fixed-point iteration (FPI) is usually

generated by a nonlinear function g, thus inducing a second order error term in $O(\|x_i - x^*\|^2)$ compared to the dynamics in (LFPI).

Here, in Sect. 3.1 we first give a bound on the deviation error when there are perturbations in (LFPI). In Sect. 3.2 we then derive a regularized version of Algorithm 1 which better controls the impact of perturbations. We then study the impact of regularization on the solution when there are no perturbations in Sect. 3.3. Finally, in Sect. 3.4 we gather the results of the previous sections to bound the rate of convergence of the regularized acceleration algorithm.

3.1 Sensitivity Analysis

We now study the sensitivity of the acceleration algorithm to perturbations. Consider the following perturbed linear fixed point iteration

$$\tilde{x}_{i+1} - x^* = g(\tilde{x}_i) - x^* = G(\tilde{x}_i - x^*) + e_i \qquad \text{(Pert. LFPI)}$$

where e_i is the noise injected in x_{i+1} at iteration i. For now, we do not assume any structure on the noise, so e_i may be the nonlinearity of g, stochastic noise, roundoff error, etc. The iterates of this process will be compared to their noiseless counterpart,

$$x_{i+1} - x^* = G(x_i - x^*),$$

with $x_0 = \tilde{x}_0$. We now apply our acceleration algorithm on the sequences x_i and $\tilde{x}_i$ and compare the results. We first form the residuals,

$$r_i = g(x_i) - x_i = x_{i+1} - x_i \qquad \text{and} \qquad \tilde{r}_i = g(\tilde{x}_i) - \tilde{x}_i = \tilde{x}_{i+1} - \tilde{x}_i.$$

Consider the matrices of residuals $R = [r_0, \ldots, r_k]$ and $\tilde{R} = [\tilde{r}_0, \ldots, \tilde{r}_k]$. We write P the *perturbation matrix* defined as

$$P \triangleq \tilde{R}^T \tilde{R} - R^T R. \tag{17}$$

The next proposition describes the sensitivity of Algorithm 1 using R and P.

Proposition 3.1 *Let the sequences x_i be generated by (LFPI) and $\tilde{x}_i$ by (Pert. LFPI), with $x_0 = \tilde{x}_0$, with R and $\tilde{R}$ the residual matrices defined above and P the perturbation matrix in (17). Assume c^* and $\tilde{c}^*$ are computed using formula (15) with matrices R and $\tilde{R}$ respectively. Let*

$$\Delta\tilde{c}^* \triangleq \tilde{c}^* - c^*. \tag{18}$$

Then the norm of $\Delta\tilde{c}^*$ is bounded by

$$\|\Delta\tilde{c}^*\| \leq \|P\|\|(R^T R + P)^{-1}\|\|c^*\|. \tag{19}$$

Proof We start with the sequence $\tilde{x}_i$. Let $\tilde{\mu}^*$ be the dual variable of the equality constraint of (14). Both $\tilde{c}^* = c^* + \Delta\tilde{c}^*$ and $\tilde{\mu}^* = \mu^* + \Delta\mu^*$ should satisfy the KKT system

$$\begin{bmatrix} 2\tilde{R}^T\tilde{R} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} \tilde{c}^* \\ \tilde{\mu}^* \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \Leftrightarrow \quad \begin{bmatrix} 2(R^TR + P) & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} c^* + \Delta\tilde{c}^* \\ \mu^* + \Delta\mu^* \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Indeed, using the definition of c^* and μ^* in (16),

$$\begin{aligned} \begin{bmatrix} 2(R^TR + P) & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} c^* + \Delta\tilde{c}^* \\ \mu^* + \Delta\mu^* \end{pmatrix} &= \begin{bmatrix} 2R^TR & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} c^* \\ \mu^* \end{pmatrix} + \begin{bmatrix} 2R^TR & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} \Delta\tilde{c}^* \\ \Delta\mu^* \end{pmatrix} \\ &\quad + \begin{bmatrix} 2P & 0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} c^* + \Delta\tilde{c}^* \\ \mu^* + \Delta\mu^* \end{pmatrix}, \\ &= \begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{bmatrix} 2R^TR & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} \Delta\tilde{c}^* \\ \Delta\mu^* \end{pmatrix} + \begin{bmatrix} 2P & 0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} c^* + \Delta\tilde{c}^* \\ \mu^* + \Delta\mu^* \end{pmatrix}. \end{aligned}$$

With this simplification, the system becomes

$$\begin{bmatrix} 2R^TR & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} \Delta\tilde{c}^* \\ \Delta\mu^* \end{pmatrix} + \begin{bmatrix} 2P & 0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} c^* + \Delta\tilde{c}^* \\ \mu^* + \Delta\mu^* \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

It remains to isolate c^*,

$$\begin{bmatrix} 2(R^TR + P) & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{pmatrix} \Delta\tilde{c}^* \\ \Delta\mu^* \end{pmatrix} = \begin{pmatrix} 2Pc^* \\ 0 \end{pmatrix}.$$

The explicit solution in obtained by inverting the block matrix, and is written

$$\Delta\tilde{c}^* = \left(I - \frac{(R^TR + P)^{-1}\mathbf{1}\mathbf{1}^T}{\mathbf{1}^T(R^TR + P)^{-1}\mathbf{1}} \right) (R^TR + P)^{-1} Pc^*.$$

We can bound the norm of $\Delta\tilde{c}^*$ by

$$\|\Delta\tilde{c}^*\| = \left\| I - \frac{(R^TR + P)^{-1}\mathbf{1}\mathbf{1}^T}{\mathbf{1}^T(R^TR + P)^{-1}\mathbf{1}} \right\| \|(R^TR + P)^{-1}\| \|P\| \|c^*\|.$$

Because the first factor is the norm of a projector of rank $k - 1$, its value is bounded by 1, so we get the desired result. □

This proposition bounds the relative error on $\tilde{c}^*$ in comparison with c^*. We will see that the perturbation magnitude can be arbitrarily large, which is the key issue with the convergence results in Smith et al. [28, §7]. Even when $\|P\|$ is small, the term $\|(R^TR + P)^{-1}\|$ is problematic. Our problem here is the structure of the residuals matrix R,

$$R = [r_0, Gr_0, G^2r_0, ..., G^kr_0],$$

which matches exactly the structure of *Krylov matrices*, i.e. square matrices K formed using a matrix M and a vector v, and computed as $K = [v, Mv, M^2v, ..., M^kv]$. Tyrtyshnikov [30] showed that the condition number of Krylov matrices (see Sect. 4.3) is lower bounded by a function which grows exponentially with k. Now, the error bound (19) contains the norm of the inverse of a perturbed squared Krylov matrix, which makes the situation even worse. In other words, even if the perturbations are small, their impact on the solution can be arbitrarily large. Even in practical cases where k is small (for example, $k = 5$), $\tilde{R}^T\tilde{R}$ is usually a singular or nearly-singular matrix. This particular issue means the linear system $(R^TR)^{-1}\mathbf{1}$ in (15) needs to be regularized.

3.2 Regularized nonlinear acceleration of convergence

In this section, we will analyze the following acceleration algorithm, which uses Tikhonov regularization to solve the linear system in (15).

Algorithm 2 Regularized Nonlinear Acceleration (RNA)

Input: Iterates $\tilde{x}_0, \tilde{x}_1, ..., \tilde{x}_{k+1} \in \mathbb{R}^d$ produced by (FPI), and a regularization parameter $\lambda > 0$.
1: Compute $\tilde{R} = [\tilde{r}_0, ..., \tilde{r}_k]$, where $\tilde{r}_i = \tilde{x}_{i+1} - \tilde{x}_i$
2: Solve

$$\tilde{c}^*_\lambda = \underset{c^T\mathbf{1}=1}{\operatorname{argmin}} \|\tilde{R}c\|^2 + \lambda\|c\|^2,$$

or equivalently solve $(\tilde{R}^T\tilde{R} + \lambda I)z = \mathbf{1}$ then set $\tilde{c}^*_\lambda = z/\mathbf{1}^Tz$.
Output: Approximation of x^* computed as $\sum_{i=0}^k (\tilde{c}^*_\lambda)_i \tilde{x}_i$

Regularization controls the norm of the coefficients produced by the algorithm and reduces the impact of perturbations, as shown in the following proposition.

Proposition 3.2 *Consider the sequences x_i satisfying* (LFPI) *and $\tilde{x}_i$ satisfying* (Pert. LFPI) *with $x_0 = \tilde{x}_0$. Let c^*_λ and $\tilde{c}^*_\lambda$ the output of Algorithm 2 with parameter λ applied to x_i and $\tilde{x}_i$ respectively. Let R and $\tilde{R}$ the matrices of residuals and P be defined in* (17). *Define $\Delta\tilde{c}^*_\lambda = \tilde{c}^*_\lambda - c^*_\lambda$. Then, we have the following bounds,*

$$\|\tilde{c}^*_\lambda\| \leq \sqrt{\frac{\lambda + \|\tilde{R}\|^2}{(k+1)\lambda}}, \tag{20}$$

$$\|\Delta\tilde{c}^*_\lambda\| \leq \frac{\|P\|}{\lambda}\|c^*_\lambda\|, \tag{21}$$

which control the stability of the solution $\tilde{c}^*_\lambda$.

Proof Using the same proof technique of Propositions 2.4 and 3.1 , we have

$$\tilde{c}^*_\lambda = \frac{(\tilde{R}^T\tilde{R} + \lambda I)^{-1}\mathbf{1}}{\mathbf{1}^T(\tilde{R}^T\tilde{R} + \lambda I)^{-1}\mathbf{1}}, \tag{22}$$

$$\Delta\tilde{c}_\lambda^* = \left(I - \frac{(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\mathbf{1}\mathbf{1}^T}{\mathbf{1}^T(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\mathbf{1}}\right)(\tilde{R}^T\tilde{R}+\lambda I)^{-1}Pc_\lambda^*. \tag{23}$$

We begin by the bound on $\tilde{c}_\lambda^*$. Indeed, with (22),

$$\begin{aligned}
\|\tilde{c}_\lambda^*\|^2 &= \frac{\mathbf{1}^T(\tilde{R}^T\tilde{R}+\lambda I)^{-2}\mathbf{1}}{(\mathbf{1}^T(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\mathbf{1})^2}, \\
&\leq \frac{1}{k+1}\max_{\|v\|=1}\frac{v^T(\tilde{R}^T\tilde{R}+\lambda I)^{-2}v}{(v^T(\tilde{R}^T\tilde{R}+\lambda I)^{-1}v)^2}, \\
&= \frac{1}{k+1}\max_{\|v\|=1}\frac{\|(\tilde{R}^T\tilde{R}+\lambda I)^{-\frac{1}{2}}(\tilde{R}^T\tilde{R}+\lambda I)^{-\frac{1}{2}}v\|^2}{\|(\tilde{R}^T\tilde{R}+\lambda I)^{-\frac{1}{2}}v\|^4}, \\
&\leq \frac{1}{k+1}\|(\tilde{R}^T\tilde{R}+\lambda I)^{-\frac{1}{2}}\|^2\max_{\|v\|=1}\frac{1}{\|(\tilde{R}^T\tilde{R}+\lambda I)^{-\frac{1}{2}}v\|^2}, \\
&= \frac{1}{k+1}\|(\tilde{R}^T\tilde{R}+\lambda I)^{-\frac{1}{2}}\|^2\|(\tilde{R}^T\tilde{R}+\lambda I)^{\frac{1}{2}}\|^2.
\end{aligned}$$

The norm of the coefficients $\tilde{c}_\lambda^*$ are thus bounded by

$$\|\tilde{c}_\lambda^*\| \leq \sqrt{\frac{1}{k+1}\frac{\|\tilde{R}^T\tilde{R}\|+\lambda}{\lambda}} = \sqrt{\frac{\|\tilde{R}\|^2+\lambda}{(k+1)\lambda}}.$$

We will now bound $\|\Delta\tilde{c}_\lambda\|$. With equation (23),

$$\begin{aligned}
\|\Delta\tilde{c}_\lambda\| &= \left\|\left(I - \frac{(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\mathbf{1}\mathbf{1}^T}{\mathbf{1}^T(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\mathbf{1}}\right)(\tilde{R}^T\tilde{R}+\lambda I)^{-1}Pc_\lambda^*\right\|, \\
&\leq \left\|I - \frac{(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\mathbf{1}\mathbf{1}^T}{\mathbf{1}^T(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\mathbf{1}}\right\|\left\|(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\right\|\|P\|\left\|c_\lambda^*\right\|, \\
&\leq \left\|(\tilde{R}^T\tilde{R}+\lambda I)^{-1}\right\|\|P\|\left\|c_\lambda^*\right\|,
\end{aligned}$$

where the last inequality is obtained by bounding the norm of a projector. Since $\tilde{R}^T\tilde{R} \succeq 0$, we have $(\tilde{R}^T\tilde{R}+\lambda I) \succeq \lambda I$, we get

$$\|\Delta\tilde{c}_\lambda\| \leq \frac{\|P\|}{\lambda}\|c_\lambda^*\|,$$

which is the desired result. □

Regularization allows a better control of the impact of perturbations, but also changes the solution c^* into c_λ^*. The next section analyses the impact of regularization on the extrapolated solution when there are no perturbations.

3.3 Regularized Chebyshev polynomial

The previous section shows that regularization is important for the control of the perturbations present in (Pert. LFPI). However, the convergence analysis becomes more complicated in the perturbation-free case, and we introduce *regularized Chebyshev polynomials.*

Definition 3.3 The regularized Chebyshev polynomial $C_\sigma^*(x, k, \alpha)$ of degree k, range σ and regularization parameter α is defined as the solution of

$$C_\sigma^*(x, k, \alpha) = \underset{C \in \mathbb{R}_k[x] \,:\, C(1)=1}{\operatorname{argmin}} \max_{x \in [0,\sigma]} C^2(x) + \alpha \|C\|^2,$$

where $\|C\|$ corresponds to the ℓ_2 norm of the coefficients of polynomial C. We write the maximum value as

$$S_\sigma(k, \alpha) \triangleq \sqrt{\max_{x \in [0,\sigma]} (C_\sigma^*(x, k, \alpha))^2 + \alpha \|C_\sigma^*(x, k, \alpha)\|^2}. \tag{24}$$

Using this specific polynomial we can now bound the accuracy of the extrapolated point using the regularized algorithm.

Proposition 3.4 *Let c_λ^* be the output of Algorithm 2 using the sequence x_i generated by (LFPI) (with $\|G\| \leq \sigma < 1$) and the parameter $\lambda > 0$. The accuracy of the extrapolation is bounded by*

$$\left\| \sum_{i=0}^{k} (c_\lambda^*)_i x_i - x^* \right\| \leq \|(G - I)^{-1}\| \sqrt{S_\sigma^2 \left(k, \frac{\lambda}{\|x_0 - x^*\|^2} \right) \|x_0 - x^*\|^2 - \lambda \|c_\lambda^*\|^2}. \tag{25}$$

Proof Consider the optimization problem in Algorithm 2,

$$\min_{c^T \mathbf{1} = 1} \|Rc\|^2 + \lambda \|c\|^2.$$

Since $r_i = (G - I)(x_i - x^*) = (G - I)G^i(x_0 - x^*)$, if we use the polynomial p with coefficients c, the problem becomes

$$\begin{aligned}
&\min_{\{p \in \mathbb{R}_k[x] :\, p(1)=1\}} \left\{ \|(G - I)p(G)(x_0 - x^*)\|^2 + \lambda \|p\|^2 \right\}, \\
&\quad \leq \|x_0 - x^*\|^2 \min_{\{p \in \mathbb{R}_k[x] :\, p(1)=1\}} \left\{ \|(G - I)p(G)\|^2 + \frac{\lambda}{\|x_0 - x^*\|^2} \|p\|^2 \right\}, \\
&\quad \leq \|x_0 - x^*\|^2 \min_{\{p \in \mathbb{R}_k[x] :\, p(1)=1\}} \left\{ \|G - I\|^2 \|p(G)\|^2 + \frac{\lambda}{\|x_0 - x^*\|^2} \|p\|^2 \right\}, \\
&\quad \leq \|x_0 - x^*\|^2 \min_{\{p \in \mathbb{R}_k[x] :\, p(1)=1\}} \left\{ \|p(G)\|^2 + \frac{\lambda}{\|x_0 - x^*\|^2} \|p\|^2 \right\},
\end{aligned} \tag{26}$$

where $\|p\|$ is the ℓ_2 norm of the coefficients of p. For simplicity, we write $\bar{\lambda} = \lambda/\|x_0 - x^*\|$ to be the normalized value of λ. In the optimization problem, since $\|G\| \leq \sigma$, we can consider the worst-case over all symmetric matrices M with $\|M\| \leq \|G\|$ and $M \succeq 0$, written

$$\begin{aligned}&\min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \left\{\|p(G)\|^2 + \bar{\lambda}\|p\|^2\right\}\\ &\quad\leq \min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \max_{M\succeq 0,\ \|M\|\leq\sigma} \left\{\|p(M)\|^2 + \bar{\lambda}\|p\|^2\right\}.\end{aligned}$$

Because M is symmetric, we only need to look at its eigenvalues which are inside the segment $[0, \sigma]$,

$$\begin{aligned}&\min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \max_{M\succeq 0,\ \|M\|\leq\sigma} \left\{\|p(M)\|^2 + \bar{\lambda}\|p\|^2\right\}\\ &\quad= \min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \max_{x\in[0,\sigma]} \left\{p^2(x) + \bar{\lambda}\|p\|^2\right\},\\ &\quad= S_\sigma^2(k, \bar{\lambda}).\end{aligned}$$

This means that (26) is bounded by

$$\min_{\{p\in\mathbb{R}_k[x]:\, p(1)=1\}} \left\{\|(G-I)p(G)(x_0 - x^*)\|^2 + \lambda\|p\|^2\right\} \leq \|x_0 - x^*\|^2 S_\sigma^2(k, \bar{\lambda}). \tag{27}$$

It remains to link the optimization problem to the accuracy of the extrapolation. Indeed,

$$\begin{aligned}\left\|\sum_{i=0}^k (c_\lambda^*)_i x_i - x^*\right\|^2 &= \left\|(G-I)^{-1}\sum_{i=0}^k (c_\lambda^*)_i r_i\right\|^2,\\ &\leq \left\|(G-I)^{-1}\right\|^2 \left\|\sum_{i=0}^k (c_\lambda^*)_i r_i\right\|^2,\\ &= \left\|(G-I)^{-1}\right\|^2 \left(\left\|\sum_{i=0}^k (c_\lambda^*)_i r_i\right\|^2 + (\lambda-\lambda)\|c_\lambda^*\|^2\right).\end{aligned}$$

By definition, of c_λ^*,

$$\left\|\sum_{i=0}^k (c_\lambda^*)_i r_i\right\|^2 + \lambda\|c_\lambda^*\|^2 = \min_{p\in\mathbb{R}_k[x]\,:\, p(1)=1} \left\{\|(G-I)p(G)(x_0 - x^*)\|^2 + \lambda\|p\|^2\right\}.$$

We proved in (27) that this quantity can be bounded by $S_\sigma^2(k,\bar{\lambda})\|x_0 - x^*\|^2$, so we finally have

$$\left\| \sum_{i=0}^{k} (c_\lambda^*)_i x_i - x^* \right\| \leq \|(G-I)^{-1}\| \sqrt{S_\sigma^2(k,\bar{\lambda})\|x_0 - x^*\|^2 - \lambda \|c_\lambda^*\|^2},$$

which is the desired result. □

Regularized Chebyshev polynomials are crucial for the bound on the accuracy of Algorithm 2. Unfortunately, there is no explicit expressions for $S_\sigma(k,\alpha)$ in (24). However, this value can be computed numerically using sum-of-squares optimization. We show in Fig. 2 the difference of performances when using the coefficients of the regularized Chebyshev polynomial instead of its non-regularized version.

We briefly recall basic results on Sum of Squares (SOS) polynomials and moment problems [14,22,25], which will allow us to formulate problem (24) as a (tractable) semidefinite program. A univariate polynomial is positive if and only if it is a sum of squares. Furthermore, if we let $m(x) = (1, x, \ldots, x^k)^T$ we have, for any $q(x) \in \mathbb{R}_{[2k]}$,

$$\begin{array}{c} q(x) \geq 0, \text{ for all } x \in \mathbb{R} \\ \Updownarrow \\ q(x) = m(x)^T C m(x), \text{ for some } C \succeq 0, \end{array}$$

which means that checking if a polynomial is non-negative on the real line is equivalent to solving a linear matrix inequality (see e.g. [4, §4.2] for details). We can thus write the problem of computing the maximum of a polynomial over the real line as

$$\begin{array}{ll} \text{minimize} & t \\ \text{subject to} & t - p(x) = m(x)^T C m(x), \quad \text{for all } x \in \mathbb{R} \\ & C \succeq 0, \end{array} \tag{28}$$

which is a semidefinite program in the variables $p \in \mathbb{R}^{k+1}$, $C \in \mathbf{S}_{k+1}$ and $t \in \mathbb{R}$, because the first contraint is equivalent to a set of linear equality constraints. Then, showing that $p(x) \geq 0$ on the segment $[0, \sigma]$ is equivalent to showing that the rational fraction

$$p\left(\frac{\sigma x^2}{1+x^2}\right)$$

is non-negative on the real line, or equivalently, that the polynomial

$$(1+x^2)^k \, p\left(\frac{\sigma x^2}{1+x^2}\right)$$

is non-negative on the real line. Overall, this implies that problem (24) can be written

$$\begin{array}{lll} S_\sigma(k,\alpha) = & \min. & t^2 + \alpha^2 \|q\|_2^2 \\ & \text{s.t.} & t - (1+x^2)^{k+1} \left(\left(1 - \frac{\sigma x^2}{1+x^2}\right) q\left(\frac{\sigma x^2}{1+x^2}\right)\right) = m(x)^T C m(x), \quad \text{for all } x \in \mathbb{R} \\ & & \mathbf{1}^T q = 1, \; C \succeq 0, \end{array} \tag{29}$$

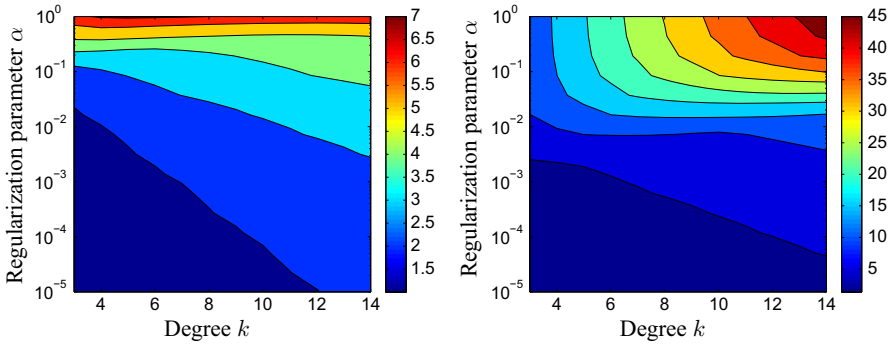

Fig. 2 Ratio between the (worst-case) number of iterations required to reach an arbitrary accuracy using the coefficients of the regularized and non-regularized Chebyshev polynomial for combining the x_i. On the left, $\sigma = 0.9$ and on the right $\sigma = 0.999$. We see that the impact of the regularization is more important when k is big, or σ close to 1

which is a semidefinite program in the variables $q \in \mathbb{R}^{k+1}$, $C \in \mathbf{S}_{k+2}$ and $t \in \mathbb{R}$.

3.4 Convergence rate

We will now prove global accuracy bounds, using the following decomposition of the error term,

$$\sum_{i=0}^{k}(\tilde{c}_\lambda^*)_i\tilde{x}_i - x^* = \underbrace{\sum_{i=0}^{k}(c_\lambda^*)_i x_i - x^*}_{\textbf{Linear case}} + \underbrace{\sum_{i=0}^{k}(\Delta\tilde{c}_\lambda^*)_i x_i}_{\textbf{Stability}} + \underbrace{\sum_{i=0}^{k}(\tilde{c}_\lambda^*)_i(\tilde{x}_i - x_i)}_{\textbf{Nonlinearity}}, \quad (30)$$

where $\Delta\tilde{c}_\lambda^* = \tilde{c}_\lambda^* - c_\lambda^*$. In the equation above, the first term is the accuracy of the accelerated method in the noiseless case. The second term corresponds the stability of the coefficients computed by the regularized algorithm when we have some perturbations in the sequence. The last term is the induced error by the combination of the perturbations. The following Theorem shows how to bound these three terms by putting together the results of Propositions 3.4 and 3.2 .

Theorem 3.5 *Let $\bar{x}$ be an arbitrary point in $\mathbb{R}^n$. Given iterates $\tilde{x}_i$, $i = 0, \ldots, k+1$ generated by* (Pert. LFPI)*, Algorithm* (2) *outputs $x_{extr} = \sum_{i=0}^{k}(\tilde{c}_\lambda^*)_i\tilde{x}_i$. Consider the matrices $\bar{X}$ and $\mathcal{E}$, with columns $\bar{X}_i = x_i - \bar{x}$ and $\mathcal{E}_i = \tilde{x}_i - x_i$ respectively. We have the following bound on the extrapolated point,*

$$\|x_{extr} - x^*\| \leq \|x_0 - x^*\| S_\sigma(k, \bar{\lambda})\sqrt{\kappa^2 + \frac{\|\bar{X}\|^2\|P\|^2}{\lambda^3}} + \frac{\|\mathcal{E}\|}{\sqrt{k+1}}\sqrt{1 + \frac{\|\tilde{R}\|^2}{\lambda}}.$$

where $\kappa > 1$ with $\|(G - I)^{-1}\| \leq \frac{1}{1-\sigma} = \kappa$.

Proof The proof is divided into four parts, where the three first parts bound each term of (30) and the last one combines everything. The bound on the first term comes explicitly from Proposition 3.4, with

$$\left\| \sum_{i=0}^{k} (c_\lambda^*)_i x_i - x^* \right\| \leq \kappa \sqrt{S_\sigma^2(k, \bar{\lambda}) \|x_0 - x^*\|^2 - \lambda \|c_\lambda^*\|^2}, \tag{31}$$

where $\bar{\lambda} = \lambda / \|x_0 - x^*\|^2$. The second term can be bounded using the fact that both c_λ^* and $\tilde{c}_\lambda^*$ sum to one, so $\Delta\tilde{c}_\lambda^*$ sum to zero. In this case,

$$\begin{aligned} \left\| \sum_{i=0}^{k} (\Delta\tilde{c}_\lambda^*)_i x_i \right\| &= \left\| \sum_{i=0}^{k} (\Delta\tilde{c}_\lambda^*)_i (x_i - \bar{x}) \right\|, \\ &\leq \|\Delta\tilde{c}_\lambda^*\| \|\bar{X}\|. \end{aligned}$$

Proposition 3.2 bounds the value of $\|\Delta\tilde{c}_\lambda^*\|$ and yields

$$\left\| \sum_{i=0}^{k} (\Delta\tilde{c}_\lambda^*)_i x_i \right\| \leq \|X\| \frac{\|P\|}{\lambda} \|c_\lambda^*\|. \tag{32}$$

For the third term in (30), we have

$$\left\| \sum_{i=0}^{k} (\tilde{c}_\lambda^*)_i (\tilde{x}_i - x_i) \right\| \leq \|\tilde{c}_\lambda^*\| \|\mathcal{E}\|.$$

The norm $\|\tilde{c}_\lambda^*\|$ can be bounded using Proposition 3.2, with

$$\left\| \sum_{i=0}^{k} (\tilde{c}_\lambda^*)_i (\tilde{x}_i - x_i) \right\| \leq \frac{\|\mathcal{E}\|}{\sqrt{k+1}} \sqrt{1 + \frac{\|\tilde{R}\|^2}{\lambda}}. \tag{33}$$

We finally combine the bounds (31), (32) and (33) according to the decomposition (30), to get

$$\begin{aligned} &\left\| \sum_{i=0}^{k} (\tilde{c}_\lambda^*)_i x_i - x^* \right\| \\ &\leq \kappa \sqrt{S_\sigma^2(k, \bar{\lambda}) \|x_0 - x^*\|^2 - \lambda \|c_\lambda^*\|^2} + \|\bar{X}\| \frac{\|P\|}{\lambda} \|c_\lambda^*\| \\ &\quad + \frac{\|\mathcal{E}\|}{\sqrt{k+1}} \sqrt{1 + \frac{\|\tilde{R}\|^2}{\lambda}}. \end{aligned} \tag{34}$$

Here, $\|c_\lambda^*\|$ appears twice in the expression. We remove it by maximizing the bound over $\|c_\lambda^*\|$. The first two terms of (34) can be written

$$x \mapsto \sqrt{a - \lambda x^2} + bx.$$

with $a = S_\sigma^2(k, \bar{\lambda})\|x_0 - x^*\|^2$ and $b = \|\bar{X}\|\frac{\|P\|}{\lambda}$. By Proposition A.1 (in the "Appendix"), its maximum value is equal to

$$\sqrt{a}\sqrt{\kappa^2 + \frac{b^2}{\lambda}},$$

which is

$$S_\sigma(k, \bar{\lambda})\|x_0 - x^*\|\sqrt{\kappa^2 + \frac{\|\bar{X}\|^2\|P\|^2}{\lambda^3}}.$$

The bound on extrapolation accuracy in (34) now becomes

$$\left\|\sum_{i=0}^{k}(\tilde{c}_\lambda^*)_i x_i - x^*\right\| \leq S_\sigma(k, \bar{\lambda})\|x_0 - x^*\|\sqrt{\kappa^2 + \frac{\|\bar{X}\|^2\|P\|^2}{\lambda^3}} + \frac{\|\mathcal{E}\|}{\sqrt{k+1}}\sqrt{1 + \frac{\|\tilde{R}\|^2}{\lambda}}.$$

which is the desired result. □

We can further simplify the bound above by bounding $\|P\|$ using σ, $\mathcal{E}$ and $\|\bar{X}\|$.

Proposition 3.6 *Let P the perturbation matrix defined in (17). Then*

$$\|P\| \leq 4(\|\mathcal{E}\|\|R\| + \|\mathcal{E}\|^2),$$
$$\|R\| \leq \frac{1 - \sigma^{k+1}}{1 - \sigma}\|x_0 - x^*\|,$$

where $\mathcal{E}$ is defined in Theorem 3.5, R is the matrix of residuals for the sequence x_i generated by (LFPI), for $\|G\| \leq \sigma$.

Proof We begin by the bound on R,

$$\|R\| \leq \sum_{i=0}^{k}\|r_i\| \leq \sum_{i=0}^{k}\|G\|^i\|r_0\| \leq \sum_{i=0}^{k}\sigma^i\|r_0\|.$$

Since $r_0 = (G - I)(x_0 - x^*)$, and $\|G - I\| \leq 1$, we have $\|r_0\| \leq \|x_0 - x^*\|$. Injecting this result in the previous bound gives the desired result,

$$\|R\| \leq \sum_{i=0}^{k}\sigma^i\|x_0 - x^*\| = \frac{1 - \sigma^{k+1}}{1 - \sigma}\|x_0 - x^*\|.$$

Now we prove the bound on $\|P\|$. Let $\tilde{R} = R + \Delta$ for some perturbation matrix Δ. Then

$$\begin{aligned}\|P\| &= \|R^T R - \tilde{R}^T \tilde{R}\|, \\ &\leq 2\|\Delta\|\|R\| + \|\Delta\|^2.\end{aligned}$$

It remains to bound $\|\Delta\|$. Consider $\bar{X}$, where each column of $\bar{X} = (x_i - \bar{x})$ for some point $\bar{x}$. Then we can build R from X,

$$R = \bar{X} \begin{bmatrix} -1 & 1 & & \\ & -1 & 1 & \\ & & \ddots & \end{bmatrix} = \bar{X} D.$$

It is possible to show $\|D\| \leq 2$. Using the same logic, we can build $\tilde{R}$,

$$\tilde{R} = (X + \mathcal{E})D = R + \mathcal{E}D.$$

By identification, we have $\Delta = \mathcal{E}D$, so $\|\Delta\| \leq \|D\|\|\mathcal{E}\| \leq 2\|\mathcal{E}\|$. □

Assuming again $\|G\| \leq \sigma$, the following propositions bound $\|\bar{X}\|$ when $\bar{x} = x^*$.

Proposition 3.7 *Let $\bar{X}$ be the matrix built with the columns $\bar{X}_i = x_i - \bar{x}$, where the sequence x_i is generated by* (LFPI) *and $\bar{x} = x^*$. If $\|G\| \leq \sigma$, where G is the matrix present in* (LFPI)*, the norm of $\bar{X}$ is bounded by*

$$\|\bar{X}\| \leq \frac{1 - \sigma^{k+1}}{1 - \sigma}\|x_0 - x^*\|. \tag{35}$$

Proof Since each column of $\bar{X}$ correspond to $x_i - x^*$,

$$\|\bar{X}\| \leq \sum_{i=0}^{k} \|x_i - x^*\| \leq \sum_{i=0}^{k} \|G^i(x_0 - x^*)\| \leq \sum_{i=0}^{k} \|G\|^i \|x_0 - x^*)\|.$$

Because $\|G\| \leq \sigma < 1$,

$$\|\bar{X}\| \leq \frac{1 - \sigma^{k+1}}{1 - \sigma}\|x_0 - x^*\|,$$

which is the desired result. □

The bound of Theorem 3.5 is quite generic. For now, we only need a sequence $\tilde{x}_k$ generated by a perturbed fixed-point process which is convergent and differentiable, and the accuracy depends on matrices $\tilde{R}$ and $\mathcal{E}$. The next section will bound these quantities when the fixed point process is the gradient descent algorithm.

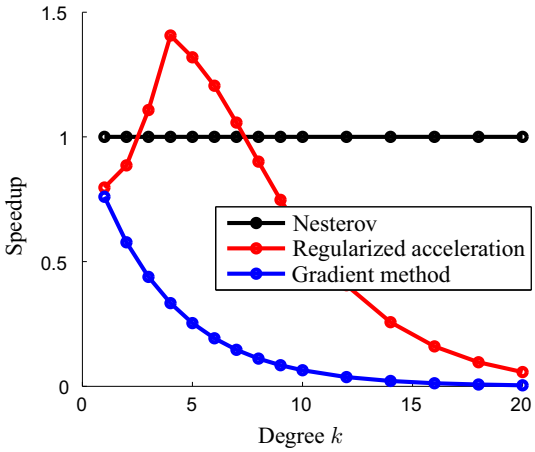

Fig. 3 Convergence speedup relative to Nesterov's accelerated method of theoretical bound in Theorem 3.5 and gradient method, using upper bounds from Propositions 3.6, 3.7 and 3.8. We see that our (highly conservative) bound shows a slight speedup when k is well chosen

3.5 Accelerating gradient descent

Assume the sequence $\tilde{x}_i$ is generated by the gradient descend algorithm,

$$\tilde{x}_{i+1} = \tilde{x}_i - \frac{1}{L} f'(\tilde{x}_i),$$

where f is a μ-strongly convex, L-smooth function with a Lipschitz-continuous Hessian with constant M. In this case, we can bound the values $\|\tilde{R}\|$ and $\|\mathcal{E}\|$ and hence $\|P\|$. We show the following result in Sect. A.2.

Proposition 3.8 *When using gradient method on a μ-strongly convex, L-smooth function with a Lipschitz-continuous Hessian with constant M, we have the following bounds,*

$$\|\tilde{R}\| \leq \frac{1-\sigma^{k+1}}{1-\sigma} L \|x_0 - x^*\|, \tag{36}$$

$$\|\mathcal{E}\| \leq (k+2)^2 \frac{M}{4L} \|x_0 - x^*\|^2, \tag{37}$$

where $\sigma = 1 - \frac{\mu}{L}$ satisfies $\|G\| \leq \sigma$.

Using these expressions, we can compare convergence rates between convergence acceleration in Algorithm 2 and Nesterov's method. In Fig. 3 we illustrate the difference on a particular instance where $\|x_0 - x^*\| = 10^{-4}$, $L = 1$, $\mu = M = 0.1$. We see that, despite the highly conservative nature of this bound, for small k at least our method is faster than Nesterov's acceleration.

When using the gradient method, this result bounds all quantities present in Theorem 3.5 as a function of μ, L, M and $\|x_0 - x^*\|$. Asymptotically, i.e. when $\|x_0 - x^*\| \to 0$ and we are starting close enough to the optimal point, we show that we recover the acceleration rate for linear sequences in Proposition 2.3 if the regularization parameter λ il well-chosen.

Proposition 3.9 *Assume we used the gradient method on a L-smooth and μ-strongly convex function with Lipschitz-continuous Hessian to generate the sequence $\tilde{x}_i$. Setting $\lambda = O(\|x_0 - x^*\|^s)$ with $s \in]2, \frac{8}{3}[$ and $\|x_0 - x^*\| \to 0$ (i.e., starting close enough to the optimal point), then the rate of convergence of the extrapolated point is bounded by*

$$\lim_{\|x_0 - x^*\| \to 0} \frac{\|\sum_{i=0}^k \tilde{c}_\lambda^* \tilde{x}_i - x^*\|}{\|x_0 - x^*\|} \leq \kappa \frac{2\beta^k}{1+\beta^{2k}}, \qquad \beta = \frac{1-\sqrt{\kappa^{-1}}}{1+\sqrt{\kappa^{-1}}},$$

where $\kappa = \frac{L}{\mu}$.

Proof The bounds above show

$$\|\bar{X}\| = \|\tilde{R}\| = O(\|x_0 - x^*\|), \qquad \|\mathcal{E}\| = O(\|x - x^*\|^2), \qquad \|P\| = O(\|x_0 - x^*\|^3).$$

Let $\lambda = O(\|x_0 - x^*\|^s)$ for some scalar s. The bound of Theorem 3.5 normalized by $\|x_0 - x^*\|$ becomes

$$S_\sigma(k, O(\|x_0 - x^*\|^{s-2}))\sqrt{\kappa^2 + O(\|x_0 - x^*\|^{8-3s})} + \sqrt{O(\|x_0 - x^*\|^2) + O(\|x_0 - x^*\|^{4-s})}.$$

If $4 - s < 0$, clearly the last terms vanishes when $\|x_0 - x^*\| \to 0$. It remains to analyze

$$\lim_{\|x_0 - x^*\| \to 0} S_\sigma(k, O(\|x_0 - x^*\|^{s-2}))\sqrt{\kappa^2 + O(\|x_0 - x^*\|^{8-3s})}.$$

If $s \in]2, 8/3[$ then $s - 2 > 0$ and $8 - 3s > 0$, implying $(\|x_0 - x^*\|^{s-2}) \to 0$ and $O(\|x_0 - x^*\|^{8-3s}) \to 0$ when $\|x_0 - x^*\| \to 0$. The bound finally becomes

$$\lim_{\|x_0 - x^*\| \to 0} \frac{\|\sum_{i=0}^k \tilde{c}_\lambda^* \tilde{x}_i - x^*\|}{\|x_0 - x^*\|} \leq \kappa S_\sigma(k, 0).$$

However, $S_\sigma(k, 0)$ is exactly equal to the maximum value of the rescaled (non-regularized) Chebyshev polynomial $T_k(x, \sigma)$, so by equation (9),

$$S_\sigma(k, 0) = \max_{x \in [0, \sigma]} T_k(x, \sigma) = \frac{2\beta^k}{1+\beta^{2k}}.$$

This result conclude the proof. □

In other words, the result above means that the bound of Theorem 3.5 tends to be the bound of Proposition 2.3 (for the case where $k < m$) and, asymptotically, we recover the optimal rate of convergence in [23]. In fact, the result may also hold for other kinds of methods, because the proof only needs

$$\|\bar{X}\| = \|\tilde{R}\| = O(\|x_0 - x^*\|), \qquad \|\mathcal{E}\| = O(\|x - x^*\|^2), \qquad \|P\| = O(\|x_0 - x^*\|^3).$$

These assumptions are not too restrictive, and are often encountered when using deterministic, twice-differentiable, linearly convergent iterations g in (FPI).

In practice of course, $\|x - x^*\|$ is unknown and the regularization parameter λ should decrease fast enough to ensure $S_\sigma(k, \bar{\lambda}) \to S_\sigma(k, 0)$, but not too fast otherwise the algorithm becomes unstable. An adaptive strategy thus ensures a good convergence rate, which is what we detail next.

3.6 Adaptive regularization

The major problem of the regularized Algorithm 2 is the presence of the parameter λ, unknown in advance. Of course, one can use the bound in Theorem 3.5 to search the best λ, but this requires a lot of information on the problem, like the constants L, μ and M as well as the distance to the optimum $\|x_0 - x^*\|$. Moreover, the bound is extremely pessimistic and does not correspond to the good numerical performances of the algorithm.

To avoid this problem we use adaptive strategy to find λ, based on grid search, which requires k additional calls to $f(x)$. In comparison, we also need to call k times the oracle for common adaptive strategy in the (accelerated) gradient method. For example, the backtracking line-search over the constant L requires the evaluation of $f(x_i)$ at each iteration $i = 1...k$.

Finally, the introduction of the regularization parameter introduces some damping in the acceleration algorithm, in the sense that the step length $x_{\text{extr}}(\lambda) - x_0$ is reduced with higher values of λ. A simple line search over the step-size, which consists in finding a good scalar t which minimizes the function, solving

$$\min_{t>0} f\big(x_0 + t(\underbrace{x_{\text{extr}}(\lambda) - x_0}_{\text{Extrapolation step}})\big)$$

significantly improves the solution. Nevertheless, this requires further calls to $f(x)$, and an inexact line-search is usually preferable. We start with $t = 1$, then multiply the value by two until the objective function increases,

$$f\big(x_0 + t(x_{\text{extr}}(\lambda) - x_0)\big) < f\big(x_0 + 2t(x_{\text{extr}}(\lambda) - x_0)\big).$$

In our numerical experiments, this line-search dramatically increases acceleration performances.

We summarize all the steps detailed above as the *Adaptive Regularized Convergence Acceleration* Algorithm 3. The only required inputs are the sequence $\tilde{x}_i$ generated by the optimization algorithm and the objective function f.

3.7 Computational complexity of convergence acceleration

In Algorithm 2, computing the coefficients $\tilde{c}^*_\lambda$ means solving the $k \times k$ system $(\tilde{R}^T \tilde{R} + \lambda I)z = \mathbf{1}$. We then get $\tilde{c}^*_\lambda = z/(\mathbf{1}^T z)$. This can be done in both batch and online mode. We will see that, in any case, we end with a complexity of $O(nk^2 + k^3)$, for a small

Algorithm 3 Adaptive Regularized Nonlinear Acceleration of Convergence

Input: Sequence $\{\tilde{x}_0, \tilde{x}_1, ..., \tilde{x}_{k+1}\}$, bounds $[\lambda_{\min}, \lambda_{\max}]$, objective function $f(x)$.
1: Divide the segment $[\lambda_{\min}, \lambda_{\max}]$ into k points $\{\lambda_j\}$ using a logarithmic scale.
2: Compute the residual matrix $\tilde{R}$ such that $\tilde{R}_i = \tilde{x}_{i+1} - \tilde{x}_i$.
3: Build the matrix $M = \tilde{R}^T \tilde{R} / \|\tilde{R}^T \tilde{R}\|$
4: **for** j in $1...k$ **do**
5: Solve in z the linear system $(M + \lambda_j)z = \mathbf{1}$
6: Normalize the solution, $\tilde{c}^*_{\lambda_j} = z/\mathbf{1}^T z$
7: Compute $x_{\text{extr}}(\lambda_j) = \sum_{i=0}^{k} (\tilde{c}^*_{\lambda_j})_i \tilde{x}_i$
8: **end for**
9: Pick $x^*_{\text{extr}} = \underset{j=1..k}{\text{argmin}}\, f(x_{\text{extr}}(\lambda_j))$
10: Define $F_t = f(x_0 + t(x^*_{\text{extr}} - x_0))$
11: Initialize with $t = 1$
12: **while** $F_{2t} < F_t$ **do**
13: Update $t = 2t$
14: **end while**
Output: Return $(x_0 + t(x^*_{\text{extr}} - x_0))$, the extrapolated point.

value of k (usually, $k = 5$). The complexity of the acceleration algorithm is linear in the dimension, thus adding a negligible additional computation cost to the original procedure.

3.7.1 Online updates

Here, we receive the vectors r_i one by one from the optimization algorithm, and we would like to solve the linear system in parallel of the optimization algorithm. In this case, we perform low-rank updates on the Cholesky factorization of the system matrix. At iteration i, we have the Cholesky factorization $LL^T = \tilde{R}^T \tilde{R} + \lambda I$, where L is a triangular matrix. We receive a new vector r_+ and we want

$$L_+ L_+^T = \begin{bmatrix} L & 0 \\ a^T & b \end{bmatrix} \begin{bmatrix} L^T & a \\ 0 & b \end{bmatrix} = \begin{bmatrix} \tilde{R}^T \tilde{U} + \lambda I & \tilde{R}^T r_+ \\ (\tilde{R}^T r_+)^T & r_+^T r_+ + \lambda \end{bmatrix}.$$

We can explicitly solve this system in variables a and b, and the solutions are

$$a = L^{-1} \tilde{R}^T r_+, \quad b = a^T a + \lambda.$$

The complexity of this update is thus $O(i\, n + i^2)$, i.e. the matrix-vector multiplication of $\tilde{R}^T r_+$ with cost $O(i\, n)$ and solving a $i \times i$ triangular system with cost $O(i^2)$. Since we need to do it k times, the final complexity is thus $O(nk^2 + k^3)$.

3.7.2 Batch mode

The complexity is divided in two parts: First, we need to build the linear system itself. Since $\tilde{R} \in \mathbb{R}^{n \times k}$, it takes $O(nk^2)$ flops to perform the multiplication. Then we need to solve the linear system $(\tilde{R}^T \tilde{R} + \lambda I)z = \mathbf{1}$ which can be done by Gaussian elimination

(in particular when k is small), by Cholesky factorization or by using an iterative method like conjugate gradient. It takes $O(k^3)$ flops to solve the linear system in the worst case, meaning that the overall complexity is $O(nk^2 + k^3)$.

4 Extensions & links with other methods

4.1 Smooth minimization

We can extend our results to smooth functions that are not strongly convex using a simple regularization trick which we trace back at least to Hazan [12]. Suppose we seek to solve

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

in the variable $x \in \mathbb{R}^n$, where $f(x)$ has a Lipschitz continuous gradient with parameter L with respect to the Euclidean norm, but is not strongly convex. Assume for simplicity that the initial iterate x_0 is close enough to the optimum so that $D \triangleq \|x_0 - x^*\| \geq \|x_k - x^*\|$ for any $k \geq 0$. We can approximate the above problem by

$$\min_{x \in \mathbb{R}^n} f_\varepsilon(x) \triangleq f(x) + \frac{\varepsilon}{2D^2}\|x_0 - x\|_2^2 \tag{38}$$

in the variable $x \in \mathbb{R}^n$, where $f_\varepsilon(x)$ has a Lipschitz continuous gradient with parameter $L + \varepsilon/D^2$ with respect to the Euclidean norm, is strongly convex with parameter ε/D^2 with respect to the same norm. Furthermore $f_\varepsilon(x)$ is an ε approximation of $f(x)$ near the optimum and we get

$$\begin{aligned} f(x_k) - f(x^*) &= f_\varepsilon(x_k) - \frac{\varepsilon}{2D^2}\|x_0 - x_k\|_2^2 - f_\varepsilon(x^*) + \frac{\varepsilon}{2D^2}\|x_0 - x^*\|_2^2, \\ &\leq f_\varepsilon(x_k) - f_\varepsilon(x^*) + \frac{\varepsilon}{2}, \\ &\leq f_\varepsilon(x_k) - f_\varepsilon(x_\varepsilon^*) + \frac{\varepsilon}{2}, \end{aligned}$$

using the smoothness of $f_\varepsilon(x)$ and writing x_ε^* the optimum of problem (38). It suffices to optimize f_ε up to $\varepsilon/2$ to find an ε-solution for the original problem. The linear convergence of gradient [23] algorithms guarantees

$$f_\varepsilon(x_k) - f_\varepsilon(x_\varepsilon^*) = \frac{(L+\varepsilon)D^2}{2} r^k, \qquad r = 1 - \frac{2\varepsilon}{LD^2 + 2\varepsilon}.$$

The number of iterations required to reach a target precision $\varepsilon/2$ is thus bounded by

$$k = O\left(\frac{\log((L+\varepsilon)D^2/\varepsilon)}{\log(1/r)}\right).$$

By replacing the value of r, we have

$$\log(1/r) \sim 1 - r = \frac{LD^2}{\varepsilon},$$

while accelerated algorithms have $r = 1 - \sqrt{\varepsilon/(LD^2 + \varepsilon)}$ which yields

$$\log(1/r) \sim \sqrt{\frac{LD^2}{\varepsilon}}.$$

Up to a logarithmic constant, these upper bounds match the complexity of gradient and accelerated gradient methods. Overall, an algorithm for strongly convex function used with this regularization trick recovers an ε-approximated solution. This means we can always reduce a not strongly convex problem to (1), where our acceleration analysis applies.

4.2 Convergence acceleration on gradient method for quadratic functions

Assume we want to minimize a quadratic function f. Its gradient reads, for $A \in \mathbb{R}^{n\times n}$ a symmetric positive definite matrix,

$$\nabla f(x) = A(x - x^*).$$

This formulation is equivalent to $\nabla f = Ax - b$, where $b = Ax^*$ but it will be more convenient in this section to manipulate directly x^*. Let $\mu I \preceq A \preceq LI$ so that the function f is strongly convex of constant μ and smooth of constant L. If we use the fixed-step gradient method, with step-size $1/L$,

$$x_{i+1} = x_k - \frac{1}{L}\nabla f(x_k) = x_k - \frac{1}{L}A(x_k - x^*). \tag{39}$$

The fixed point iteration corresponds to

$$g(x) = (I - A/L)(x - x^*) + x^*.$$

Notice that $g(x_{k+1})$ and (39) are equivalent. The Jacobian of g is thus equal to $(I - A/L)$. We have the following bounds on G,

$$0 \preceq G \preceq \left(1 - \frac{\mu}{L}\right) I.$$

By consequence, $\sigma = 1 - \frac{\mu}{L}$, thus the rate of convergence of our method is linear and the bound is

$$\|x_k - x^*\| \leq \left(1 - \frac{\mu}{L}\right)^k \|x_0 - x^*\|.$$

However, if we use Algorithm (1), we combine the iterates x_i with coefficients c^* (computed by formula (15)). By equations (5) and (13) the accuracy of this extrapolation is bounded by

$$\left\| \sum_{i=0}^{N} c_i^* x_i - x^* \right\| \leq \frac{L}{\mu} \frac{2\beta^k}{1+\beta^{2k}} \|x_0 - x^*\|, \qquad \text{where } \beta = \frac{1-\sqrt{\frac{\mu}{L}}}{1+\sqrt{\frac{\mu}{L}}}. \tag{40}$$

This bound matches the rate obtained using the optimal method in [23]. Outside of the normalization constraint, this is very similar to the convergence analysis of Lanczos' method.

4.3 Convergence acceleration versus conjugate gradient

The rate of convergence obtained above also matches that of the conjugate gradient within a factor L/μ. Indeed, the acceleration algorithm has a strong link with the conjugate gradient. Denote $\|v\|_M = \sqrt{v^T M v}$ the norm induced by the positive definite matrix M. Also, assume we want to solve $Ax = b$ using conjugate gradient method (where A is assumed to be symmetric and positive definite). By definition, at the k-th iteration, the conjugate gradient computes an approximation of x^* which follows

$$\operatorname*{argmin}_{x \in \mathcal{K}_k} \|x - x^*\|_A,$$

where $\mathcal{K}_k = \text{span}\{b, Ab, ..., A^{k-1}b\} = \text{span}\{Ax^*, A^2x^*, ..., A^kx^*\}$ is called a Krylov subspace. Since the constraint $x \in \mathcal{K}_k$ impose us to build x from a linear combination of the basis of $\mathcal{K}_k$, we can write

$$x = \sum_{i=0}^{k-1} c_i A^{i+1} x^* = q(A)x^*,$$

where $q(x)$ is a polynomial of degree k and $q(1) = 0$. So the conjugate gradient method solves

$$\operatorname*{argmin}_{\{q \in \mathbb{R}_k[x] : q(0)=0\}} \|q(A)x^* - x^*\|_A = \operatorname*{argmin}_{\{\hat{q} \in \mathbb{R}_k[x] : \hat{q}(0)=1\}} \|\hat{q}(A)x^*\|_A,$$

which is very similar to the equations in (12). However, while conjugate gradient has access to an oracle giving the result of the product between A and any vector v, the acceleration algorithm can only use the iterations produced by (LFPI), so it does not require the knowledge of A. Moreover, the convergence of conjugate gradient is analyzed in another norm ($\|\cdot\|_A$ instead of $\|\cdot\|_2$), which explains why a condition number appears in the bound (40).

Analysis of convergence on conjugate gradient often use Chebyshev's polynomial, like the acceleration algorithm (1). We will now see that Nesterov's algorithm generates also a polynomial, making the convergence analysis for quadratics easier.

4.4 Chebyshev's acceleration and Nesterov's accelerated gradient method

In Proposition 2.1, we bounded the rate of convergence of Algorithm 1 using Chebyshev polynomials. In fact, this is exactly the idea behind Chebyshev's semi-iterative method, which uses these coefficients in order to accelerate gradient descent on quadratic functions. Here, we present Chebyshev semi-iterative acceleration and its analysis, then use the same arguments on Nesterov's method. These points were also discussed in [11].

Assume as above that we use the gradient method to minimize a quadratic function, we get the recurrence (39). We see easily that

$$x_k = x^* + G^k(x_0 - x^*).$$

Since $\|G\|_2 \leq 1 - \frac{\mu}{L} = \sigma$, the rate of convergence is $\|x_k - x^*\|_2 \leq \sigma^k \|x_0 - x^*\|_2$. Moreover, if we average the vectors x_i using coefficients c_i (with unitary sum) from 0 to k, we get

$$\sum_{i=0}^{k} c_i x_i = x^* + p(G)(x_0 - x^*)$$

for $p \in \mathbb{R}_k[x]$ a polynomial of coefficients c. Instead of using Algorithm (1), which minimizes the combination of the residuals instead of the error term, we will use the coefficients of the rescaled Chebyshev polynomial (8). Recall this polynomial makes $\|p(G)\|_2$ small for all matrices G such that $0 \preceq G \preceq \sigma I$. In other terms, the rescaled Chebyshev polynomial satisfies

$$\begin{aligned} T(x) &= \arg \min_{\substack{p \in \mathbb{R}[x] \\ p(1)=1}} \max_{0 \preceq G \preceq \sigma I} \|p(G)\|_2, \\ &= C_k(t_\sigma(x)). \end{aligned}$$

where T_k and t_σ are also defined in (8). Furthermore, the Chebyshev polynomials can be constructed using a three-terms recurrence

$$C_k(x) = xC_{k-1}(x) - C_{k-2}(x).$$

The same holds for $T_k(x)$, with

$$\begin{aligned} \alpha_k &= t(1)\alpha_{k-1} - \alpha_{k-2}, \\ z_{k-1} &= y_{k-1} - \nabla f(y_{k-1}), \end{aligned}$$

$$y_k = \frac{\alpha_{k-1}}{\alpha_k}\left(\frac{2z_{k-1}}{\sigma} - y_{k-1}\right) - \frac{\alpha_{k-2}}{\alpha_k} y_{k-2}.$$

This scheme looks very similar to Nesterov's accelerated gradient method, which reads

$$\begin{aligned} z_{k-1} &= y_{k-1} - \nabla f(y_{k-1}) \\ y_k &= z_{k-1} + \beta_k(z_{k-1} - z_{k-2}) \end{aligned}$$

Compared with Chebyshev acceleration, Nesterov's scheme is iteratively building a polynomial $N_k(x)$ with $y_k - y^* = N_k(G)(y_0 - x^*)$. If we replace z_k by its definition in the expression of y_k in the Nesterov's scheme we get the following recurrence of order two

$$\begin{aligned} y_k - x^* &= (1+\beta_k)G(y_{k-1} - y^*) - \beta_k G(y_{k-2} - y^*), \\ &= G\left((1+\beta_k)N_{k-1}(G) - \beta_k N_{k-2}(G)\right)(y_0 - x^*). \end{aligned}$$

We can extract the polynomial N_k, which reads

$$N_k(x) = x((1+\beta_k)N_{k-1}(x) - \beta_k N_{k-2}(x)),$$

with initial conditions $N_0(x) = 1$ and $N_1(x) = x$. Notice that $N_k(1) = 1$ for all k.

When minimizing smooth strongly convex functions with Nesterov's method, we use

$$\beta_k = \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}.$$

Moreover, empirically at least, the maximum value of $N_k(x)$ in the interval $[0, \sigma]$ is $N_k(\sigma)$. We conjecture that this always holds. We thus have the following recurrence

$$N_k(\sigma) = \sigma\left((1+\beta)N_{k-1}(\sigma) - \beta N_{k-2}(\sigma)\right)$$

To get linear convergence with rate r, we need $N_k \leq rN_{k-1} \leq r^2 N_{k-2}$, or again

$$N_k(\sigma) \leq \sigma\left((1+\beta)rN_{k-2}(\sigma) - \beta N_{k-2}(\sigma)\right) = \sigma\left((1+\beta)r - \beta\right)N_{k-2}(\sigma).$$

Now, consider the condition

$$\sigma\left((1+\beta)r - \beta\right) \leq r^2.$$

We have that Nesterov's coefficients and rate, i.e. $\beta = (1-\sqrt{\mu/L})/(1+\sqrt{\mu/L})$ and $r = (1-\sqrt{\mu/L})$, satisfy this condition, showing that Nesterov's method converges with a rate at least $r = (1-\sqrt{\mu/L})$ on quadratic problems. This provides an alternate proof of Nesterov's acceleration result on these problems using Chebyshev polynomials (provided the conjecture on $N(\sigma)$ holds).

5 Numerical experiments

In this section, we evaluate the performance of the adaptive acceleration methods without/with line-search on the step size, described in Algorithm 3.

5.1 Minimizing logistic regression

We begin by testing our methods on a regularized logistic regression problem written

$$f(w) = \sum_{i=1}^{m} \log\left(1 + \exp(-y_i \xi_i^T w)\right) + \frac{\tau}{2}\|w\|_2^2,$$

where $Z = [\xi_1, ..., \xi_m]^T \in \mathbb{R}^{m\times n}$ is the design matrix and y is a $\{-1, 1\}^m$ vector of labels. The Lipschitz constant of the logistic regression is $L = \|Z\|_2^2/4 + \tau$ and the strong convexity parameter is $\mu = \tau$. We solve this problem using several algorithms.

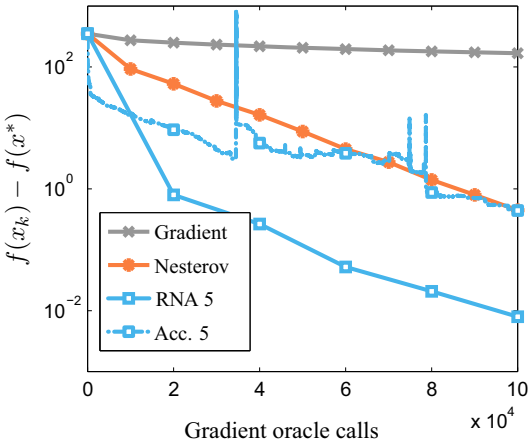

Fig. 4 Logistic regression on *Madelon UCI Dataset* with a condition number equal to $1.2 \cdot 10^9$, solved using Gradient method, Nesterov's method and two versions of the acceleration algorithm applied to the gradient descent: the acceleration Algorithm 1 (called *Acc. 5*) and the adaptive Regularized Nonlinear Acceleration algorithm 3 (called *RNA 5*) applied to 5 iterations of the gradient descend. We see that without regularization, the acceleration is unstable because $\|(\tilde{R}^T\tilde{R})^{-1}\|_2$ is huge (cf. Proposition 3.1)

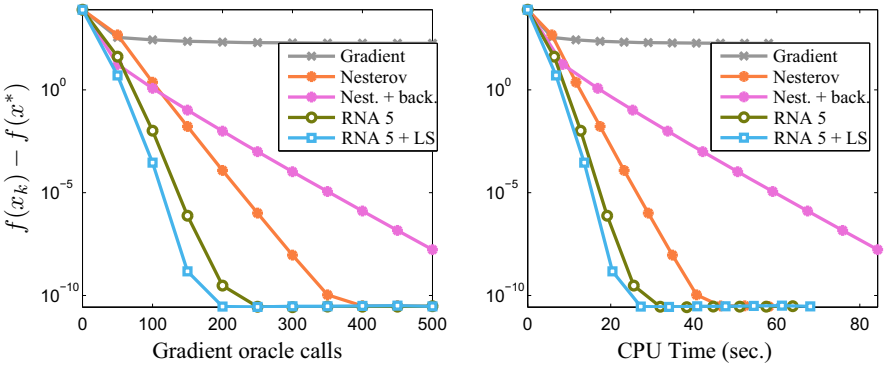

Fig. 5 Logistic regression on sido0 dataset, with $\tau = 10^2$ (condition number = $1.5 \cdot 10^5$)

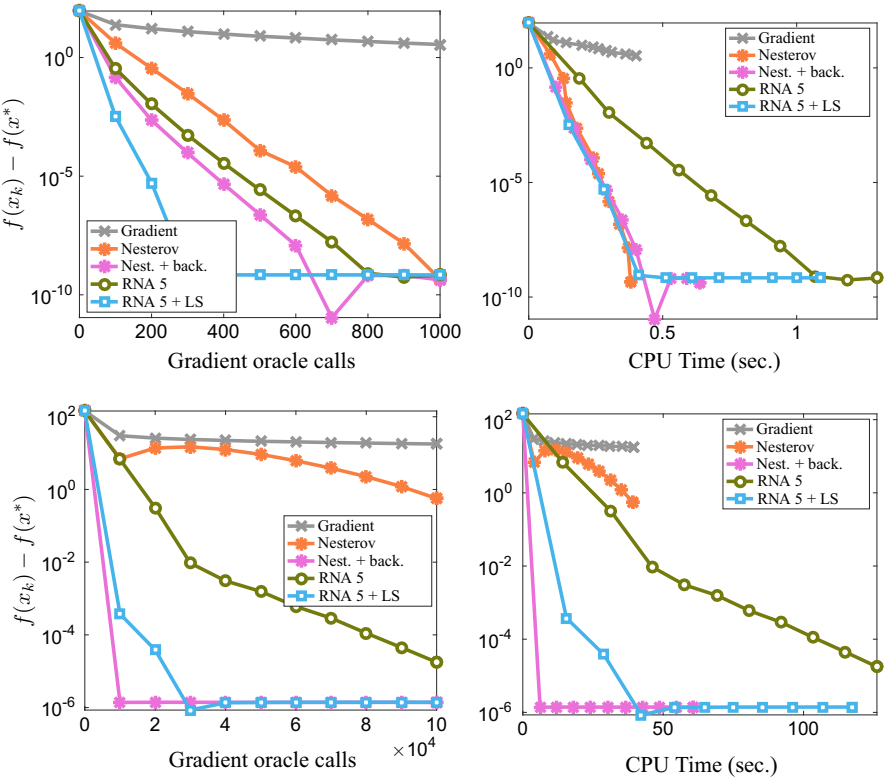

Fig. 6 Logistic regression on sonar dataset. From top to bottom, we used $\tau = 10^{-1}$ (condition number = $7 \cdot 10^3$) and $\tau = 10^{-6}$ (condition number = $7 \cdot 10^8$)

- Fixed-step gradient method for smooth strongly convex functions [23, Th. 2.1.15]

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

- Accelerated gradient method for smooth strongly convex functions [23, Th. 2.2.3]

$$x_{k+1} = y_k - \frac{1}{L} \nabla f(y_k),$$
$$y_{k+1} = x_{k+1} + \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}} (x_{k+1} - x_k).$$

- The accelerated gradient method with backtracking line-search on the parameter L.
- The Adaptive acceleration Algorithm 3 on k iterations of gradient descent without line search (written RNA k).
- The Black-box acceleration Algorithm 3 (written RNA k + LS) on k iterations of gradient descent.

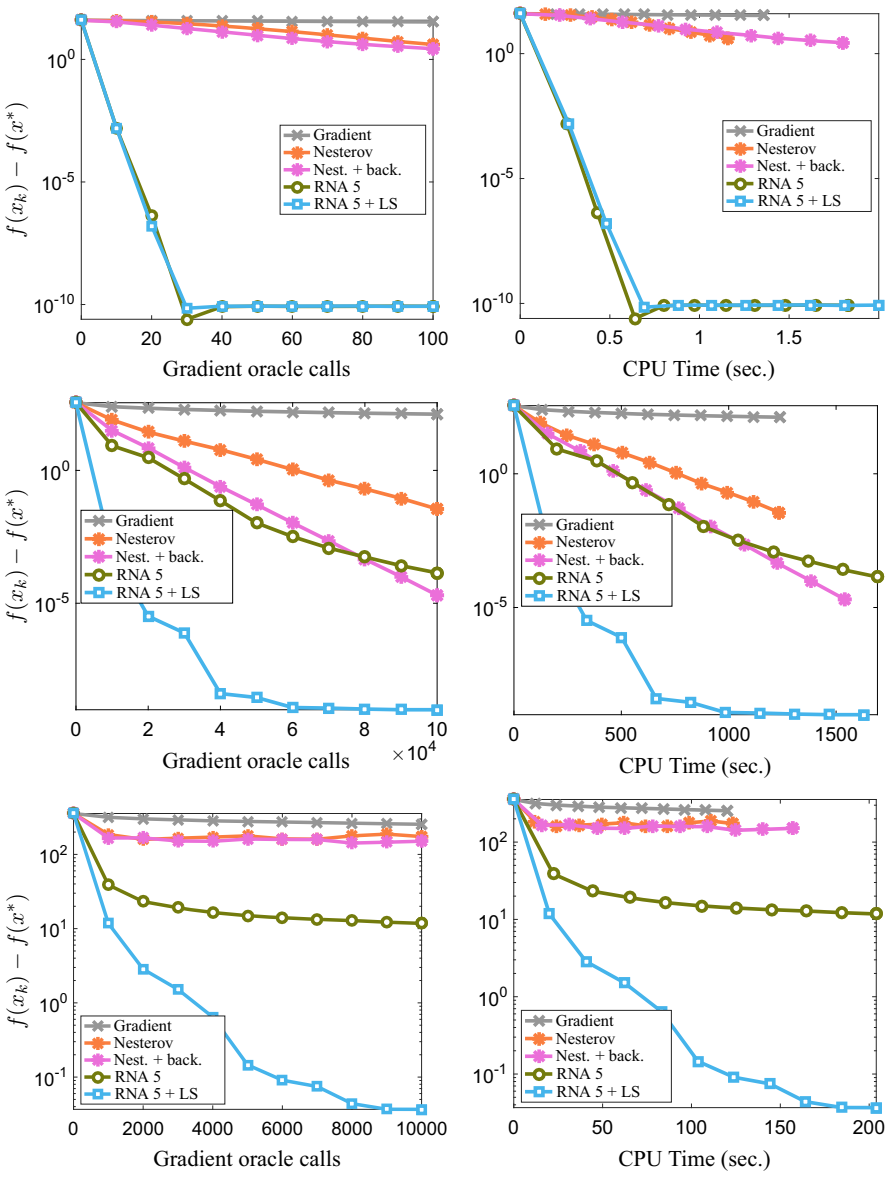

Fig. 7 Logistic regression on Madelon dataset. From top to bottom, we used , $\tau = 10^7$ (condition number $= 6 \cdot 10^3$), $\tau = 10^2$ (condition number $= 1.2 \cdot 10^9$ and $\tau = 10^{-3}$ (condition number $= 6 \cdot 10^{13}$)

The matrix Z is build using datasets *Sonar* (60 features, 208 points), *Madelon* (500 features, 4400 points) or *Sido0* (4932 features, 12,678 points), concatenated with a column of ones. The optimization is done on the raw data, i.e. without normalization. The starting point is always $w_0 = 0$.

Figure 4 shows the importance of the regularization in the acceleration algorithm. Indeed, if we use Algorithm 1 then the norm of the inverse of $\tilde{R}^T\tilde{R}$ may be huge, so the computation of the coefficients $\tilde{c}_\lambda^*$ is unstable. This leads to an unreliable acceleration method, which may improve sometimes the accuracy, but often making the process divergent. In Figs. 5, 6 and 7, we see that our algorithm has a similar behavior to the conjugate gradient: unlike the Nesterov's method, where we need to provide parameters μ and L, the acceleration algorithm adapts himself in function of the spectrum of G (so it can exploit the good local strong convexity parameter), without any prior specification. We can, for example, observe this behavior when the global strong convexity parameter is bad but not the local one.

6 Conclusion and perspectives

In this paper, we developed a method which is able to accelerate, under some regularity conditions, the convergence of a sequence $\{\tilde{x}_i\}$ without any information on the algorithm which generated this sequence. The regularization parameter used in the acceleration method is found by a simple and inexpensive grid-search. The algorithm itself is simple as it only requires solving a small linear system. Also, we showed (using gradient method on logistic regression) that the strategy which consists in restarting the algorithm after an extrapolation method can lead to significantly improved convergence rates. Future work will consist in improving the performance of the algorithm by exploiting the structure of the perturbations matrix in some cases and extending the algorithm to the stochastic case and to the non-symmetric case.

Acknowledgements AA is at the département d'informatique de l'ENS, École normale supérieure, UMR CNRS 8548, PSL Research University, 75005 Paris, France, and INRIA Sierra project-team. The authors would like to acknowledge support from a starting grant from the European Research Council (ERC project SIPA), from the ITN MacSeNet (project number 642685), as well as support from the chaire *Économie des nouvelles données* with the *data science* joint research initiative with the *fonds AXA pour la recherche*, and from a Google focused award.

Appendix A: Missing propositions and proofs

A.1: Missing propositions

Proposition A.1 *Consider the function*

$$f(x) = \sqrt{a - \lambda x^2} + bx$$

defined for $x \in [0, \sqrt{a/\lambda}]$. *The its maximal value is attained at*

$$x_{opt} = \frac{b\sqrt{a}}{\sqrt{\lambda^2\kappa^2 + \lambda b^2}}$$

and its maximal value is thus, if $x_{opt} \in [0, \sqrt{a/\lambda}]$,

$$f_{\max} = \sqrt{a}\sqrt{\kappa^2 + \frac{b^2}{\lambda}}. \tag{41}$$

Proof The (positive) root of the derivative of f follows

$$b\sqrt{a - \lambda x^2} - \kappa\lambda x = 0 \qquad \Leftrightarrow \qquad x = \frac{b\sqrt{a}}{\sqrt{\lambda^2\kappa^2 + \lambda b^2}}.$$

If we inject the solution in our function, we obtain its maximal value,

$$\begin{aligned}
\kappa\sqrt{a - \lambda\left(\frac{b\sqrt{a}}{\sqrt{\lambda^2\kappa^2 + \lambda b^2}}\right)^2} + b\frac{b\sqrt{a}}{\sqrt{\lambda^2\kappa^2 + \lambda b^2}} &= \kappa\sqrt{a - \lambda\frac{b^2 a}{\lambda^2\kappa^2 + \lambda b^2}} \\
&\quad + b\frac{b\sqrt{a}}{\sqrt{\lambda^2\kappa^2 + \lambda b^2}}, \\
&= \kappa\sqrt{a - \lambda\frac{b^2 a}{\lambda^2\kappa^2 + \lambda b^2}} \\
&\quad + b\frac{b\sqrt{a}}{\sqrt{\lambda^2\kappa^2 + \lambda b^2}}, \\
&= \kappa\sqrt{\frac{a\lambda^2\kappa^2}{\lambda^2\kappa^2 + \lambda b^2}} + b\frac{b\sqrt{a}}{\sqrt{\lambda^2\kappa^2 + \lambda b^2}}, \\
&= \sqrt{a}\frac{\kappa^2\lambda + b^2}{\sqrt{\lambda^2\kappa^2 + \lambda b^2}}, \\
&= \frac{\sqrt{a}}{\lambda}\sqrt{\lambda^2\kappa^2 + \lambda b^2}.
\end{aligned}$$

The simplification with λ in the last equality concludes the proof. □

A.2: Proof of proposition 3.8

First, we show that the choice $\sigma = 1 - \frac{\mu}{L}$ satisfies $\|G\| = \|g'(x^*)\| \leq \sigma$. Our fixed-point function g reads

$$g(x) = x - \frac{1}{L}f'(x).$$

Since $g'(x) = I - \frac{1}{L}f''(x)$, we have $g'(x^*) = I - \frac{1}{L}f''(x^*)$. Because f is μ-strongly convex, $f''(x) \succeq \mu I$, in particular at $x = x^*$. In conclusion,

$$\|g'(x^*)\| = \|I - \frac{1}{L}f''(x^*)\| \leq 1 - \frac{\mu}{L}.$$

Now, consider the matrix $\tilde{R}$. Since the $i-th$ column $\tilde{R}_i$ is equal to $\tilde{x}_i - \tilde{x}_{i-1}$,

$$\begin{aligned}\|\tilde{R}_i\| &= \|\tilde{x}_i - \tilde{x}_{i-1}\|,\\ &= \frac{1}{L}\|f'(\tilde{x}_i)\|,\\ &\leq \|\tilde{x}_i - x^*\|.\end{aligned}$$

In the last inequality, we used the fact that f is L-Lipschitz, so $\|f(x) - f(x^*)\| \leq L\|x - x^*\|$. It is also possible to prove [23] that gradient method converges at rate

$$\|\tilde{x}_{i+1} - x^*\| \leq \sigma\|x_i - x^*\|.$$

It remains to link this quantity to $\|\tilde{R}\|$,

$$\begin{aligned}\|\tilde{R}\| &\leq \sum_{i=0}^{k}\|R_i\|,\\ &\leq \sum_{i=0}^{k}\sigma^i\|x_0 - x^*\|,\\ &= \frac{1-\sigma^{k+1}}{1-\sigma}\|x_0 - x^*\|.\end{aligned}$$

We continue with $\|\mathcal{E}\|$. We express $\|\mathcal{E}_i\| = \|\tilde{x}_{i+1} - x_{i+1}\|_2$ in function of $\|\tilde{x}_0 - x_0\|_2$ using a recursion with $\|\tilde{x}_i - x_i\|_2$,

$$\begin{aligned}\tilde{x}_{i+1} - x_{i+1} &= \tilde{x}_i - \frac{1}{L}\nabla f(\tilde{x}_i) - x_i + \frac{1}{L}\nabla^2 f(x^*)(x_i - x^*),\\ &= \tilde{x}_i - x_i - \frac{1}{L}(\nabla f(\tilde{x}_i) - \nabla^2 f(x^*)(x_i - x^*)),\\ &= \left(I - \frac{\nabla^2 f(x^*)}{L}\right)(\tilde{x}_i - x_i) - \frac{1}{L}(\nabla f(\tilde{x}_i) - \nabla^2 f(x^*)(\tilde{x}_i - x^*)).\end{aligned}$$

Since our function has a Lipschitz-continuous Hessian, it is possible to show that ([23], Lemma 1.2.4)

$$\left\|\nabla f(y) - \nabla f(x) - \nabla^2 f(x)(y-x)\right\|_2 \leq \frac{M}{2}\|y - x\|^2. \tag{42}$$

We can thus bound the norm of the error at the ith iteration,

$$\begin{aligned}\|x_{i+1} - \tilde{x}_{i+1}\|_2 &\leq \left\|I - \frac{\nabla^2 f(x^*)}{L})\right\|_2\|x_i - \tilde{x}_i\|_2 + \frac{1}{L}\left\|\nabla f(\tilde{x}_i) - \nabla^2 f(x^*)(\tilde{x}_i - x^*)\right\|_2,\\ &= \|g''(x^*)\|_2\|x_i - \tilde{x}_i\|_2 + \frac{1}{L}\left\|\nabla f(\tilde{x}_i) - \nabla f(x^*) - \nabla^2 f(x^*)(\tilde{x}_i - x^*)\right\|_2.\end{aligned}$$

By equation (42), and because $\|g''(x^*)\| \leq \sigma$, we have

$$\begin{aligned}\|x_{i+1} - \tilde{x}_{i+1}\|_2 &\leq \sigma \|x_i - \tilde{x}_i\|_2 + \frac{M}{2L} \left\|\tilde{x}_i - x^*\right\|_2^2, \\ &\leq \sigma \|x_i - \tilde{x}_i\|_2 + \frac{M}{2L}\sigma^{2i}\|x_0 - x^*\|_2^2, \\ &\leq \|x_i - \tilde{x}_i\|_2 + \frac{M}{2L}\|x_0 - x^*\|_2^2.\end{aligned}$$

The simplification in the last line greatly simplifies future computations. We thus have the bound

$$\|x_{i+1} - \tilde{x}_{i+1}\|_2 \leq (i+1)\frac{M}{2L}\|x_0 - x^*\|^2.$$

Finally,

$$\begin{aligned}\|\mathcal{E}\| &\leq \sum_{i=0}^{k} \|x_{i+1} - \tilde{x}_{i+1}\|_2, \\ &\leq \sum_{i=0}^{k} (i+1)\frac{M}{2L}\|x_0 - x^*\|^2, \\ &\leq (k+2)^2\frac{M}{4L}\|x_0 - x^*\|^2.\end{aligned}$$

Despite the simplification made earlier, the results of this bounds are close to the one obtained without simplification.

References

1. Aitken, A. C.: XXV.—On Bernoulli's Numerical Solution of Algebraic Equations. In: Proceedings of the Royal Society of Edinburgh, vol. 46, pp. 289–305 (1927)
2. Anderson, D.G.: Iterative procedures for nonlinear integral equations. J. ACM (JACM) **12**(4), 547–560 (1965)
3. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(1), 183–202 (2009)
4. Ben-Tal, A., Nemirovski, A.: Lectures on modern convex optimization: analysis, algorithms, and engineering applications. SIAM (2001)
5. Brezinski, C.: Accélération de la convergence en analyse numérique, vol. 584. Springer, Berlin (2006)
6. Cabay, S., Jackson, L.: A polynomial extrapolation method for finding limits and antilimits of vector sequences. SIAM J. Numer. Anal. **13**(5), 734–752 (1976)
7. Drori, Y., Teboulle, M.: Performance of first-order methods for smooth convex minimization: a novel approach. Math. Program. **145**(1–2), 451–482 (2014)
8. Durbin, J.: The fitting of time-series models. Revue de l'Institut International de Statistique, pp. 233–244 (1960)
9. Eddy, R.: Extrapolating to the limit of a vector sequence. In: Information Linkage Between Applied Mathematics and Industry, pp. 387–396 (1979)

10. Golub, G.H., Varga, R.S.: Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order richardson iterative methods. Numerische Mathematik **3**(1), 147–156 (1961)
11. Hardt, M.: The zen of gradient descent (2013)
12. Hazan, E.: Personal communication (2014)
13. Heinig, G., Rost, K.: Fast algorithms for toeplitz and hankel matrices. Linear Algebra Appl. **435**(1), 1–59 (2011)
14. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. SIAM J. Optim. **11**(3), 796–817 (2001)
15. Lessard, L., Recht, B., Packard, A.: Analysis and design of optimization algorithms via integral quadratic constraints. SIAM J. Optim. **26**(1), 57–95 (2016)
16. Levinson, N.: The wiener rms error criterion in filter design and prediction, appendix b of wiener, n. (1949). Extrapolation, Interpolation, and Smoothing of Stationary Time Series (1949)
17. Lin, H., Mairal, J., Harchaoui, Z.: A universal catalyst for first-order optimization. In: Advances in Neural Information Processing Systems, pp. 3384–3392 (2015)
18. Mešina, M.: Convergence acceleration for the iterative solution of the equations x = ax + f. Comput. Methods Appl. Mech. Eng. **10**(2), 165–173 (1977)
19. Nemirovskii, A., Nesterov, Y.E.: Optimal methods of smooth convex minimization. USSR Comput. Math. Math. Phys. **25**(2), 21–30 (1985)
20. Nemirovskiy, A.S., Polyak, B.T.: Iterative methods for solving linear ill-posed problems under precise information. Eng. Cyber. **4**, 50–56 (1984)
21. Nesterov, Y.: A method of solving a convex programming problem with convergence rate o (1/k2). In: Soviet Mathematics Doklady, vol. 27, pp. 372–376 (1983)
22. Nesterov, Y.: Squared functional systems and optimization problems. In: High performance optimization, pp. 405–440. Springer, Berlin (2000)
23. Nesterov, Y.: Introductory Lectures on Convex Optimization: A Basic Course, vol. 87. Springer, Berlin (2013)
24. Nesterov, Y.: Universal gradient methods for convex optimization problems. Math. Program. **152**(1–2), 381–404 (2015)
25. Parrilo, P.A.: Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization, Ph.D. thesis, California Institute of Technology (2000)
26. Shanks, D.: Non-linear transformations of divergent and slowly convergent sequences. Stud. Appl. Math. **34**(1–4), 1–42 (1955)
27. Sidi, A., Ford, W.F., Smith, D.A.: Acceleration of convergence of vector sequences. SIAM J. Numer. Anal. **23**(1), 178–196 (1986)
28. Smith, D.A., Ford, W.F., Sidi, A.: Extrapolation methods for vector sequences. SIAM Rev. **29**(2), 199–233 (1987)
29. Su, W., Boyd, S., Candes, E.: In: Advances in Neural Information Processing Systems, pp. 2510–2518 (2014)
30. Tyrtyshnikov, E.E.: How bad are hankel matrices? Numerische Mathematik **67**(2), 261–269 (1994)
31. Wibisono, A., Wilson, A.C.: On accelerated methods in optimization. arXiv preprint (2015). arXiv:1509.03616
32. Wynn, P.: On a device for computing the e m (s n) transformation. In: Mathematical Tables and Other Aids to Computation, pp. 91–96 (1956)