université PARIS-SACLAY

CentraleSupélec

NNT : 2019SACLC042

Thèse de doctorat

Iterative methods with retards for the solution of large-scale linear systems

Thèse de doctorat de l'Université Paris-Saclay
préparée à CentraleSupélec

Ecole doctorale n°573 Approches interdisciplinaires: fondements, applications et innovation (INTERFACES)
Spécialité de doctorat : Mathématiques appliquées

Thèse présentée et soutenue à Gif-sur-Yvette, le 14 juin 2019, par

QINMENG ZOU

Composition du Jury :

Raphaël COUTURIER Professeur, Université de Franche-Comté	Président
Frédéric MAGOULES Professeur, CentraleSupélec	Directeur de thèse
Eric MONACELLI Professeur, Université de Versailles Saint-Quentin-en-Yvelines	Examinateur
Juliette RYAN Maître de Recherches, ONERA Professeur Associé, Université Paris 13	Examinateur
Pierre SPITERI Professeur Emérite, Institut National Polytechnique de Toulouse	Rapporteur
Damien TROMEUR-DERVOUT Professeur, Université Claude Bernard Lyon 1	Rapporteur

Iterative methods with retards for the solution of large-scale linear systems

Qinmeng ZOU

Laboratory MICS (Mathematics in Interaction with Computer Science)
CentraleSupélec, Université Paris-Saclay

This dissertation is submitted for the degree of
Doctor of Philosophy

June 2019

Acknowledgements

Most of all, I would like to thank my advisor, Frédéric Magoulès, for his endless guidance, feedback, and encouragement. His patient and hard work have inspired me over the course of my PhD study. Many thanks to my Master supervisor Lei Yu, for his valuable advice and introducing me to computer science. Many thanks, as well, to my committee members, Professors Raphaël Couturier, Eric Monacelli, Juliette Ryan, Pierre Spitéri, and Damien Tromeur-Dervout, all of whom provided advice and encouragement. I would also like to thank my research colleagues, Abal-Kassim Cheik Ahamed, Hanyu Zhang, Ge Song, Guillaume Gbikpi-Benissan, Mengchen Wang, for their help and friendship, who made this period enjoyable and interesting. Finally, I thank my parents and my fiancée Jing Zhang for always supporting me.

Abstract

Any perturbation in linear systems may severely degrade the performance of iterative methods when conjugate directions are constructed. This issue can be partially remedied by lagged gradient methods, which does not guarantee descent in the quadratic function but can improve the convergence compared with traditional gradient methods. Later work focused on alternate gradient methods with two or more steplengths in order to break the zigzag pattern. Recent papers suggested that revealing of second-order information along with lagged steps could reduce asymptotically the search spaces in smaller and smaller dimensions. This led to gradient methods with alignment in which essential and auxiliary steps are conducted alternately. Numerical experiments have demonstrated their effectiveness.

This dissertation first considers efficient gradient methods for solving symmetric positive definite linear systems. We begin by studying an alternate method with two-dimensional finite termination property. Then we derive more spectral properties for traditional steplengths. These properties allow us to expand the family of gradient methods with alignment and establish the convergence of new methods. We also treat gradient iterations as an inexpensive process embedded in splitting methods. In particular we address the parameter estimation problem and suggest to use fast gradient methods as low-precision inner solvers. For the parallel case we focus on the lagged formulations for which it is possible to reduce communication costs. We also present some new properties and methods for s-dimensional gradient iterations.

To sum up, this dissertation is concerned with three inter-related topics in which gradient iterations can be employed as efficient solvers, as embedded tools for splitting methods and as parallel solvers for reducing communication. Numerical examples are presented at the end of each topic to support our theoretical findings.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Background

In this section we provide some general background information on iterative methods for the solution of linear systems. The original development of this family of methods can date back to 1823, when Gauss wrote a private letter to Gerling (translated by Forsythe):

> I recommend this method to you for imitation. You will hardly ever again eliminate directly, at least not when you have more than 2 unknowns.

Here, "this method" was later viewed as a relaxation process, called Gauss-Seidel method. A general description with cyclic relaxation process was then formally published by Seidel [136]. In 1845, Jacobi [94] proposed a variant of Gauss' process now known as the Jacobi method. As mentioned in [153], these two methods were originally called "method of successive displacements" and "method of simultaneous displacements", respectively, see also [67, 148] for historical references.

Another direction of approach was based on the gradient methods, first proposed by Cauchy [33] in 1847, which was later known as the steepest descent (SD) method or the Cauchy method:

> Il suffira donc, ou de résoudre cette dernière équation, ou du moins d'attribuer à θ une valeur suffisamment petite, pour obtenir une nouvelle valeur de u inférieure à u. Si la nouvelle valeur de u n'est pas un minimum, on pourra en déduire, en opérant toujours de la même manière, une troisième valeur plus petite encore; et, en continuant ainsi, on trouvera successivement des valeurs de u de plus en plus petites ....

Here, u is the function value being reduced and θ later became known as the steplength of gradient methods (see Chapter 2). Temple is quoted as having found in 1938 that the problem

of solving a symmetric positive definite linear system is equivalent to that of minimizing a quadratic function (see [153]). The Cauchy method was then formalized by Kantorovitch in 1945 for solving such minimization problem (see [133]). This method is also referred to as the one-dimensional projection process (see [131]).

In 1910, Richardson [130] proposed an iterative method for solving linear systems, commonly called Richardson method, in which a sequence of extrapolation factors are exploited to improve the convergence. Choosing a constant value for all factors yields the stationary Richardson method (see [152]), or loosely called constant gradient method. The word "relaxation" was likely first used by Southwell [140] in 1940. Ten years later, Frankel [69] and Young [151] devised simultaneously the point successive overrelaxation (SOR) method. This method was referred to as the extrapolated Liebmann method in Frankel [69]. Inspired by the finite difference solution of elliptic partial differential equations, in 1955, Peaceman and Rachford [120] proposed the alternating direction implicit (ADI) method. In Chapter 4, we shall discuss an analogous splitting method called Hermitian and skew-Hermitian (HSS) method [11]. In the same year, Sheldon [137] introduced the symmetric SOR (SSOR) method. The relationship between the basic methods and the Chebyshev polynomials was also revealed at that moment, see Golub and Varga [82] and the references therein. For a detailed overview of the above methods, see [153, 133].

At the present time, there are still many cases that direct methods based on matrix factorizations can be effectively used for the solution of fairly large linear systems, see Duff et al. [59] and the references therein for a detailed overview of these methods. After around mid-1970s, however, Krylov subspace methods started gaining acceptance and became more popular than both relaxation methods and direct methods for solving a wide variety of problems. The original development of these methods can date back to 1950, when Lanczos [101] started to investigate eigenvalue algorithms, resulting in a pioneering work on what is now known as the Lanczos algorithm for finding eigenvalues of symmetric matrices. This work was then extended by Arnoldi [4] to the nonsymmetric case, called Arnoldi algorithm. For the solution of linear systems, in 1952, Hestenes and Stiefel [89] completed their seminal work of conjugate gradient (CG) method. Lanczos [102] proposed a similar method as well as the two-sided Lanczos procedure. These publications have enormous impact on scientific research, as quoted from [133]:"These four papers together set the foundations of many methods that were developed later". The potential of CG was not generally recognized until the work of Reid [129] in which CG was suggested as an iterative solver for sparse linear systems arising from the solution of Laplace's equation. See [81] for a chronological overview of early publications. Since then, CG became the method of choice and led to considerable research work in the solution of indefinite and nonsymmetric linear systems.

See [84, 131, 146] for good overviews of Krylov subspace methods, see also Section 4.1 and the references therein.

Another approach dealing with asynchronous iterations was considered by Chazan and Miranker [35] in 1969 for linear problems, and further generalized with contraction properties by Miellou [113], Baudet [18], El Tarazi [62], Miellou and Spiteri [114], Frommer and Szyld [75], Frommer and Szyld [76], and El Baz et al. [61]. Some papers made use of the partial ordering such as El Tarazi [63], El Baz et al. [60], and Miellou et al. [115]. Finally, Bertsekas [23] (see also [24]) proposed a general model based on the nested sets. In the past few years asynchronous iterative methods have received increasing attention, see Chau [34], Bahi et al. [9], Gbikpi-Benissan [78], and Partimbene [119] for more details.

Experience reveals that it is not easy to formalize the asynchronous variants based on the projection-type methods. Another direction to reduce communication in parallel environment is based on the communication-avoiding Krylov methods, see Hoemmen [91] for a good review. This strategy improves parallel properties for distributed environment, but leads to increase in computational complexity. An important observation is that the technique used in asynchronous iterations, termed "retards", can be used for the construction of similar coefficients for gradient methods, which leads to significant gains in efficiency.

In 1988, Barzilai and Borwein [17] proposed two gradient methods, later called Barzilai-Borwein (BB) methods (they shall be denoted by BB and BB2, respectively, see Section 2.3), that generates nonmonotone convergence curve, which means that the corresponding quadratic function might increase in some points throughout the iterative process. The irregular convergence led to so much suspicion about their robustness. It is interesting to note that the BiCG method [64, 102] faced the same challenge in the late 1970s until the introduction of the BiCGSTAB method [145]. Barzilai and Borwein [17] stated their motivation:

> The motivation for this choice is that it provides a two-point approximation to the secant equation underlying quasi-Newton methods ....

They concluded by saying:

> In view of the highly remarkable behavior of the new algorithms and the fact that they are not monotone, it is probably not surprising that our analysis, too, is entirely nonstandard. ... Finally, note that, since the two-point algorithms are not descent algorithms, they have an advantage in that the restriction to descent algorithms often results in small step sizes for ill-conditioned problems.

This pair of methods is indeed the first trial of lagged gradient iterations. In 1999, Friedlander et al. [74] presented a convergence framework, called gradient method with retards (GMR),

including the BB methods, but also admitting other variants. This is the origin of the word "retards" in the "iterative methods with retards". See Chapter 2 for details.

Most of the efficient gradient methods use retards in the gradient vector without focusing on the theoretical aspects. It is clear, however, that in many real problems they are less convincing than the methods based on spectral properties, called gradient methods with alignment. The challenge consists in finding more spectral properties other than the results based on the steepest descent by Nocedal et al. [117], De Asmundis et al. [56] and De Asmundis et al. [57], and proving the convergence of the new steplengths. On the other hand, when using parallel computing environment, the need for gradient algorithms specifically formalized for parallel processing is particularly important because communication among the processors has to be controlled.

This thesis shall not address the issue of preconditioners, although in Sections 4.3.1 and 5.5 there exist analogous techniques that are mentioned for practical purposes. Overviews for this topic can be found in [19, 149].

1.2 Summary

This thesis is organized as follows:

- In Chapter 2, we give an overview of gradient methods. We summarize the convergence theories of monotone gradient methods as well as some general gradient frameworks and the asymptotic results of steepest descent parameters upon the number of iterations. We also present a new choice of alternate steplength motivated by the two-dimensional finite termination property and the potential benefits of nonmonotone behavior.

- In Chapter 3, we derive the spectral properties of minimal gradient. We extend and propose three new gradient methods with alignment based on asymptotically optimal and minimal gradient steplengths, helping to extend the Cauchy-short framework to general cases. We then establish the convergence of new methods for the solution of symmetric positive definite systems. We show numerical experiments based on randomly generated matrices, two-point boundary value problems and a matrix drawn from a public collection.

- In Chapter 4, we introduce the application of gradient methods to the Hermitian and skew-Hermitian splitting method. We show that the asymptotic properties of steepest descent introduced in Chapter 2 hold for Hermitian systems. We propose several approaches to estimate the optimal value of parameter γ in the splitting method based

on gradient iterations. We also suggest low-precision lagged gradient methods as inner solvers. We conduct numerical experiments on randomly generated matrices and three-dimensional boundary value problems.

- In Chapter 5, we present parallel implementation of gradient methods. We focus on the gradient variants which can reduce communication costs and provide the relevant algorithms. For the s-dimensional gradient methods, we find two new properties related to the range of the descent steplengths. As a result, we propose two lagged s-dimensional methods based on the intermediate variables instead of the previous steplengths. Then we compare three gradient iterative schemes (basic scheme, lagged scheme, asynchronous scheme) and discuss the practical implementation issues. Numerical results are shown in the end for randomly generated matrices and linear systems drawn from a public matrix collection.

- Finally, in Chapter 6, we give some concluding remarks and discuss the future work.

1.3 Contribution

The primary contributions of this thesis include:

- We have proposed a new lagged gradient method with two-dimensional finite termination property. This method is faster than prior work for well-conditioned linear systems. Our related publications include:

  – Q. Zou, F. Magoulès, A new cyclic gradient method adapted to large-scale linear systems, in Proceedings of 17th DCABES, IEEE, 2018 (see [159]).

- We have developed the spectral properties of minimal gradient. We have proposed three new gradient methods with alignment and proved their convergence. These new methods are competitive with prior work and generate smoother convergence curves. Our related publications include:

  – Q. Zou, F. Magoulès, Fast gradient methods with alignment for symmetric linear systems without using Cauchy step, in progress.

- We have proposed several approaches to estimate the parameter in the Hermitian and skew-Hermitian splitting method based on gradient iterations. We have suggested to use lagged gradient methods as inner solvers. Our related publications include:

  - Q. Zou, F. Magoulès, Parameter estimation in HSS method using gradient iterations, in progress.

- We have proposed two new properties for s-dimensional gradient methods. We have proposed two new lagged s-dimensional methods and suggested to use cyclic formulations to reduce communication costs. We have provided parallel algorithms for these methods. Our related publications include:

  - Q. Zou, F. Magoulès, Reducing the effect of global synchronization in delayed gradient methods for symmetric linear systems, in progress.
  - Q. Zou, F. Magoulès, Parallel iterative methods with retards for linear systems, in Proceedings of 6th PARENG, Civil-Comp Press, 2019 (see [161])

During this thesis, we have also made the following contributions which shall not be discussed in the following sections:

- We have formalized the asynchronous Laplace transform process for time discretization. Our related publications include:

  - F. Magoulès, Q. Zou, Asynchronous time-parallel method based on Laplace transform, Int. J. Comput. Math., in review.

- We have implemented asynchronous convergence detection using a modified recursive doubling algorithm. Our related publications include:

  - Q. Zou, F. Magoulès, Convergence detection of asynchronous iterations based on modified recursive doubling, in Proceedings of 17th DCABES, IEEE, 2018 (see [160]).

- We have written a book about parallel programming of asynchronous iterations. In particular, this book explains how to use JACK (see [107]) to implement asynchronous iterative methods. Our related publications include:

  - G. Gbikpi-Benissan, Q. Zou, F. Magoulès, Asynchronous iterative methods: programming models and parallel implementation, Institute of Computer Science, 2018 (see [79]).

- We have applied asynchronous Parareal time discretization method (see [108]) to the Black-Scholes equation. Our related publications include:

  - F. Magoulès, G. Gbikpi-Benissan, Q. Zou, Asynchronous iterations of Parareal algorithm for option pricing models, Mathematics, 2018 (see [111]).

- Q. Zou, G. Gbikpi-Benissan, F. Magoulès, Asynchronous Parareal algorithm applied to European option pricing, in Proceedings of 16th DCABES, IEEE, 2017 (see [162]).
- Q. Zou, G. Gbikpi-Benissan, F. Magoulès, Asynchronous communications library for the parallel-in-time solution of Black-Scholes Equation, in Proceedings of 16th DCABES, IEEE, 2017 (see [163]).

Chapter 2

Gradient methods

In this chapter, we present a detailed survey of gradient iterative methods, convergence theories and asymptotic properties. Basic methods are monotone and sometimes optimal in terms of specific norms. Lagged methods often achieve better performance due to their nonmonotone behavior. Recent work suggests that spectral properties could be exploited to accelerate convergence. In the last section, we provide a new method with two-dimensional finite termination property.

2.1 Introduction

Consider the linear system

$$Ax = b, \tag{2.1}$$

where $A \in \mathbb{R}^{N\times N}$ is symmetric positive definite (SPD) and $b \in \mathbb{R}^N$. Let $\{v_1, \ldots, v_N\}$ be the orthonormal eigenvectors of A associated with the eigenvalues $\{\lambda_1, \ldots, \lambda_N\}$. We assume that

$$0 < \lambda_1 \leq \cdots \leq \lambda_N,$$

and thus the condition number is

$$\kappa = \frac{\lambda_N}{\lambda_1}.$$

The solution x_* is the unique global minimizer of strictly convex quadratic function

$$f(x) = \frac{1}{2}x^\mathsf{T}Ax - b^\mathsf{T}x. \tag{2.2}$$

For $n = 0, 1, \ldots$, the gradient method is of the form

$$x_{n+1} = x_n - \alpha_n g_n, \tag{2.3}$$

where $g_n = \nabla f(x_n) = Ax_n - b$. This gives the updating formula

$$g_{n+1} = g_n - \alpha_n A g_n. \tag{2.4}$$

The residual vector r_n is defined as $r_n = b - Ax_n$ and thus $r_n = -g_n$. The error iterates are of the form

$$e_{n+1} = (I - \alpha_n A)e_n, \tag{2.5}$$

where $e_n = x_* - x_n$. There exist real numbers $\xi_{i,n}$ and $\zeta_{i,n}$ such that

$$e_n = \sum_{i=1}^{N} \xi_{i,n} v_i, \quad g_n = \sum_{i=1}^{N} \zeta_{i,n} v_i.$$

Thus,

$$\xi_{i,n+1} = (1 - \alpha_n \lambda_i)\xi_{i,n}, \quad \zeta_{i,n+1} = (1 - \alpha_n \lambda_i)\zeta_{i,n},$$

and

$$\zeta_{i,n} = -\xi_{i,n}\lambda_i.$$

2.2 Basic gradient methods

The steepest descent (SD) method, proposed originally by Cauchy [33], defines the steplength by the reciprocal of the Rayleigh quotient

$$\alpha_n^{\mathrm{SD}} = \frac{g_n^{\mathsf{T}} g_n}{g_n^{\mathsf{T}} A g_n},$$

which is also called Cauchy steplength. It minimizes the quadratic function f or the A-norm error of the linear system and gives theoretically an optimal result in each step

$$\alpha_n^{\mathrm{SD}} = \arg\min_{\alpha} f(x_n - \alpha g_n) = \arg\min_{\alpha} \|(I - \alpha A)e_n\|_A^2 .$$

This classical method is known to behave badly in practice. The directions generated tend to asymptotically alternate between two orthogonal directions which leads to a low convergence rate [1].

The minimal gradient (MG) method was proposed by Krasnosel'skii and Krein [99] (see also [98]) which is of the form

$$\alpha_n^{\mathrm{MG}} = \frac{g_n^{\mathsf{T}} A g_n}{g_n^{\mathsf{T}} A^2 g_n}.$$

It minimizes the 2-norm gradient value

$$\alpha_n^{\mathrm{MG}} = \arg\min_{\alpha} \|g_n - \alpha A g_n\|^2.$$

Traditionally it does not have a specific name. From Kozjakin and Krasnosel'skii [98] we know that it was originally called "minimal residues". However, this term might cause confusion since there exists a Krylov subspace method called MINRES [118] which minimizes the norm of the residual through the Lanczos process. On the other hand, MG is also a special case of Orthomin(k) method when $k = 1$ [84]. Here the name "minimal gradient" comes from [50] since it gives an optimal gradient result in each step.

Dai and Yang [49] proposed a new gradient method of the form

$$\alpha_n^{\mathrm{AO}} = \frac{\|g_n\|}{\|A g_n\|}.$$

It asymptotically converges to the optimal steplength

$$\lim_{n\to\infty} \alpha_n^{\mathrm{AO}} = \alpha_n^{\mathrm{OPT}} = \frac{2}{\lambda_1 + \lambda_N},$$

which minimizes the coefficient matrix

$$\alpha_n^{\mathrm{OPT}} = \arg\min_{\alpha} \|I - \alpha A\|.$$

Thus we call it asymptotically optimal (AO) method.

In practice, SD usually performs better than MG, but both of them are badly effected by ill-conditioning. Moreover, AO performs generally as bad as SD and MG.

Based on the aforementioned steplengths, several variants have been investigated. Raydan and Svaiter [128] proposed the relaxed steepest descent (RSD) method as below

$$\alpha_n^{\mathrm{RSD}} = \theta_n \alpha_n^{\mathrm{SD}}, \quad \theta_n \in (0, 2),$$

where the sequence $\{\theta_n\}$ should have an accumulation point. They found that the introduction of relaxation might accelerate the convergence of the original SD method. In [50], similar formulations are considered by Dai and Yuan under the name of shortened SD. The alternate

minimization (AM) method which alternately minimizes the A-norm error value and the 2-norm gradient value was also proposed in [50], namely

$$\alpha_n^{\mathrm{AM}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & \text{for even } n, \\ \alpha_n^{\mathrm{MG}}, & \text{for odd } n. \end{cases}$$

Further, as mentioned in Ascher et al. [5], more combinations are possible such as the harmonic mean (HM) method

$$\alpha_n^{\mathrm{HM}} = 2\left(\frac{1}{\alpha_n^{\mathrm{SD}}} + \frac{1}{\alpha_n^{\mathrm{MG}}}\right)^{-1},$$

and the relaxed minimization (RM) method

$$\alpha_n^{\mathrm{RM}} = \omega_n \alpha_n^{\mathrm{SD}} + (1-\omega_n)\alpha_n^{\mathrm{MG}}, \quad \omega_n \in (0, 1).$$

All methods described above are monotone methods, i.e., the steplength α_n is such that $f(x_{n+1}) < f(x_n)$ for all n.

Finally, we make a slight digression for the conjugate gradient (CG) method [89]. According to the definition of gradient methods defined by Equation (2.3), obviously, CG does not belong to this camp. CG is often the method of choice that will terminate in at most N iterations theoretically. It is very attractive because of its high efficiency and low storage requirement. Nevertheless, when low precision is required, CG often performs not good [74]; if there exists a nonquadratic term in function f, other gradient methods can also become competitive [65, 143]; moreover, any derivation such as roundoff errors can seriously degrade its performance [47]. The purpose of this monograph is to investigate the properties of gradient methods and develop new methods. We mention Saad [131] for more theoretical arguments about CG and other Krylov subspace methods.

2.3 Fast gradient methods

The first lagged gradient method is the Barzilai-Borwein (BB) method that was originally proposed in Barzilai and Borwein [17]. The BB method is of the form

$$\alpha_n^{\mathrm{BB}} = \frac{g_{n-1}^{\mathsf{T}} g_{n-1}}{g_{n-1}^{\mathsf{T}} A g_{n-1}},$$

which remedies the ill-conditioning problem in Cauchy method by using nonmonotone steplength. The motivation arose in providing a two-point approximation to the quasi-Newton methods, namely

$$\alpha_n^{\mathrm{BB}} = \arg\min_{\alpha} \left\| \frac{1}{\alpha}\Delta x - \Delta g \right\|^2,$$

where $\Delta x = x_n - x_{n-1}$ and $\Delta g = g_n - g_{n-1}$. A similar method developed by symmetry in [17] is of the form

$$\alpha_n^{\mathrm{BB2}} = \frac{g_{n-1}^{\mathsf{T}} A g_{n-1}}{g_{n-1}^{\mathsf{T}} A^2 g_{n-1}},$$

which imposes as well a quasi-Newton property

$$\alpha_n^{\mathrm{BB2}} = \arg\min_{\alpha} \|\Delta x - \alpha \Delta g\|^2.$$

Notice that $\alpha_n^{\mathrm{BB}} = \alpha_{n-1}^{\mathrm{SD}}$ and $\alpha_n^{\mathrm{BB2}} = \alpha_{n-1}^{\mathrm{MG}}$. Practical experience is somewhat in favor of the first one. The convergence analysis of these methods was given in [125, 126, 48]. The preconditioned version was established in [116]. While we address mainly the problem of Equation (2.1) in this monograph, we note that the BB method has been successfully applied to the general unconstrained minimization problems by Raydan [127], in which a globalization strategy based on the line search condition of Grippo et al. [87] was introduced. A more recent chapter by Fletcher [65] discussed the efficiency of the BB method.

In the years that followed numerous generalizations have appeared. An ingenious framework called gradient method with retards (GMR) was introduced by Friedlander et al. [74]. Given m a positive integer and given $\{q_1, \ldots, q_m\}$ a set of positive numbers, let $\bar{n} = \max\{0,\ n-m\}$. Then the GMR framework is defined as follows

$$\alpha_n^{\mathrm{GMR}} = \frac{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)+1} g_{\tau(n)}}, \tag{2.6}$$

with

$$\tau(n) \in \{\bar{n}, \bar{n}+1, \ldots, n-1, n\}, \quad \rho(n) \in \{q_1, \ldots, q_m\}, \quad q_j \geq 0.$$

Several promising methods can be derived from such formulation. For example, the alternate step (AS) method [45] can be written as

$$\alpha_n^{\mathrm{AS}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & \text{for even } n, \\ \alpha_n^{\mathrm{BB}}, & \text{for odd } n. \end{cases}$$

In [74], the first cyclic gradient method under the name of cyclic steepest descent (CSD) was proposed, which can be summarized as follows

$$\alpha_n^{\mathrm{CSD}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod d = 0, \\ \alpha_{n-1}, & \text{otherwise}, \end{cases} \qquad d \geq 2.$$

Another important formulation called cyclic Barzilai-Borwein (CBB) was investigated in [45, 52], which is of the form

$$\alpha_n^{\mathrm{CBB}} = \begin{cases} \alpha_n^{\mathrm{BB}}, & n \bmod d = 0, \\ \alpha_{n-1}, & \text{otherwise}, \end{cases} \qquad d \geq 2.$$

On the other hand, given υ a positive number, a generalization of the GMR framework (DGMR) by Dai [45] covering the AO steplength was introduced as below

$$\alpha_n^{\mathrm{DGMR}} = \left(\frac{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)+\upsilon} g_{\tau(n)}} \right)^{\frac{1}{\upsilon}}. \tag{2.7}$$

Given $\rho_0 \in \mathbb{N}$, Yuan's generalization of the GMR framework (YGMR) [156] can be written as follows

$$\alpha_n^{\mathrm{YGMR}} = \left[\min_{|\rho| \leq \rho_0, \bar{n} \leq j \leq n} \frac{g_j^{\mathsf{T}} A^{\rho} g_j}{g_j^{\mathsf{T}} A^{\rho+1} g_j}, \max_{|\rho| \leq \rho_0, \bar{n} \leq j \leq n} \frac{g_j^{\mathsf{T}} A^{\rho} g_j}{g_j^{\mathsf{T}} A^{\rho+1} g_j} \right]. \tag{2.8}$$

To better choose the parameters and the appropriate steplengths for gradient methods, some investigations were conducted towards the adaptive techniques. Lamotte et al. [100] introduced an adaptive technique for the choice of the lagged parameter m in order to improve the stability. Zhou et al. [158] proposed two adaptive methods based on SD and BB respectively. The adaptive steepest descent (ASD) method is of the form

$$\alpha_n^{\mathrm{ASD}} = \begin{cases} \alpha_n^{\mathrm{MG}}, & \alpha_n^{\mathrm{MG}} > \theta \alpha_n^{\mathrm{SD}}, \\ \alpha_n^{\mathrm{SD}} - \omega \alpha_n^{\mathrm{MG}}, & \text{otherwise}, \end{cases} \qquad \theta, \omega \in (0, 1).$$

Analogously, the adaptive Barzilai-Borwein (ABB) method is of the form

$$\alpha_n^{\mathrm{ABB}} = \begin{cases} \alpha_n^{\mathrm{BB2}}, & \alpha_n^{\mathrm{BB2}} < \theta \alpha_n^{\mathrm{BB}}, \\ \alpha_n^{\mathrm{BB}}, & \text{otherwise}, \end{cases} \qquad \theta \in (0,1).$$

Since ABB overcomes already the stagnation problem of SD, the relaxation term ω disappears in the latter equation. According to the experimental results illustrated in [158], it is better to choose the parameter θ slightly bigger than 0.5 for ASD, while slightly smaller than 0.5 for ABB. Moreover, ABB performs better for very ill-conditioned problems and both of them are comparable and generally preferable to other lagged gradient methods. Frassoldati et al. [70] proposed two adaptive methods in terms of the shortened BB steplength. One of them can be written as follows

$$\alpha_n^{\mathrm{MABB}} = \begin{cases} \min\left\{ \alpha_j^{\mathrm{BB2}} \mid j = \max\{0, n-d\}, \ldots, n \right\}, & \alpha_n^{\mathrm{BB2}} < \theta \alpha_n^{\mathrm{BB}}, \\ \alpha_n^{\mathrm{BB}}, & \text{otherwise}, \end{cases} \qquad \theta \in (0,1),$$

which belongs to the GMR framework as shown in [74]. It was denoted by $\mathrm{ABB}_{\mathrm{min}\,1}$ in [70]. Here we call it modified adaptive Barzilai-Borwein (MABB) method. In [156] Yuan discussed a similar formulation using only the short BB2 step. There exists another more complicated derivation which reveals the spectral properties of the coefficient matrix, see [70] for details.

Besides, there exist several auxiliary steplengths acting as accelerators of other methods. More precisely, we can conduct occasionally the auxiliary iterative steps to improve the global performance. For example, in order to find the unique minimizer in finitely many iterations in 2-dimensions, Yuan [154] proposed a ingenious steplength as follows

$$\alpha_n^{\mathrm{Y}} = 2 \left(\sqrt{ \left(\frac{1}{\alpha_{n-1}^{\mathrm{SD}}} - \frac{1}{\alpha_n^{\mathrm{SD}}} \right)^2 + \frac{4 \|g_n\|^2}{\left(\alpha_{n-1}^{\mathrm{SD}}\right)^2 \|g_{n-1}\|^2} } + \frac{1}{\alpha_{n-1}^{\mathrm{SD}}} + \frac{1}{\alpha_n^{\mathrm{SD}}} \right)^{-1},$$

which is called Yuan steplength. It is worth mentioning that such steplength has been proved to be much significant in terms of the underlying alignment property. Recently, a paper by De Asmundis et al. [56] (see also [55]) proposed a gradient method that exploits the spectral properties for choosing steps. The improvement resorts to the alignment of the gradient

vector involving a special steplength

$$\alpha_n^{\mathrm{A}} = \left(\frac{1}{\alpha_{n-1}^{\mathrm{SD}}} + \frac{1}{\alpha_n^{\mathrm{SD}}} \right)^{-1},$$

and sometimes the double Cauchy steplength for the sake of monotonicity

$$\alpha_n^{\mathrm{D}} = 2\alpha_n^{\mathrm{SD}} \tag{2.9}$$

In one direction, these steplengths lead to some efficient alternate gradient methods, especially the second variant provided in [51]

$$\alpha_n^{\mathrm{DY}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod 4 = 0 \text{ or } 1, \\ \alpha_n^{\mathrm{Y}}, & \text{otherwise}, \end{cases} \tag{2.10}$$

which was shown as the most efficient variant according to the experiments. As usual, it does not have a specific name. Here we call it Dai-Yuan (DY) method (see, e.g., [70]). A closer examination of Yuan variants revealed that they have a distinguish property which is called "decreasing together" [51]. It means that DY does not sink into any lower subspace spanned by eigenvectors. BB has also such feature. Important differences come from the fact that BB is a nonmonotone steplength, whereas DY is monotone thus being more stable.

On the other hand, the auxiliary steps lead to the gradient methods with alignment such as

$$\alpha_n^{\mathrm{SDA}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod (d_1 + d_2) < d_1, \\ \alpha_n^{\mathrm{A}}, & n \bmod (d_1 + d_2) = d_1, \quad d_1, d_2 \geq 1. \\ \alpha_{n-1}^{\mathrm{SDA}}, & \text{otherwise}, \end{cases}$$

This method is called steepest descent with alignment (SDA). Here, we choose the version described in [58] without using the original switch condition, and vary the form while leaving the alignment property unchanged. The main feature of this method is to foster the reduction of gradient components along the eigenvectors of A selectively, and reduce the search space into smaller and smaller dimensions. The problem tends to have a better and better condition number [57]. Shortly after, they presented another similar method based on Yuan steplength [57], called steepest descent with constant steplength (SDC), which is of the

form

$$\alpha_n^{\mathrm{SDC}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod (d_1+d_2) < d_1, \\ \alpha_n^{\mathrm{Y}}, & n \bmod (d_1+d_2) = d_1, \quad d_1, d_2 \geq 1. \\ \alpha_{n-1}^{\mathrm{SDC}}, & \text{otherwise}, \end{cases}$$

They also provided the monotone versions by using the double SD steplength. The first one called SDA with monotonicity (SDAM) can be written as follows

$$\alpha_n^{\mathrm{SDAM}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod (d_1+d_2) < d_1, \\ \alpha_n^{\mathrm{A}}, & n \bmod (d_1+d_2) = d_1, \quad d_1, d_2 \geq 1. \\ \min\left\{\alpha_{n-1}^{\mathrm{SDAM}}, \alpha_n^{\mathrm{D}}\right\}, & \text{otherwise}, \end{cases}$$

In that way, the second one is called SDC with monotonicity (SDCM)

$$\alpha_n^{\mathrm{SDCM}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod (d_1+d_2) < d_1, \\ \alpha_n^{\mathrm{Y}}, & n \bmod (d_1+d_2) = d_1, \quad d_1, d_2 \geq 1. \\ \min\left\{\alpha_{n-1}^{\mathrm{SDCM}}, \alpha_n^{\mathrm{D}}\right\}, & \text{otherwise}, \end{cases}$$

Recall that the names and the notations are adjusted to better suit the following deductions. We keep the core features in the aforementioned methods unchanged. These two methods seem to be the state of the art of gradient methods and tend to give the best performance among all of these. Recently, Gonzaga and Schneider [83] introduced a general framework of Cauchy steplength with alignment, which breaks the Cauchy cycle by periodically applying some short steplengths. Fletcher [66] proposed a new method called limited memory steepest descent (LMSD) method. Convergence analysis has been discussed in [44].

2.4 Convergence theories

By the invariance property under any orthogonal transformation, we can give the following assumption without loss of generality.

Assumption 2.1. $A = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ *with* $\lambda_1 \leq \cdots \leq \lambda_N$ *and* $\lambda_1 = 1$.

Remark. This assumption seems to be quite strict in practice. However, for the theoretical analysis, we could simply add an orthogonal transformation that transforms A to a diagonal

matrix of eigenvalues. Moreover, if $\lambda_1 \neq 1$, we could add a factor $1/\lambda_1$ to the matrix without changing the convergence property. Hence, we make this assumption in some situations for the sake of convergence analysis exclusively.

Lemma 2.1 (Kantorovich inequality)**.** *Let B be a symmetric positive definite matrix. Let λ_{max} and λ_{min} be its largest and smallest eigenvalues. Then*

$$\frac{(u^\intercal Bu)\left(u^\intercal B^{-1}u\right)}{(u^\intercal u)^2} \leq \frac{(\lambda_{max}+\lambda_{min})^2}{4\lambda_{max}\lambda_{min}}, \quad \forall u \neq 0.$$

Proof. See Saad [131]. □

Theorem 2.2. *Let A be a symmetric positive definite matrix. Consider the SD method being used to solve the linear system $Ax = b$. Then*

$$\|e_{n+1}\|_A \leq \frac{\kappa-1}{\kappa+1}\|e_n\|_A .$$

Proof. See Saad [131]. □

Theorem 2.3 (Nocedal et al., 2002, Theorem 4.1)**.** *Let A be a symmetric positive definite matrix. Consider the SD method being used to solve the linear system $Ax = b$. Then*

$$\|g_{n+1}\| \leq \frac{\kappa-1}{2\sqrt{\kappa}}\|g_n\| .$$

Proof. See Nocedal et al. [117]. □

Theorem 2.4. *Let A be a positive definite matrix. Let λ_{min} be the smallest eigenvalue of $(A+A^\intercal)/2$. Consider the MG method being used to solve the linear system $Ax = b$. Then*

$$\|g_{n+1}\| \leq \sqrt{1-\frac{\lambda_{\min}^2}{\|A\|^2}}\|g_n\| .$$

Proof. See Saad [131]. □

Theorem 2.5. *Let A be a symmetric positive definite matrix. Consider the MG method being used to solve the linear system $Ax = b$. Then*

$$\|g_{n+1}\| \leq \frac{\kappa-1}{\kappa+1}\|g_n\| .$$

Proof. See Greenbaum [84]. □

Theorem 2.6 (Friedlander et al., 1999, Theorem 2.1)**.** *Let A be a symmetric positive definite matrix. Consider the gradient method with steplength* (2.6) *being used to solve the linear system $Ax = b$. Then the sequence $\{x_n\}$ converges to x_*.*

Proof. See Friedlander et al. [74]. □

Theorem 2.7 (Raydan, 2002, Theorem 2.1)**.** *Consider the linear system $Ax = b$ where A is symmetric positive definite. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the RSD method. If the sequence $\{\theta_n\}$ has an accumulation point $\theta_* \in (0, 2)$, then the sequence $\{x_n\}$ converges to x_*.*

Proof. See Raydan and Svaiter [128]. □

Definition (R-linear convergence)**.** Let $\{u_n\}$ be a sequence of vectors which converges to u_*. If there exists a sequence $\{\varepsilon_n\}$ such that $\|u_n - u_*\| \leq \varepsilon_n$ for all n and there exists a constant c such that

$$\lim_{n\to\infty} \frac{|\varepsilon_{n+1}|}{|\varepsilon_n|} = c,$$

then u_n converges R-linearly to u_*.

Definition (Property A)**.** Let $g_{i,n}$ be the ith component of g_n and

$$G(n, \mu) = \sum_{i=1}^{\mu} g_{i,n}^2.$$

Suppose that Assumption 2.1 holds. If there exist constant c and $\exists m_0 \in \mathbb{N}$, $\exists c_1, c_2 > 0$, such that $\forall \mu \in \{1, \ldots, N-1\}$, $\forall \varepsilon > 0$, $\forall j \in \{0, \ldots, \min\{n, m_0\}\}$,

1. $\lambda_1 \leq \alpha_n^{-1} \leq c_1$;
2. if $G(n-j, \mu) \leq \varepsilon$ and $g_{\mu+1,n-j}^2 \geq c_2\varepsilon$, then $\alpha_n^{-1} \geq \frac{2}{3}\lambda_{\mu+1}$,

then the steplength α_n has Property A.

Lemma 2.8 (Dai, 2003, Theorem 4.1)**.** *Consider the linear system $Ax = b$ with Assumption 2.1 holds. If the steplength α_n has Property A, then the sequence $\{\|g_n\|\}$ generated by a gradient method converges to zero R-linearly.*

Proof. See Dai [45]. □

Theorem 2.9 (Dai, 2003, Page 408)**.** *Let A be a symmetric positive definite matrix. Consider the gradient method with steplength* (2.7) *being used to solve the linear system $Ax = b$. Then the sequence $\{x_n\}$ converges to x_*.*

Proof. See Dai [45]. □

Theorem 2.10 (Yuan, 2010, Theorem 7.2.2)**.** *Let A be a symmetric positive definite matrix. Consider the gradient method with steplength* (2.8) *being used to solve the linear system $Ax = b$. Then the sequence $\{x_n\}$ converges to x_*.*

Proof. See Yuan [156]. □

2.5 Asymptotic properties

Assumption 2.2. *The matrix A is symmetric positive definite satisfying $0 < \lambda_1 < \cdots < \lambda_N$ and the starting point x_0 is such that $\zeta_{1,0} \neq 0$ and $\zeta_{N,0} \neq 0$.*

Remark. We point out that this assumption is not restrictive since if there exist some repeated eigenvalues, then we can choose the corresponding eigenvectors so that the superfluous ones vanish (see, e.g., Fletcher [65]). Moreover, if $\zeta_{1,0}$ or $\zeta_{N,0}$ equals zero, then the second condition can be simply replaced by the components involving inner indices without changing the results discussed later on.

Lemma 2.11 (Akaike, 1959, Theorems 1 and 2)**.** *Let p_0 be a probability measure attached to $\{\lambda_1, \ldots, \lambda_N\}$ where $p_{i,0} = p_0(\lambda_i)$ and $0 < \lambda_1 < \cdots < \lambda_N$. Consider a transformation such that*

$$p_{i,n+1} = \frac{\left(\sum_{j=1}^N \lambda_j p_{j,n} - \lambda_i\right)^2}{\sum_{l=1}^N \left(\sum_{j=1}^N \lambda_j p_{j,n} - \lambda_l\right)^2 p_{l,n}} p_{i,n}.$$

Then,

$$\lim_{n\to\infty} p_{i,2n} = \begin{cases} p_*, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ 1 - p_*, & i = N, \end{cases}$$

and

$$\lim_{n\to\infty} p_{i,2n+1} = \begin{cases} 1 - p_*, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ p_*, & i = N, \end{cases}$$

for some $p_* \in (0, 1)$.

Proof. See Akaike [1], see also [68, 124]. □

Theorem 2.12 (Akaike, 1959, Theorem 4)**.** *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the SD method. Then*

$$\lim_{n\to\infty} \frac{\zeta_{i,2n}^2}{\sum_{j=1}^N \zeta_{j,2n}^2} = \begin{cases} \frac{1}{1+c^2}, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ \frac{c^2}{1+c^2}, & i = N, \end{cases}$$

and

$$\lim_{n\to\infty} \frac{\zeta_{i,2n+1}^2}{\sum_{j=1}^N \zeta_{j,2n+1}^2} = \begin{cases} \frac{c^2}{1+c^2}, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ \frac{1}{1+c^2}, & i = N, \end{cases}$$

for some constant c. Moreover, $\zeta_{1,2n}$, $\zeta_{N,2n}$, $\zeta_{1,2n+1}$, $\zeta_{N,2n+1}$ have fixed signs for large n.

Proof. See Akaike [1], see also [68]. □

Theorem 2.13 (Akaike, 1959, Page 11)**.** *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the SD method. Then*

$$\lim_{n\to\infty} \frac{f(x_{n+1}) - f(x_*)}{f(x_n) - f(x_*)} = \frac{c^2(\kappa-1)^2}{(c^2+\kappa)(1+c^2\kappa)},$$

for some constant c.

Proof. See Akaike [1]. □

Theorem 2.14 (Nocedal et al., 2002, Lemma 3.3)**.** *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the SD method. Then*

$$\lim_{n\to\infty} \alpha_{2n}^{\mathrm{SD}} = \frac{1+c^2}{\lambda_1(1+c^2\kappa)},$$

and

$$\lim_{n\to\infty} \alpha_{2n+1}^{\mathrm{SD}} = \frac{1+c^2}{\lambda_1(c^2+\kappa)},$$

for some constant c.

Proof. See Nocedal et al. [117]. □

Theorem 2.15 (Nocedal et al., 2002, Theorem 5.1)**.** *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the*

SD method. Then

$$\lim_{n\to\infty}\frac{\|g_{2n+1}\|^2}{\|g_{2n}\|^2}=\frac{c^2(\kappa-1)^2}{(1+c^2\kappa)^2},$$

$$\lim_{n\to\infty}\frac{\|g_{2n+2}\|^2}{\|g_{2n+1}\|^2}=\frac{c^2(\kappa-1)^2}{(c^2+\kappa)^2},$$

and

$$\lim_{n\to\infty}\frac{\|g_{n+2}\|^2}{\|g_n\|^2}=\lim_{n\to\infty}\frac{f(x_{n+1})-f(x_*)}{f(x_n)-f(x_*)}=\frac{c^2(\kappa-1)^2}{(c^2+\kappa)(1+c^2\kappa)},$$

for some constant c.

Proof. See Nocedal et al. [117]. □

Theorem 2.16 (Nocedal et al., 2002, Theorem 5.2)**.** *The constant c in theorems 2.12, 2.13, 2.14 and 2.15 satisfies the following properties*

1. *c is given by the limits*

$$c=\lim_{n\to\infty}\frac{\zeta_{N,2n}}{\zeta_{1,2n}}=-\lim_{k\to\infty}\frac{\zeta_{1,2n+1}}{\zeta_{N,2n+1}};$$

2. *c is uniquely determined by the starting point x_0 and by the eigenvalues and the eigenvectors of A;*

3. *if the set*

$$\mathscr{I}=\left\{i=2,\ldots,N-1 \mid v_i^{\intercal}g_0\neq 0,\ \lambda_i\neq\frac{1}{\alpha_n},\ \forall n\geq 0\right\}$$

is nonempty, and

$$\varphi_\delta^{-1}\leq c^2\leq\varphi_\delta,$$

where

$$\varphi_\delta=\frac{2+\eta_\delta+\sqrt{\eta_\delta^2+4\eta_\delta}}{2},$$

with

$$\eta_\delta=4\left(\frac{1+\delta^2}{1-\delta^2}\right),\quad \delta=\min_{i\in\mathscr{I}}\left|\frac{\lambda_i-\frac{\lambda_N+\lambda_1}{2}}{\frac{\lambda_N-\lambda_1}{2}}\right|.$$

Proof. See Nocedal et al. [117]. □

Theorem 2.17 (Dai and Yang, 2006, Theorem 2.1)**.** *Consider the linear system $Ax=b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the*

AO method. Then

$$\lim_{n\to\infty} \alpha_n^{\mathrm{AO}} = \frac{2}{\lambda_1 + \lambda_N}.$$

Proof. See Dai and Yang [49]. □

Theorem 2.18 (De Asmundis et al., 2013, theorem 3.1)**.** *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the SD method. Then*

$$\lim_{n\to\infty} \alpha_n^{\mathrm{A}} = \frac{1}{\lambda_1 + \lambda_N}.$$

Proof. See De Asmundis et al. [56]. □

Theorem 2.19 (De Asmundis et al., 2013, theorem 3.2)**.** *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the gradient method with constant steplength*

$$\hat{\alpha} = \frac{1}{\lambda_1 + \lambda_n}.$$

Then the sequence $\{x_n\}$ converges to x_* for any starting point x_0. Moreover,

$$\lim_{n\to\infty} \frac{\zeta_{i,n}}{\zeta_{1,n}} = 0, \quad i = 2, 3, \ldots, N.$$

Proof. See De Asmundis et al. [56]. □

Theorem 2.20 (De Asmundis et al., 2013, theorem 4.1)**.** *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the gradient method with steplength* (2.9)*. Then,*

$$\lim_{n\to\infty} \frac{g_{n+1}}{\prod_{j=0}^{n}(1 - \alpha_j \lambda_N)} = \zeta_{N,0} v_N,$$

$$\lim_{n\to\infty} \alpha_n^{\mathrm{D}} = \frac{2}{\lambda_N},$$

$$\lim_{n\to\infty} \nabla f(x_n - \alpha_n^{SD} g_n) = 0.$$

Proof. See De Asmundis et al. [56]. □

Theorem 2.21 (De Asmundis et al., 2014, Theorem 3.1)**.** *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated*

by the SD method. Then

$$\lim_{n\to\infty} \alpha_n^{\mathrm{Y}} = \frac{1}{\lambda_N}.$$

Moreover,

$$\lim_{n\to\infty} \left(\frac{1}{\alpha_{n-1}^{\mathrm{SD}} \alpha_n^{\mathrm{SD}}} - \frac{\|g_n\|^2}{\left(\alpha_{n-1}^{\mathrm{SD}}\right)^2 \|g_{n-1}\|^2} \right) = \lambda_1 \lambda_N.$$

Proof. See De Asmundis et al. [57]. □

2.6 First trial: two-dimensional finite termination

In this part, we show a simple way to extend the existing methods, which can lay the foundation for the following chapters, see also [159]. We take the two-dimensional finite termination property as an example since the pioneering work of Yuan step did not expect that the revealing of second order information could result in alignment methods. In contrast, the motivation was to develop an iterative method that could ensure convergence for a 2-by-2 SPD system within certain iterations.

Some literatures showed that Yuan steplength may lead to efficient algorithms [154, 51], in which all methods have two-dimensional finite termination property. For example, if (2.10) is applied to a linear system in two-dimensional space, then the algorithm will terminate in at most 3 iterations. This property seems to be useless in practice. However, experience shows that they often perform well in higher dimensions and are competitive with BB methods for large-scale problems [51].

Inspired by the Yuan steplength, we suggest a simple way of modifying SD to a cyclic gradient method. Consider a steplength of the form

$$\alpha_n^{\mathrm{YB}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod 3 = 0 \text{ or } 2, \\ \alpha_n^{\mathrm{Y}}, & n \bmod 3 = 1. \end{cases} \tag{2.11}$$

Here we have modified the order of steps compared to the original YB formula shown in [154]. It keeps the two-dimensional finite termination property and performs as well as BB for large-scale problems and better for small-scale ones. Given a positive number d, we could add easily the cyclic behavior at the end of (2.11):

$$\text{if } n \bmod (3+d) > 2, \text{ then } \alpha_n = \alpha_{n-1}.$$

Additionally, we find that De Asmundis et al. [56] provides an interesting view about SD iterations, in which the alignment technique was proposed to force the gradients into one-dimensional subspace and avoid the zigzag pattern. Notice that the inverse of constant Rayleigh quotient such as SD and BB steplengths has a chance to share similar property (For details, see Theorem 3.7. Here we just give an example and will not pursue this property further). Constant SD with retards may sometimes leads to alignment behavior and keeps the nonmonotone benefit. On the other hand, we try to maintain the Yuan process in the first several iterations. These motivations lead to a new method of the form

$$\alpha_n^{\mathrm{CY}} = \begin{cases} \alpha_n^{\mathrm{Y}}, & n \bmod (d_1+d_2+2) = 1, \\ \alpha_n^{\mathrm{SD}}, & n \bmod (d_1+d_2+2) < d_1+2, \\ \alpha_{n-1}, & \text{otherwise}, \end{cases} \tag{2.12}$$

where $d_1, d_2 \geq 1$. This formula seems to be complex, but indeed easy to understand. There are three components in (2.12): the first SD and Y are used to insure the finite termination property; the parameter d_1 acting on the second part of SD is used to keep several zigzag iterations; finally, the lagged term d_2 yields occasionally alignment behavior and provides nonmonotone steps to leap from the lower subspace. The "magic number" d_1+d_2+2 comes from the fact that d_1 is the repeated times of SD, d_2 is the repeated times of SD with retards, and finally "2" represents we have two steps in the beginning: SD and Y.

We first provide the convergence result of CY in two-dimensional case, which can be presented as follows.

Theorem 2.22. *Consider the CY method being used to solve* (2.1) *where A is of size* 2. *Then there exists* $n \leq 3$ *such that the sequence* $\{x_n\}$ *converges to* x_*.

The proof is similar to the other two-dimensional finite termination methods so that we do not address again, see [154] for more details. Now we study the convergence in any dimensions. By the invariance property under any orthogonal transformation, we can assume without loss of generality that A is a diagonal matrix. We follow the convergence framework established by Dai [45], as shown in Theorem 2.8. The resulting theorem is given as follows.

Theorem 2.23. *Consider the CY method being used to solve* (2.1). *Then the sequence* $\{x_n\}$ *converges to* x_*.

Proof. Note that (2.12) has three alternate steplengths, whereas the SD updating process and the constant process both follow the framework (2.6), which has been proven to satisfy Property A in [45]. Therefore, we only discuss the Yuan steplength.

Recall that Yuan steplength has the following property

$$\left(\frac{1}{\alpha_{n-1}^{\mathrm{SD}}}+\frac{1}{\alpha_n^{\mathrm{SD}}}\right)^{-1}<\alpha_n^{\mathrm{Y}}<\min\left\{\alpha_{n-1}^{\mathrm{SD}},\,\alpha_n^{\mathrm{SD}}\right\},$$

which was given in [154]. It follows that

$$\lambda_1\leq\frac{1}{\alpha_n^{\mathrm{SD}}}<\frac{1}{\alpha_n^{\mathrm{Y}}}<\frac{1}{\alpha_{n-1}^{\mathrm{SD}}}+\frac{1}{\alpha_n^{\mathrm{SD}}}\leq 2\lambda_N.$$

Hence, the first condition of Property A follows by setting $c_1=2\lambda_N$. For the second one, let $c_2=2$ and $m_0=1$. It follows that $j=0$. Assume that

$$G(n,\mu)\leq\varepsilon,\quad g_{\mu+1,n}^2\geq c_2\varepsilon,$$

for all $\mu\in\{1,\ldots,N-1\}$, and $\varepsilon>0$. Let $g_{i,n}$ be the ith component in nth iteration. The inverse of Yuan steplength becomes

$$\begin{aligned}\frac{1}{\alpha_n^{\mathrm{Y}}}>\frac{1}{\alpha_n^{\mathrm{SD}}}&=\frac{\sum_{i=1}^N\lambda_i g_{i,n}^2}{\sum_{i=1}^N g_{i,n}^2}\geq\frac{\lambda_{\mu+1}\sum_{i=\mu+1}^N g_{i,n}^2}{\sum_{i=1}^{\mu}g_{i,n}^2+\sum_{i=\mu+1}^N g_{i,n}^2}\\&\geq\frac{\lambda_{\mu+1}}{\frac{\sum_{i=1}^{\mu}g_{i,n}^2}{g_{\mu+1,n}^2}+1}\geq\frac{\lambda_{\mu+1}}{\frac{\varepsilon}{2\varepsilon}+1}=\frac{2}{3}\lambda_{\mu+1}.\end{aligned}$$

This completes the proof. □

We conduct an experiment with a well-conditioned problem provided by the University of Florida Sparse Matrix Collection [53], where $N=5\times10^4$ and the number of non-zero values is 349968. All parameters are chosen under a training problem given in [17], such that $d_1=4, d_2=3$ for CY, $d=3$ for CSD, and $d=4$ for CBB. The right-hand side b is generated with random components in $[-10,10]$. The residual threshold is defined as $\varepsilon=\|b-Ax_n\|/\|b\|$. Results are shown in Table 2.1. We use bold numbers to indicate the most efficient algorithms under each threshold. Backslash depicts slow convergence in which more than 10^4 iterations are needed. From Table 2.1, we can see that the CY method performs better than other methods except CG. CY is less efficient than CG in the highest precision, but it is competitive in other situations.

Here we take CY as an example to illustrate two-dimensional finite termination methods. Table 2.1 is only a specific case in favor of the new method. There exist other cases that could lead to opposite results, depending on the distribution of eigenvalues and the elements of

Table 2.1 Gradient methods with different residual thresholds for a well-conditioned problem.

	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
CG	58	735	2617	**4251**	**5786**	**7535**
CY	**13**	**208**	**1153**	**4275**	**6181**	\
CSD	27	212	**1470**	4357	6713	\
CBB	31	241	1965	7534	\	\
DY	**15**	**200**	1595	6415	\	\
SD	61	5773	\	\	\	\

right-hand side. Hence, we conclude that CY is competitive with CG in a few special cases and better than some other nonmonotone gradient methods for large-scale problems [159]. Nevertheless, our experience is that it is still less efficient than alignment methods and seems to be as oscillating as BB.

Chapter 3

Gradient methods with alignment

The performance of gradient methods has been considerably improved by the introduction of delayed parameters. After two and a half decades, the revealing of second-order information within the Hessian matrix has recently given rise to the Cauchy-based methods with alignment. They reduce asymptotically the search spaces in smaller and smaller dimensions and are considered as the state of the art of gradient methods. This chapter reveals the spectral properties of minimal gradient and asymptotically optimal steps, and then suggests three new methods with alignment without using the Cauchy steplength. The convergence results are provided, and numerical experiments show that the new methods provide competitive and more stable alternatives to the classical Cauchy-based methods. In particular, alignment gradient methods present advantages over the Krylov subspace methods in some situations, which makes them attractive in practice.

3.1 Spectral analysis of minimal gradient

Recall that the minimal gradient (MG) method is of the form

$$\alpha_n^{\mathrm{MG}} = \frac{g_n^{\mathsf{T}} A g_n}{g_n^{\mathsf{T}} A^2 g_n}.$$

It minimizes the 2-norm gradient value

$$\alpha_n^{\mathrm{MG}} = \arg\min_{\alpha} \|g_n - \alpha A g_n\|^2,$$

where $\|\cdot\|$ denotes the Euclidean norm of a vector.

We know from [1] that the SD method is asymptotically reduced to a search in the 2-dimensional subspace generated by the two eigenvectors corresponding to the largest

and the smallest eigenvalues of A. Eventually the directions generated tend to zigzag in two orthogonal directions that gives rise to a slow convergence rate. Such argument was demonstrated by using Theorem 2.12, see Chapter 2 for more details.

We now give our main result on the spectral properties of MG. These arguments lead to the gradient methods with alignment which shall be described in Section 3.2. Theorem 3.1 has been proved in [124] for a framework called P-gradient algorithms, while Theorems 3.2 to 3.6 for the MG method have not appeared in any literature.

Theorem 3.1. *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the MG method. Then*

$$\lim_{n\to\infty} \frac{\lambda_i \zeta_{i,2n}^2}{\sum_{j=1}^N \lambda_j \zeta_{j,2n}^2} = \begin{cases} \frac{1}{1+c^2}, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ \frac{c^2}{1+c^2}, & i = N, \end{cases}$$

and

$$\lim_{n\to\infty} \frac{\lambda_i \zeta_{i,2n+1}^2}{\sum_{j=1}^N \lambda_j \zeta_{j,2n+1}^2} = \begin{cases} \frac{c^2}{1+c^2}, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ \frac{1}{1+c^2}, & i = N, \end{cases}$$

where c is a fixed but unknown number. Moreover, $\zeta_{1,2n}$, $\zeta_{N,2n}$, $\zeta_{1,2n+1}$, $\zeta_{N,2n+1}$ have fixed signs for large n.

Proof. Since

$$\zeta_{i,n+1} = \left(1 - \alpha_n^{\mathrm{MG}} \lambda_i\right) \zeta_{i,n},$$

il follows that

$$\zeta_{i,n+1} = \left(1 - \frac{\sum_{j=1}^N \lambda_j \zeta_{j,n}^2}{\sum_{j=1}^N \lambda_j^2 \zeta_{j,n}^2} \lambda_i\right) \zeta_{i,n}.$$

For any i and n, let us write $\hat{p}_{i,n} = \lambda_i \zeta_{i,n}^2$, it follows that

$$\hat{p}_{i,n+1} = \left(1 - \frac{\sum_{j=1}^N \hat{p}_{j,n}}{\sum_{j=1}^N \lambda_j \hat{p}_{j,n}} \lambda_i\right)^2 \hat{p}_{i,n}.$$

Moreover, we define a probability measure

$$p_{i,n} = \frac{\hat{p}_{i,n}}{\sum_{j=1}^{N} \hat{p}_{j,n}},$$

from which we notice that $\sum_{i=1}^{N} p_{i,n} = 1$. Hence,

$$\begin{aligned} p_{i,n+1} &= \left(\frac{\sum_{j=1}^{N} \lambda_j p_{j,n} - \lambda_i}{\sum_{j=1}^{N} \lambda_j p_{j,n}} \right)^2 \frac{\hat{p}_{i,n}}{\sum_{l=1}^{N} \hat{p}_{l,n+1}} \\ &= \frac{\left(\sum_{j=1}^{N} \lambda_j p_{j,n} - \lambda_i \right)^2}{\sum_{l=1}^{N} \left(\sum_{j=1}^{N} \lambda_j p_{j,n} - \lambda_l \right)^2 \hat{p}_{l,n}} \hat{p}_{i,n} \\ &= \frac{\left(\sum_{j=1}^{N} \lambda_j p_{j,n} - \lambda_i \right)^2}{\sum_{l=1}^{N} \left(\sum_{j=1}^{N} \lambda_j p_{j,n} - \lambda_l \right)^2 p_{l,n}} p_{i,n}, \end{aligned}$$

Along with Lemma 2.11 the desired result follows. □

The next theorem reveals the asymptotic behavior of quadratic function (2.2).

Theorem 3.2. *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the MG method. Then*

$$\lim_{n\to\infty} \frac{f(x_{2n+1}) - f(x_*)}{f(x_{2n}) - f(x_*)} = \frac{c^2(1+c^2\kappa^2)(\kappa-1)^2}{(c^2+\kappa^2)(1+c^2\kappa)^2},$$

and

$$\lim_{n\to\infty} \frac{f(x_{2n+2}) - f(x_*)}{f(x_{2n+1}) - f(x_*)} = \frac{c^2(c^2+\kappa^2)(\kappa-1)^2}{(1+c^2\kappa^2)(c^2+\kappa)^2},$$

where c has the same value as in Theorem 3.1.

Proof. For any n, it follows from (2.2) that

$$\begin{aligned} \frac{f(x_{n+1}) - f(x_*)}{f(x_n) - f(x_*)} &= \frac{g_n^{\mathsf{T}} A^{-1} g_n + \left(\alpha_n^{\mathrm{MG}}\right)^2 g_n^{\mathsf{T}} A g_n - 2\alpha_n^{\mathrm{MG}} g_n^{\mathsf{T}} g_n}{g_n^{\mathsf{T}} A^{-1} g_n} \\ &= 1 + \frac{\left(g_n^{\mathsf{T}} A g_n\right)^3}{\left(g_n^{\mathsf{T}} A^{-1} g_n\right)\left(g_n^{\mathsf{T}} A^2 g_n\right)^2} - \frac{2\left(g_n^{\mathsf{T}} A g_n\right)\left(g_n^{\mathsf{T}} g_n\right)}{\left(g_n^{\mathsf{T}} A^2 g_n\right)\left(g_n^{\mathsf{T}} A^{-1} g_n\right)}. \end{aligned}$$

Let us write

$$p_{i,n} = \frac{\lambda_i \zeta_{i,n}^2}{\sum_{j=1}^N \lambda_j \zeta_{j,n}^2},$$

which yields

$$\frac{f(x_{n+1}) - f(x_*)}{f(x_n) - f(x_*)} = 1 + \frac{1}{\left(\sum_{j=1}^N \lambda_j^{-2} p_{j,n}\right)\left(\sum_{j=1}^N \lambda_j p_{j,n}\right)^2} - \frac{2\sum_{j=1}^N \lambda_j^{-1} p_{j,n}}{\left(\sum_{j=1}^N \lambda_j^{-2} p_{j,n}\right)\left(\sum_{j=1}^N \lambda_j p_{j,n}\right)}.$$

If n is an even number, from theorem 3.1, one finds that

$$\begin{aligned}
\lim_{n\to\infty} \frac{f(x_{n+1}) - f(x_*)}{f(x_n) - f(x_*)} &= 1 + \frac{1}{\left(\frac{\kappa^2+c^2}{1+c^2}\right)\left(\frac{1+\kappa c^2}{\kappa(1+c^2)}\right)^2} - \frac{2\left(\frac{\kappa+c^2}{1+c^2}\right)}{\left(\frac{\kappa^2+c^2}{1+c^2}\right)\left(\frac{1+\kappa c^2}{\kappa(1+c^2)}\right)} \\
&= 1 + \frac{\kappa^2(1+c^2)^3 - 2\kappa(c^2+\kappa)(1+c^2)(1+c^2\kappa)}{(c^2+\kappa^2)(1+c^2\kappa)^2} \\
&= 1 + \frac{-\kappa^2 c^6 - 2\kappa^3 c^4 + \kappa^2 c^4 - 2\kappa c^4 - 2\kappa^3 c^2 + \kappa^2 c^2 - 2\kappa c^2 - \kappa^2}{(c^2+\kappa^2)(1+c^2\kappa)^2} \\
&= \frac{\kappa^4 c^4 - 2\kappa^3 c^4 + \kappa^2 c^4 + \kappa^2 c^2 - 2\kappa c^2 + c^2}{(c^2+\kappa^2)(1+c^2\kappa)^2} \\
&= \frac{c^2(1+c^2\kappa^2)(\kappa-1)^2}{(c^2+\kappa^2)(1+c^2\kappa)^2}.
\end{aligned}$$

Similarly, if n is an odd number, it follows that

$$\begin{aligned}
\lim_{n\to\infty} \frac{f(x_{n+1}) - f(x_*)}{f(x_n) - f(x_*)} &= 1 + \frac{1}{\left(\frac{\kappa^2 c^2+1}{1+c^2}\right)\left(\frac{c^2+\kappa}{\kappa(1+c^2)}\right)^2} - \frac{2\left(\frac{\kappa c^2+1}{1+c^2}\right)}{\left(\frac{\kappa^2 c^2+1}{1+c^2}\right)\left(\frac{c^2+\kappa}{\kappa(1+c^2)}\right)} \\
&= 1 + \frac{\kappa^2(1+c^2)^3 - 2\kappa(\kappa c^2+1)(1+c^2)(c^2+\kappa)}{(c^2\kappa^2+1)(c^2+\kappa)^2} \\
&= 1 + \frac{-\kappa^2 c^6 - 2\kappa^3 c^4 + \kappa^2 c^4 - 2\kappa c^4 - 2\kappa^3 c^2 + \kappa^2 c^2 - 2\kappa c^2 - \kappa^2}{(c^2\kappa^2+1)(c^2+\kappa)^2} \\
&= \frac{\kappa^2 c^4 - 2\kappa c^4 + c^4 + \kappa^4 c^2 - 2\kappa^3 c^2 + \kappa^2 c^2}{(c^2\kappa^2+1)(c^2+\kappa)^2} \\
&= \frac{c^2(c^2+\kappa^2)(\kappa-1)^2}{(1+c^2\kappa^2)(c^2+\kappa)^2}.
\end{aligned}$$

This completes our proof. □

The next theorem gives the asymptotic behavior of MG steplength.

Theorem 3.3. *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the MG method. Then*

$$\lim_{n\to\infty} \alpha_{2n}^{\mathrm{MG}} = \frac{1+c^2}{\lambda_1(1+c^2\kappa)},$$

and

$$\lim_{n\to\infty} \alpha_{2n+1}^{\mathrm{MG}} = \frac{1+c^2}{\lambda_1(c^2+\kappa)},$$

where c has the same value as in Theorem 3.1.

Proof. By theorem 3.1, it follows that

$$\lim_{n\to\infty} \alpha_{2n}^{\mathrm{MG}} = \lim_{n\to\infty} \frac{\sum_{j=1}^{N} \lambda_j \zeta_{j,2n}^2}{\sum_{j=1}^{N} \lambda_j^2 \zeta_{j,2n}^2} = \frac{1}{\lambda_1 \frac{1}{1+c^2} + \lambda_N \frac{c^2}{1+c^2}} = \frac{1+c^2}{\lambda_1(1+c^2\kappa)}.$$

Similarly,

$$\lim_{n\to\infty} \alpha_{2n+1}^{\mathrm{MG}} = \frac{1}{\lambda_1 \frac{c^2}{1+c^2} + \lambda_N \frac{1}{1+c^2}} = \frac{1+c^2}{\lambda_1(c^2+\kappa)}.$$

This completes our proof. □

The next theorem gives the asymptotic property of the gradient vector.

Theorem 3.4. *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the MG method. Then*

$$\lim_{n\to\infty} \frac{\|g_{n+1}\|^2}{\|g_n\|^2} = \frac{c^2(\kappa-1)^2}{(c^2+\kappa)(1+c^2\kappa)},$$

where c has the same value as in Theorem 3.1.

Proof. For any n, it follows from (2.4) that

$$\frac{\|g_{n+1}\|^2}{\|g_n\|^2} = \frac{\sum_{j=1}^{N} \left(1-\alpha_n^{\mathrm{MG}}\lambda_j\right)^2 \zeta_{j,n}^2}{\sum_{j=1}^{N} \zeta_{j,n}^2}.$$

Combining theorem 3.1 and theorem 3.3 implies

$$\begin{aligned}\lim_{n\to\infty}\frac{\|g_{2n+1}\|^2}{\|g_{2n}\|^2} &= \frac{\left(1-\frac{1+c^2}{1+c^2\kappa}\right)^2\lambda_1^{-1}\frac{1}{1+c^2}+\left(1-\frac{(1+c^2)\kappa}{1+c^2\kappa}\right)^2\lambda_N^{-1}\frac{c^2}{1+c^2}}{\lambda_1^{-1}\frac{1}{1+c^2}+\lambda_N^{-1}\frac{c^2}{1+c^2}}\\ &= \frac{\frac{(\kappa-1)^2c^4}{(1+c^2\kappa)^2}\kappa+\frac{(\kappa-1)^2}{(1+c^2\kappa)^2}c^2}{c^2+\kappa}\\ &= \frac{c^2(\kappa-1)^2}{(c^2+\kappa)(1+c^2\kappa)},\end{aligned}$$

and

$$\begin{aligned}\lim_{n\to\infty}\frac{\|g_{2n+2}\|^2}{\|g_{2n+1}\|^2} &= \frac{\left(1-\frac{1+c^2}{c^2+\kappa}\right)^2\lambda_1^{-1}\frac{c^2}{1+c^2}+\left(1-\frac{(1+c^2)\kappa}{c^2+\kappa}\right)^2\lambda_N^{-1}\frac{1}{1+c^2}}{\lambda_1^{-1}\frac{c^2}{1+c^2}+\lambda_N^{-1}\frac{1}{1+c^2}}\\ &= \frac{\frac{(\kappa-1)^2}{(c^2+\kappa)^2}c^2\kappa+\frac{(\kappa-1)^2c^4}{(c^2+\kappa)^2}}{1+c^2\kappa}\\ &= \frac{c^2(\kappa-1)^2}{(c^2+\kappa)(1+c^2\kappa)},\end{aligned}$$

which yields the desired conclusion. □

Then, we present the relationship between the quadratic function and the gradient vector.

Corollary 3.5. *Consider the linear system $Ax=b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the MG method. Then*

$$\lim_{n\to\infty}\frac{f(x_{2n+2})-f(x_*)}{f(x_{2n})-f(x_*)}=\lim_{n\to\infty}\frac{\|g_{n+1}\|^4}{\|g_n\|^4}.$$

Proof. The desired conclusion follows immediately by combining theorem 3.2 and theorem 3.4. □

Finally, we give another property of the gradient vector.

Theorem 3.6. *Consider the linear system $Ax=b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the MG method. Then*

$$\lim_{n\to\infty}\frac{g_{2n+1}^{\mathsf{T}}Ag_{2n+1}}{g_{2n}^{\mathsf{T}}Ag_{2n}}=\frac{c^2(\kappa-1)^2}{(1+c^2\kappa)^2},$$

and

$$\lim_{n\to\infty}\frac{g_{2n+2}^{\mathsf{T}}Ag_{2n+2}}{g_{2n+1}^{\mathsf{T}}Ag_{2n+1}}=\frac{c^2(\kappa-1)^2}{(c^2+\kappa)^2},$$

where c has the same value as in Theorem 3.1.

Proof. Combining theorem 3.1 and theorem 3.3 implies

$$\begin{aligned}\lim_{n\to\infty}\frac{g_{2n+1}^{\mathsf{T}}Ag_{2n+1}}{g_{2n}^{\mathsf{T}}Ag_{2n}}&=\left(1-\frac{1+c^2}{\lambda_1(1+c^2\kappa)}\lambda_1\right)^2\frac{1}{1+c^2}+\left(1-\frac{1+c^2}{\lambda_1(1+c^2\kappa)}\lambda_N\right)^2\frac{c^2}{1+c^2}\\&=\frac{c^4(\kappa-1)^2}{(1+c^2\kappa)^2}\frac{1}{1+c^2}+\frac{(\kappa-1)^2}{(1+c^2\kappa)^2}\frac{c^2}{1+c^2}\\&=\frac{c^2(\kappa-1)^2}{(1+c^2\kappa)^2},\end{aligned}$$

and

$$\begin{aligned}\lim_{n\to\infty}\frac{g_{2n+2}^{\mathsf{T}}Ag_{2n+2}}{g_{2n+1}^{\mathsf{T}}Ag_{2n+1}}&=\left(1-\frac{1+c^2}{\lambda_1(c^2+\kappa)}\lambda_1\right)^2\frac{c^2}{1+c^2}+\left(1-\frac{1+c^2}{\lambda_1(c^2+\kappa)}\lambda_N\right)^2\frac{1}{1+c^2}\\&=\frac{(\kappa-1)^2}{(c^2+\kappa)^2}\frac{c^2}{1+c^2}+\frac{c^4(\kappa-1)^2}{(c^2+\kappa)^2}\frac{1}{1+c^2}\\&=\frac{c^2(\kappa-1)^2}{(c^2+\kappa)^2},\end{aligned}$$

which completes our proof. □

We point out that some results also hold for the SD method. Other results are similar, though the left-hand sides require an extra operation with A, see Section 2.5 for the relevant properties of SD.

3.2 New gradient methods with alignment

As far as we know, all existing gradient methods with alignment are based on the Cauchy steplength. After a further rearrangement of steps, Gonzaga and Schneider [83] concludes that one could break the Cauchy cycle by periodically applying some short steplengths to accelerate the convergence of gradient methods. We show here that such condition is not necessary and several methods that potentially possess the same feature without Cauchy step can be derived.

De Asmundis et al. [56] observed that a constant equal to $1/(\lambda_1+\lambda_N)$ could lead to alignment property. Here we extend it to a more general case.

Theorem 3.7. *Consider the linear system* (2.1) *and the gradient method* (2.3) *with a positive constant steplength* $\hat{\alpha}$ *such that*

$$\hat{\alpha} \leq \frac{2}{\lambda_1+\lambda_N} \tag{3.1}$$

being used to solve (2.1)*. Then the sequence* $\{x_n\}$ *converges to* x_* *for any starting point* x_0*. Moreover, if equality holds, then*

$$\lim_{n\to\infty} \frac{\zeta_{i,n}}{\zeta_{1,n}} = \begin{cases} 0, & i = 2,3,\ldots,N-1, \\ \frac{\zeta_{N,0}}{\zeta_{1,0}}(-1)^n, & i = N; \end{cases} \tag{3.2}$$

otherwise,

$$\lim_{n\to\infty} \frac{\zeta_{i,n}}{\zeta_{1,n}} = 0, \quad i = 2,3,\ldots,N. \tag{3.3}$$

Proof. We have

$$\hat{\alpha} \leq \frac{2}{\lambda_1+\lambda_N} < \frac{2}{\lambda_N} \leq 2\alpha_n^{\mathrm{SD}}.$$

By [128], it is easy to deduce that the sequence $\{x_n\}$ converges to x_* with a steplength $\alpha < 2\alpha_n^{\mathrm{SD}}$. Hence, the first statement holds. One finds that

$$\lim_{n\to\infty} \frac{\zeta_{i,n}}{\zeta_{1,n}} = \frac{\zeta_{i,0}}{\zeta_{1,0}} \lim_{n\to\infty} \left(\frac{1-\hat{\alpha}\lambda_i}{1-\hat{\alpha}\lambda_1}\right)^n.$$

Let

$$\varphi_i = \frac{1-\hat{\alpha}\lambda_i}{1-\hat{\alpha}\lambda_1}.$$

For (3.3) to be satisfied, we need to impose the condition $|\varphi_i| < 1$ for all $i = 2,3,\ldots,N$, which yields

$$(\lambda_i+\lambda_1)\hat{\alpha} < 2, \quad (\lambda_i-\lambda_1)\hat{\alpha} > 0.$$

The second one is obviously satisfied, while the first one leads to

$$\hat{\alpha} < \frac{2}{\lambda_1+\lambda_N}.$$

If equality holds, then

$$\varphi_i = \frac{\lambda_N+\lambda_1-2\lambda_i}{\lambda_N-\lambda_1},$$

It is clear that $\varphi_N = -1$. Then the second statement trivially follows, which completes the proof. □

Note that $i = 1$ leads to the trivial case $\varphi_1 = 1$, and thus the limit in both (3.2) and (3.3) equals 1. From Theorem 3.7 we find that condition (3.1) has twofold effect: driving the alignment property when strict partial order holds, as shown in (3.3), and forcing the search into a two-dimensional space in the equal case, as shown in (3.2). It means that if there exist some steps asymptotically making the equality of (3.2) attainable, then it has similar tendency with the SD method, namely, alternating between two orthogonal directions. On the other hand, we can add a fractional factor to periodically break the cycle. This asymptotically yields a constant steplength strictly smaller than $2/(\lambda_1 + \lambda_N)$, leading to alignment process in the subsequent several iterations according to (3.3).

Recall that Dai and Yang [49] proposed a gradient method of the form

$$\alpha_n^{\mathrm{AO}} = \frac{\|g_n\|}{\|Ag_n\|}.$$

It asymptotically converges to the optimal steplength

$$\lim_{n\to\infty} \alpha_n^{\mathrm{AO}} = \alpha^{\mathrm{OPT}} = \frac{2}{\lambda_1 + \lambda_N},$$

which minimizes the coefficient matrix

$$\alpha^{\mathrm{OPT}} = \arg\min_{\alpha} \|I - \alpha A\|.$$

Thus we call it asymptotically optimal (AO) method. Notice that the following relationship holds

$$\alpha_n^{\mathrm{MG}} \leq \alpha_n^{\mathrm{AO}} \leq \alpha_n^{\mathrm{SD}}, \tag{3.4}$$

which can be easily proved by the Cauchy-Schwarz inequality

$$\frac{g_n^{\mathsf{T}} A g_n}{g_n^{\mathsf{T}} A^2 g_n} \leq \frac{\|g_n\| \, \|Ag_n\|}{\|Ag_n\|^2} = \frac{\|g_n\|^2}{\|Ag_n\| \, \|g_n\|} \leq \frac{g_n^{\mathsf{T}} g_n}{g_n^{\mathsf{T}} A g_n}.$$

It is known that AO generates monotone curve and often leads to slow convergence.

We observe that the AO step satisfies condition (3.3) and may potentially be improved by a cyclic breaking. For example, we can choose a shorter one to constantly align the gradient vector to the one-dimensional space spanned by v_1. Let $\tilde{\alpha}_n = \theta \alpha_n^{\mathrm{AO}}$ where $0 < \theta < 1$. It

follows that

$$\lim_{n\to\infty} \tilde{\alpha}_n < \frac{2}{\lambda_1+\lambda_N}.$$

From Theorem 3.7, we observe that $\tilde{\alpha}_n$ can asymptotically trigger the alignment behavior. Hence, we can write a new gradient method called AO with alignment (AOA) as follows

$$\alpha_n^{\mathrm{AOA}} = \begin{cases} \alpha_n^{\mathrm{AO}}, & n \bmod (d_1+d_2) < d_1, \\ \tilde{\alpha}_n, & n \bmod (d_1+d_2) = d_1, \\ \alpha_{n-1}^{\mathrm{AOA}}, & \text{otherwise}, \end{cases} \tag{3.5}$$

with $d_1, d_2 \geq 1$. Important differences between SDA and AOA come from the fact that the Cauchy step in SDA zigzags itself in two orthogonal directions, while the AO step in AOA converges to a constant and the constant leads later to the same feature.

On the other hand, since the spectral properties of MG have been studied in Section 3.1, we are now prepared to propose our new methods based on them. We first give some notations

$$\alpha_n^{\mathrm{A2}} = \left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}}} + \frac{1}{\alpha_n^{\mathrm{MG}}} \right)^{-1},$$

$$\alpha_n^{\mathrm{Y2}} = 2 \left(\sqrt{ \left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}}} - \frac{1}{\alpha_n^{\mathrm{MG}}} \right)^2 + \frac{4 g_n^{\mathsf{T}} A g_n}{\left(\alpha_{n-1}^{\mathrm{MG}}\right)^2 g_{n-1}^{\mathsf{T}} A g_{n-1}} } + \frac{1}{\alpha_{n-1}^{\mathrm{MG}}} + \frac{1}{\alpha_n^{\mathrm{MG}}} \right)^{-1}.$$

Note that Y2 has been proposed in [51] as a component of the 2-dimensional finite termination method.

Theorem 3.8. *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the MG method. Then*

$$\lim_{n\to\infty} \alpha_n^{\mathrm{A2}} = \frac{1}{\lambda_1+\lambda_N}.$$

Proof. This conclusion follows immediately by theorem 3.3. □

Theorem 3.9. *Consider the linear system $Ax = b$ with Assumption 2.2 holds. Assume that the sequence of solution vectors $\{x_n\}$ is generated by the MG method. Then*

$$\lim_{n\to\infty} \alpha_n^{\mathrm{Y2}} = \frac{1}{\lambda_N}.$$

Moreover,

$$\lim_{n\to\infty}\left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}}\alpha_n^{\mathrm{MG}}}-\frac{g_n^{\mathsf{T}}Ag_n}{\left(\alpha_{n-1}^{\mathrm{MG}}\right)^2 g_{n-1}^{\mathsf{T}}Ag_{n-1}}\right)=\lambda_1\lambda_N.$$

Proof.

$$\alpha_n^{\mathrm{Y2}}=2\left(\sqrt{\left(\alpha_n^{\mathrm{A2}}\right)^{-2}-\frac{4}{\alpha_{n-1}^{\mathrm{MG}}\alpha_n^{\mathrm{MG}}}+\frac{4g_n^{\mathsf{T}}Ag_n}{\left(\alpha_{n-1}^{\mathrm{MG}}\right)^2 g_{n-1}^{\mathsf{T}}Ag_{n-1}}}+\left(\alpha_n^{\mathrm{A2}}\right)^{-1}\right)^{-1}.$$

By combining theorem 3.3 and theorem 3.6, we notice that

$$\lim_{n\to\infty}\frac{g_{2n+2}^{\mathsf{T}}Ag_{2n+2}}{\left(\alpha_{2n+1}^{\mathrm{MG}}\right)^2 g_{2n+1}^{\mathsf{T}}Ag_{2n+1}}=\lim_{n\to\infty}\frac{g_{2n+1}^{\mathsf{T}}Ag_{2n+1}}{\left(\alpha_{2n}^{\mathrm{MG}}\right)^2 g_{2n}^{\mathsf{T}}Ag_{2n}}=\frac{\lambda_1^2c^2(\kappa-1)^2}{(1+c^2)^2}.$$

Hence, one can see that

$$\begin{aligned}\lim_{n\to\infty}\left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}}\alpha_n^{\mathrm{MG}}}-\frac{g_n^{\mathsf{T}}Ag_n}{\left(\alpha_{n-1}^{\mathrm{MG}}\right)^2 g_{n-1}^{\mathsf{T}}Ag_{n-1}}\right)&=\frac{\lambda_1^2(1+c^2\kappa)(c^2+\kappa)}{(1+c^2)^2}-\frac{\lambda_1^2c^2(\kappa-1)^2}{(1+c^2)^2}\\&=\frac{\lambda_1^2(c^2+c^4\kappa+\kappa+c^2\kappa^2-c^2\kappa^2+2c^2\kappa-c^2)}{(1+c^2)^2}\\&=\lambda_1\lambda_N.\end{aligned}$$

Further, along with theorem 3.8, we have

$$\lim_{n\to\infty}\alpha_n^{\mathrm{Y2}}=2\left(\sqrt{(\lambda_1+\lambda_N)^2-4\lambda_1\lambda_N}+\lambda_1+\lambda_N\right)^{-1}=\frac{1}{\lambda_N}.$$

This completes our proof. □

One may conclude from Theorems 3.8 and 3.9 that A2 and Y2 are similar to the auxiliary steplengths discussed in [56] and [57]. However, since MG has shorter steplength than SD, we expect that the former might be more smoother than the latter. After a substitution of labels, we are able to define MG with alignment (MGA) and MG with constant steplength (MGC) as follows

$$\alpha_n^{\mathrm{MGA}}=\begin{cases}\alpha_n^{\mathrm{MG}}, & n \bmod (d_1+d_2)<d_1,\\ \alpha_n^{\mathrm{A2}}, & n \bmod (d_1+d_2)=d_1,\\ \alpha_{n-1}^{\mathrm{MGA}}, & \text{otherwise},\end{cases}\tag{3.6}$$

$$\alpha_n^{\mathrm{MGC}} = \begin{cases} \alpha_n^{\mathrm{MG}}, & n \bmod (d_1+d_2) < d_1, \\ \alpha_n^{\mathrm{Y2}}, & n \bmod (d_1+d_2) = d_1, \\ \alpha_{n-1}^{\mathrm{MGC}}, & \text{otherwise.} \end{cases} \tag{3.7}$$

with $d_1, d_2 \geq 1$. Recall that the motivation in [56] is to align the algorithm search into the one-dimensional space spanned by v_1, which can be summarized by Theorem 3.7. On the other hand, the strategy in [57] is to foster a special steplength towards the inverse of the largest eigenvalue for which the gradient element has not vanished. One could easily conclude from Theorem 3.7 that $\hat{\alpha} = 1/\lambda_N$ satisfies also the former motivation, while $\hat{\alpha} = 1/(\lambda_1 + \lambda_N)$ may not satisfy the latter one which depends on the relative magnitude of λ_1. This may explain the superiority of SDC compared to SDA, and we will see later that this argument remains true for MGA and MGC.

The spectral properties that have been discussed above can be generalized to other basic steplengths of the form

$$\alpha_n = \frac{g_n^{\mathsf{T}} A^{\rho} g_n}{g_n^{\mathsf{T}} A^{\rho+1} g_n},$$

with $\rho \geq 0$. Nonetheless, formulations other than SD and MG are not viewed as promising since extra sparse matrix-vector multiplication is required, which often give similar convergence results but at tremendous computational cost.

3.3 Convergence analysis

For the convergence analysis of the aforementioned methods, recall that a convergence framework has been established in [45] which requires a tool called Property A. Inspired by the pioneering work of BB, Friedlander et al. [74] proposed a general framework called gradient method with retards (GMR), but AO can not be directly formalized by such framework. Given m a positive integer, let $\bar{n} = \max\{0, n-m\}$). Recall that a generalization of GMR [45] can be defined as follows

$$\alpha_n^{\mathrm{DGMR}} = \left(\frac{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)+\upsilon} g_{\tau(n)}} \right)^{\frac{1}{\upsilon}}.$$

with

$$\tau(n) \in \{\bar{n}, \bar{n}+1, \ldots, n-1, n\}, \quad \rho(n) \in \{q_1, \ldots, q_m\}, \quad q_j \geq 0, \quad \upsilon > 0.$$

Here, we call it Dai's generalization of GMR (DGMR). After a further selection of parameters $\rho(n)$ and $\tau(n)$, we observe that SD, MG, BB are both special cases of this framework, as well as many other alternate and cyclic gradient methods (see, e.g, [45, 50, 52]). The convergence of DGMR is summarized in Theorem 3.10. Dai [45] stated this result without proof. Here, a complete proof is provided and shall also be exploited later by other theorems.

Theorem 3.10. *Consider the gradient method* (2.3) *with steplength* (2.7) *being used to solve the linear system* (2.1). *Then the sequence* $\{x_n\}$ *converges to* x_* *for any starting point* x_0.

Proof. Since gradient methods are invariant under orthogonal transformations, we assume without loss of generality that $A = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ and $\lambda_1 = 1$. Let

$$R(A,u) = \frac{u^\intercal A u}{u^\intercal u}$$

be the Rayleigh quotient for non-zero vector u. Let

$$u_1 = A^{(\rho(n)+\upsilon-1)/2} g_{\tau(n)},$$

it follows that

$$\alpha_n^{\mathrm{DGMR}} = \left(\frac{1}{R(A,u_1)} \cdot \frac{g_{\tau(n)}^\intercal A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^\intercal A^{\rho(n)+\upsilon-1} g_{\tau(n)}} \right)^{\frac{1}{\upsilon}} \leq \left(\frac{1}{\lambda_1} \cdot \frac{g_{\tau(n)}^\intercal A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^\intercal A^{\rho(n)+\upsilon-1} g_{\tau(n)}} \right)^{\frac{1}{\upsilon}}.$$

Applying this result recursively, one has

$$\alpha_n^{\mathrm{DGMR}} \leq \left(\frac{1}{\lambda_1^{\upsilon-1}} \cdot \frac{g_{\tau(n)}^\intercal A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^\intercal A^{\rho(n)+1} g_{\tau(n)}} \right)^{\frac{1}{\upsilon}} \leq \frac{1}{\lambda_1}.$$

It follows from the similar deduction that

$$\alpha_n^{\mathrm{DGMR}} \geq \frac{1}{\lambda_N}.$$

Thus we can choose $c_1 = \lambda_N$, and then the first relationship of Property A trivially follows. For the second one, we choose c_2 of the form

$$c_2 = \frac{\left(\frac{2}{3}\right)^\upsilon}{1 - \left(\frac{2}{3}\right)^\upsilon} \lambda_\mu^{\bar{q}}, \tag{3.8}$$

where $\bar{q} = \max_{i\in[1,m_0]} q_i$. Let $m_0 = m$. For all $\mu \in \{1,\ldots,N-1\}$ and $j \in \{0,\ldots,\min\{n,m_0\}\}$, one obtains that

$$\left(\alpha_n^{\mathrm{DGMR}}\right)^{-1} = \left(\frac{\sum_{i=1}^{N} g_{i,\tau(n)}^2 \lambda_i^{\rho(n)+\upsilon}}{\sum_{i=1}^{\mu} g_{i,\tau(n)}^2 \lambda_i^{\rho(n)} + \sum_{i=\mu+1}^{N} g_{i,\tau(n)}^2 \lambda_i^{\rho(n)}}\right)^{\frac{1}{\upsilon}} \geq \left(\frac{\lambda_{\mu+1}^{\upsilon} \sum_{i=\mu+1}^{N} g_{i,n-j}^2 \lambda_i^{\rho(n)}}{\lambda_{\mu}^{\bar{q}} G(n-j,\mu) + \sum_{i=\mu+1}^{N} g_{i,n-j}^2 \lambda_i^{\rho(n)}}\right)^{\frac{1}{\upsilon}}.$$

For all $\varepsilon > 0$, suppose that

$$G(n-j,\mu) \leq \varepsilon, \quad g_{\mu+1,n-j}^2 \geq c_2\varepsilon.$$

Then,

$$\left(\alpha_n^{\mathrm{DGMR}}\right)^{-1} \geq \left(\frac{\lambda_{\mu+1}^{\upsilon} g_{\mu+1,n-j}^2}{\lambda_{\mu}^{\bar{q}}\varepsilon + g_{\mu+1,n-j}^2}\right)^{\frac{1}{\upsilon}} \geq \left(\frac{c_2}{\lambda_{\mu}^{\bar{q}} + c_2}\right)^{\frac{1}{\upsilon}} \lambda_{\mu+1}.$$

Substituting (3.8) into the above deduction it follows that

$$\left(\alpha_n^{\mathrm{DGMR}}\right)^{-1} \geq \frac{2}{3}\lambda_{\mu+1},$$

which ensures the second condition of Property A. Thus, the desired conclusion follows by imposing Lemma 2.8. □

Notice that the case of $\rho(n) = 0$, $\tau(n) = k$ and $\upsilon = 2$ recovers the AO steplength. As a consequence of Theorem 3.10, the convergence result of AOA can be established.

Theorem 3.11. *Consider the linear system* (2.1) *being solved by AOA. Then the sequence* $\{x_n\}$ *converges to* x_* *for any starting point* x_0.

Proof. The first part of AOA equals exactly the AO steplength which satisfies DGMR framework, thus having the Property A. The second part can be written as follows

$$\tilde{\alpha}_n = \theta\left(\frac{g_{\tau(n)}^{\mathsf{T}} g_{\tau(n)}}{g_{\tau(n)}^{\mathsf{T}} A^2 g_{\tau(n)}}\right)^{\frac{1}{2}}.$$

As seen in the proof of Theorem 3.10, we obtain that

$$\lambda_1 < \alpha_n^{-1} \leq \frac{\lambda_N}{\theta}.$$

Therefore, we can choose $c_1 = \lambda_N/\theta$. For the second condition, we can keep formula (3.8) for c_2, which gives the same result as the deduction for DGMR, and thus AOA has Property A. Then the desired conclusion follows from Lemma 2.8. □

For the convergence of MGA and MGC, we can give similar statements. Notice that the analysis of SDA and SDC can be applied here without difficulty since SD and MG share similar properties as discussed in Section 3.1.

Theorem 3.12. *Consider the linear system* (2.1) *being solved by MGA. Then the sequence* $\{x_n\}$ *converges to* x_* *for any starting point* x_0.

Proof. This proof follows as before with $m_0 = d_2$, $c_1 = 2\lambda_N$ and $c_2 = 2$. For all $j \in \{0, \ldots, \min\{n, m_0\}\}$, let $\alpha_n = \alpha^{\mathrm{A2}}_{n-j+1}$. By the fact that

$$\frac{1}{2\lambda_N} \leq \frac{\min\{\alpha^{\mathrm{MG}}_{n-j}, \alpha^{\mathrm{MG}}_{n-j+1}\}}{2} \leq \alpha_n \leq \min\{\alpha^{\mathrm{MG}}_{n-j}, \alpha^{\mathrm{MG}}_{n-j+1}\} \leq \frac{1}{\lambda_1}, \tag{3.9}$$

one can verify that the first property is true. In addition, since (3.9) implies that

$$\alpha_n^{-1} \geq \frac{1}{\min\{\alpha^{\mathrm{MG}}_{n-j}, \alpha^{\mathrm{MG}}_{n-j+1}\}} \geq \frac{1}{\alpha^{\mathrm{MG}}_{n-j}} = \frac{g_{n-j}^{\mathsf{T}} A^2 g_{n-j}}{g_{n-j}^{\mathsf{T}} A g_{n-j}},$$

by applying the proof of Theorem 3.10, it follows that

$$\alpha_n^{-1} \geq \frac{c_2}{1+c_2}\lambda_{\mu+1}.$$

Substituting c_2 yields the second property. Thus, the desired conclusion follows from Lemma 2.8. □

Theorem 3.13. *Consider the linear system* (2.1) *being solved by MGC. Then the sequence* $\{x_n\}$ *converges to* x_* *for any starting point* x_0.

Proof. Let $m_0 = d_2$. Similar to the proof of Theorem 3.12, for all $j \in \{0, \ldots, \min\{n, m_0\}\}$, we can write $\alpha_n = \alpha^{\mathrm{Y2}}_{n-j+1}$. It is clear that

$$\alpha_n \leq \min\{\alpha^{\mathrm{MG}}_{n-j}, \alpha^{\mathrm{MG}}_{n-j+1}\} \leq \frac{1}{\lambda_1}. \tag{3.10}$$

By using the Kantorovich inequality (see, e.g., Lemma 5.8 in [131]), it follows that

$$\frac{g_{n-j+1}^{\mathsf{T}} A g_{n-j+1}}{g_{n-j}^{\mathsf{T}} A g_{n-j}} = \frac{g_{n-j}^{\mathsf{T}} A g_{n-j} \cdot g_{n-j}^{\mathsf{T}} A^3 g_{n-j}}{\left(g_{n-j}^{\mathsf{T}} A^2 g_{n-j}\right)^2} - 1 \leq \frac{(\lambda_N - \lambda_1)^2}{4\lambda_N \lambda_1},$$

from which we can obtain that

$$\alpha_n \geq 2\left(\sqrt{(\lambda_N - \lambda_1)^2 + \kappa(\lambda_N - \lambda_1)^2} + 2\lambda_N\right)^{-1}. \tag{3.11}$$

Since the second member is a constant, combining (3.10) and (3.11) yields the first property. Finally, comparing (3.10) with (3.9) implies that the second result can be obtained in the same manner as that follows from the proof of Theorem 3.13. Thus, we arrive at the desired conclusion. □

3.4 Numerical experiments

In this section, we provide numerical experiments for different gradient methods by two types of problems. The first one is generated randomly by MATLAB and the second one is a two-point boundary value problem. In both examples, the right-hand side b of the linear system (2.1) is computed by $b = Ax_*$ where x_* is a random vector such that $x_* \in (-10, 10)$. Furthermore, the tests are started from zero vectors and the stopping criterion is fixed with $\|g_n\| < 10^{-6}\|g_0\|$. All experiments are performed using MATLAB R2018b on a machine with Double Intel Core i7 2.8 GHz CPU.

In the first example, we consider the random problem generated by the MATLAB built-in function `sprandsym`, which has appeared in De Asmundis et al. [56]. We would like to know the impact of parameters on the convergence behavior of alternate gradient methods. The plots in Figures 3.1 and 3.2 show some examples where AOA, SDC and MGC are used for solving random problems. Figs. 3.1 illustrates the impact of parameter θ on AOA iterations. We can see that $\theta \in [0.5, 0.7]$ leads to the most efficient algorithm. In Figures 3.2, we notice that the blue areas illustrate the situation where the choice of parameters leads to fast convergence, while the red ones show the opposite results. It is convenient to propose an adaptive way to select parameters according to the matrix dimension and the distribution of eigenvalues, but the spectral property is generally unknown to us and obtaining the distribution of eigenvalues is as difficult as solving a linear system.

In the following experiments, we choose $\theta = 0.5$ for AOA and $d_1 = 4$ and $d_2 = 4$ for all methods since according to Figures 3.1 and 3.2 they often produce good results. Figure 3.3

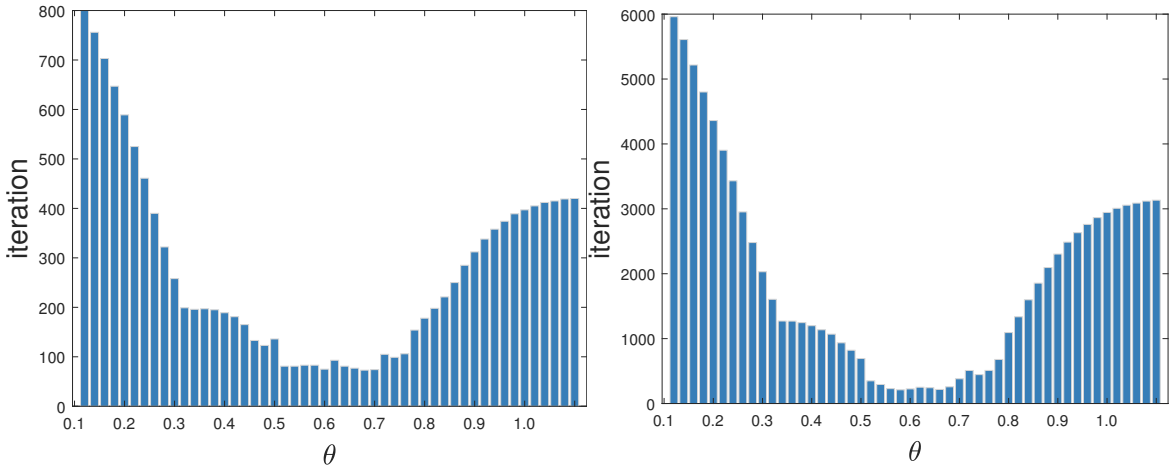

Figure 3.1 Comparison of different θ in AOA where $d_1 = 4$ and $d_2 = 4$. We generate random problems with $N = 100$: $\kappa = 10^2$ (left), $\kappa = 10^3$ (right).

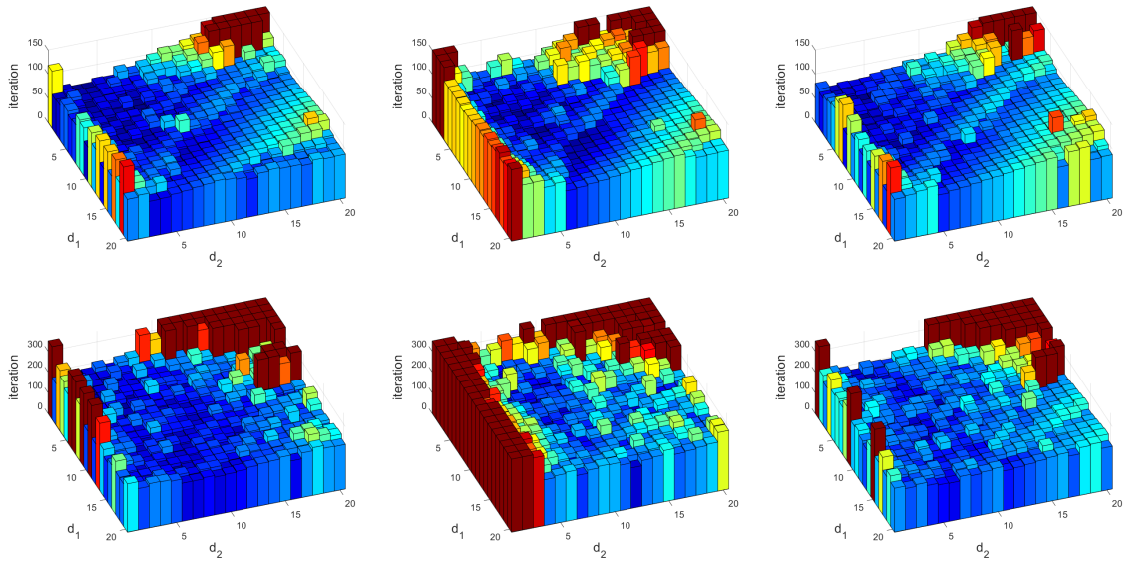

Figure 3.2 Comparison of SDC (left), AOA (center) and MGC (right) through random problems with $N = 100$: $\kappa = 10^2$ (top), $\kappa = 10^3$ (bottom).

shows the convergence behaviors of several typical gradient methods. Our tests reveal that the basic methods such as SD, MG and AO are far less efficient than others. The traditional gradient steps are unrealistic to be used in practice, especially for ill-conditioned problems. In addition, the convergence results of SDA and SDC are not slower than BB and DY in most cases. Notice that DY has nonmonotone curve in the residual figure, though it would show monotone behavior when drawing the values of function f.

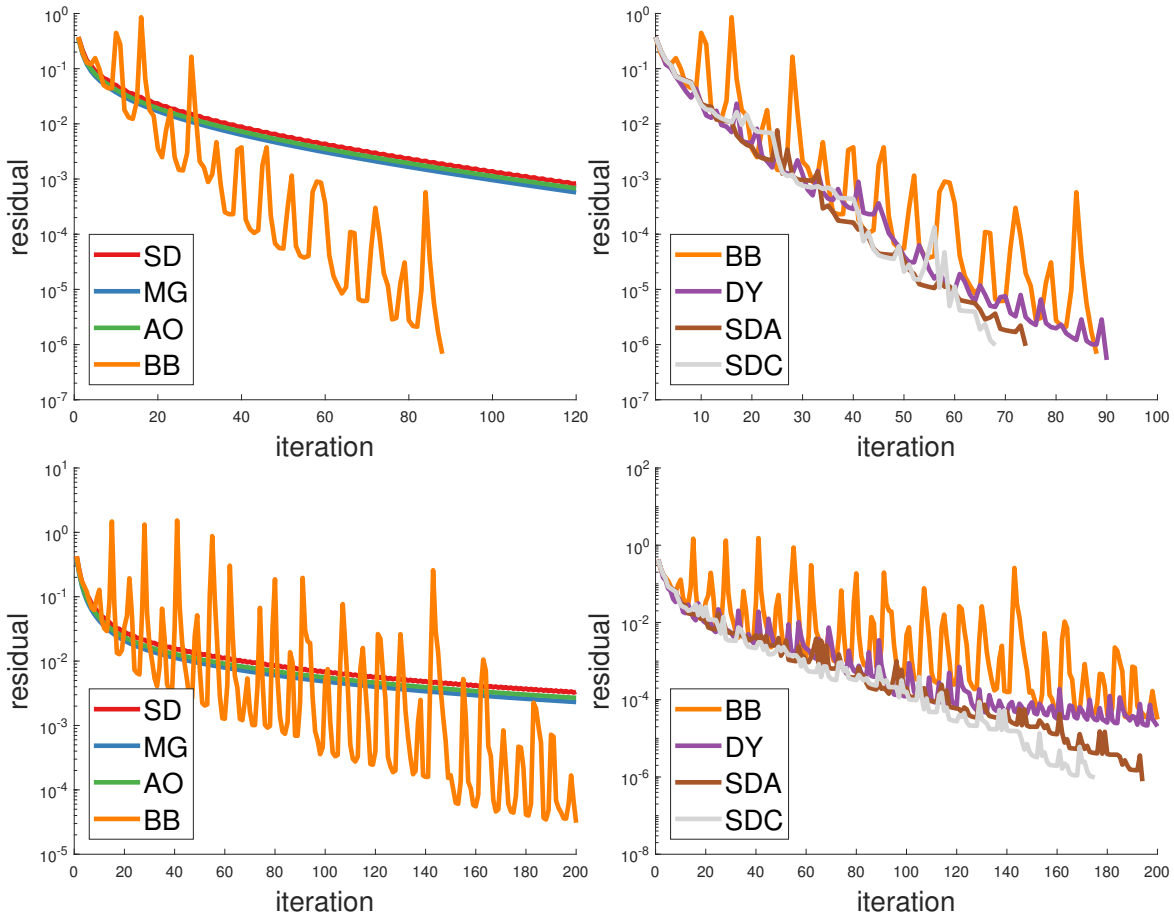

Figure 3.3 Comparison of different gradient methods through the random problems: $N = 100$, $\kappa = 100$ (top), $N = 100$, $\kappa = 1000$ (bottom).

In Table 3.1, we provide the number of iterations required by SDA and SDC as well as the new methods with $\kappa = 10^2, 10^3, 10^4, 10^5$ and $N = 200, 400, 600, 800, 1000$. In all cases, we list only the final average results in the table for which 10 repeated experiments were conducted to circumvent the extreme conditions. One finds that SDC and MGC give better results than other three methods. On the other hand, SDA deteriorates when κ becomes larger, and the comparison between AOA and MGA could not lead to a commun conclusion. This observation is contrary to our expectations, as we speculated that AOA would always have bad performance, due to its twofold asymptotically zigzag behavior, as mentioned in Section 3.2. Further tests have shown that AOA is more sensitive to the choice of parameters than MGA and MGC. The problem size seems to be a less critical issue in view of the test results.

To show the correctness of our analysis, particularly, the comparisons between the aligned methods and the basic gradient methods are illustrated in Figure 3.4. The problem size is

Table 3.1 The follows results are obtained for the problems generated randomly by the MATLAB built-in function `sprandsym`. In the table we illustrate the average number of iterations among 10 tests with $d_1 = 4$ and $d_2 = 4$ for all methods.

Conditioning	Size	SDA	SDC	AOA	MGA	MGC
$\kappa = 10^2$	$N = 200$	68	67	80	73	70
	$N = 400$	70	69	80	73	66
	$N = 600$	73	72	83	73	73
	$N = 800$	71	74	81	73	74
	$N = 1000$	70	76	80	74	75
$\kappa = 10^3$	$N = 200$	199	177	197	209	187
	$N = 400$	201	187	222	216	190
	$N = 600$	199	195	226	205	181
	$N = 800$	191	185	232	207	181
	$N = 1000$	194	182	227	209	190
$\kappa = 10^4$	$N = 200$	614	479	571	536	507
	$N = 400$	648	506	525	525	501
	$N = 600$	602	497	560	540	490
	$N = 800$	626	484	534	536	509
	$N = 1000$	619	475	547	515	488
$\kappa = 10^5$	$N = 200$	1300	1118	1246	1225	1153
	$N = 400$	1318	1176	1393	1299	1126
	$N = 600$	1374	1228	1255	1253	1231
	$N = 800$	1390	1190	1452	1269	1169
	$N = 1000$	1381	1273	1490	1321	1251

chosen as $N = 1000$. Each comparison consists of four pairs of plot: $\kappa = 10^2, 10^3, 10^4, 10^5$, respectively. The figures show that in all cases, the aligned methods terminate in relatively few iterations. Further insight into the plots can be gained by observing the oscillating behavior, which reveals that SDA usually has large magnitude of oscillation, while MGA is the smoothest one. It is known that the oscillation of a convergence curve is closely related to the numerical stability (see, e.g., [100]). In view of the convergence performance and the stability behavior for the three aligned methods, the use of the MGA step is more recommended than the SDA step.

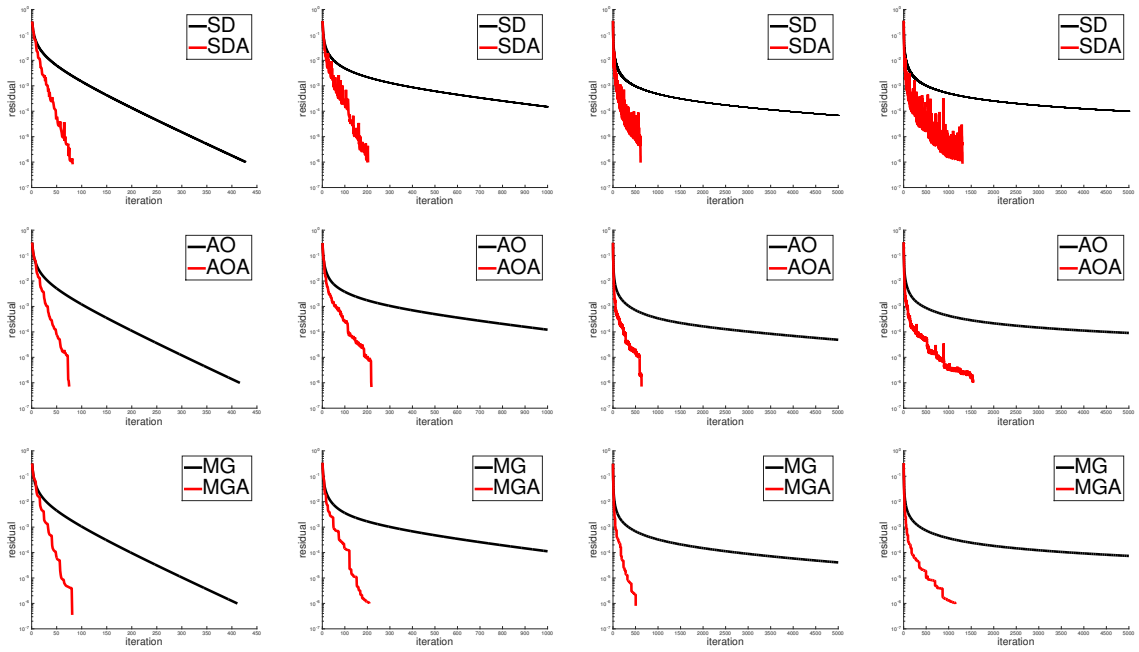

Figure 3.4 Top: comparison of SDA and SD. Middle: comparison of AOA and AO. Bottom: comparison of MGA and MG. Random problems are generated with $N = 1000$: $\kappa = 10^2$ (first), $\kappa = 10^3$ (second), $\kappa = 10^4$ (third), $\kappa = 10^5$ (fourth).

The second experiment is a two-point boundary value problem (see, e.g., [74, 50]). The tridiagonal matrix A after discretization by the finite difference method is of the form

$$A = \text{tridiag}(-\frac{1}{h^2}, \frac{2}{h^2}, -\frac{1}{h^2}),$$

where $h = 11/N$. Notice that with the augmentation of matrix dimension N, the condition number κ will also increase. The purpose of this is to confirm the previous results obtained for the new methods. Since SDA and MGA are as expected less efficient than SDC and MGC, we shall not address them again and focus on other three methods. The AOA curve is retained for the sake of comparison. Here, we provide results of the cases $N = 10^2, 10^3, 10^4, 10^5$ and illustrate the residual curves. Figure 3.5 shows that MGC are quite competitive with SDC, while AOA can not beat them in all cases.

Similar to the previous results, we can see that SDC oscillates mightily in all cases, while AOA is slightly better than SDC emerging from the fact that it yields smoother transitional curves between the spikes. MGC shows the most promising performance since it gives not only a competitive convergence speed, but also a much smoother curve than other methods. In one direction, the MG-based method minimizes indeed the residual value. On the other

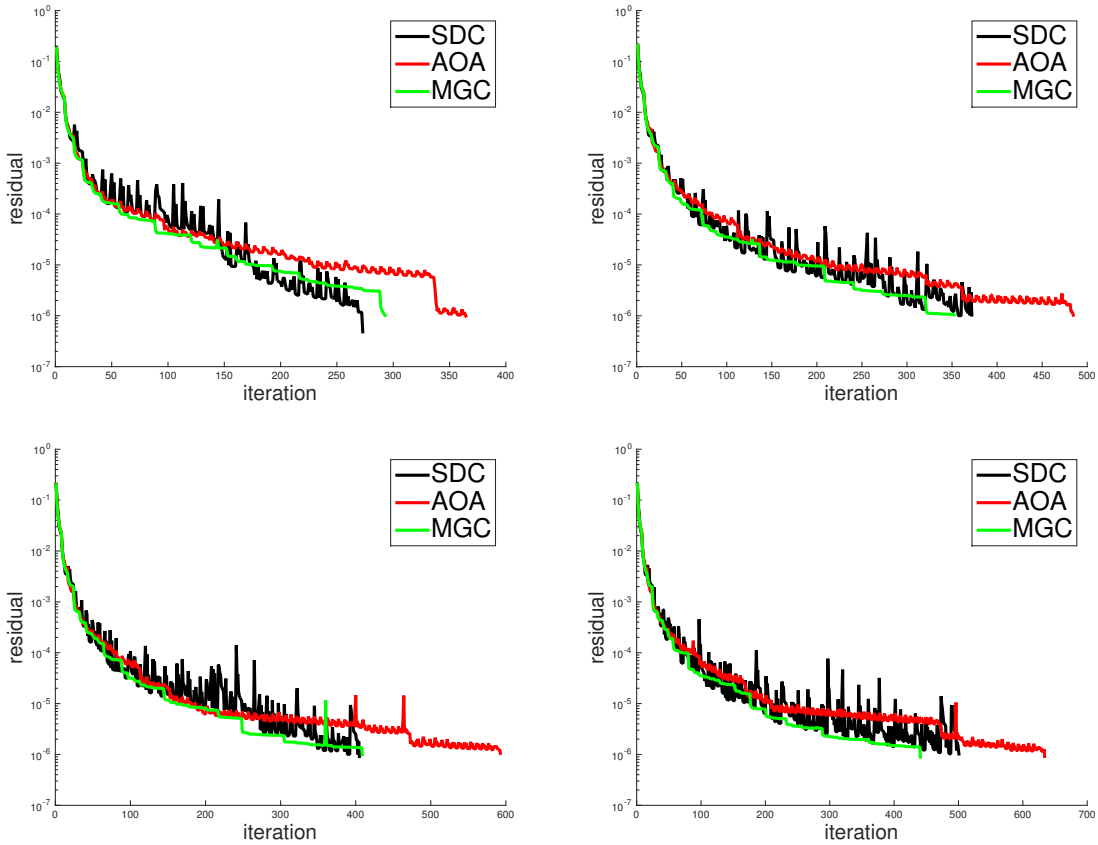

Figure 3.5 Comparison of different gradient methods through the two-point boundary value problems: $N = 10^2$ (top-left), $N = 10^3$ (top-right), $N = 10^4$ (bottom-left), $N = 10^5$ (bottom-right).

hand, it is known that stability generally favors short steplengths. Along with (3.4), the desired conclusion follows.

Finally, we compare our new methods with the conjugate gradient (CG) method [89]. Two examples are used to show the robustness and efficiency of the proposed methods. The first example concerns the random problems with perturbation generated by MATLAB, which have the following form

$$\tilde{A}x = b, \quad \tilde{A} = A + \delta V,$$

where δ is a small positive value and V is a nonsymmetric matrix. Still, let κ be the condition number of A. We choose $\delta = 10^{-4}$. V is generated by the MATLAB function `sprand`. We compare also our methods with the generalized minimum residual (GMRES) method [132] in view of the perturbation. Here we use the restarted GMRES where algorithm is restarted

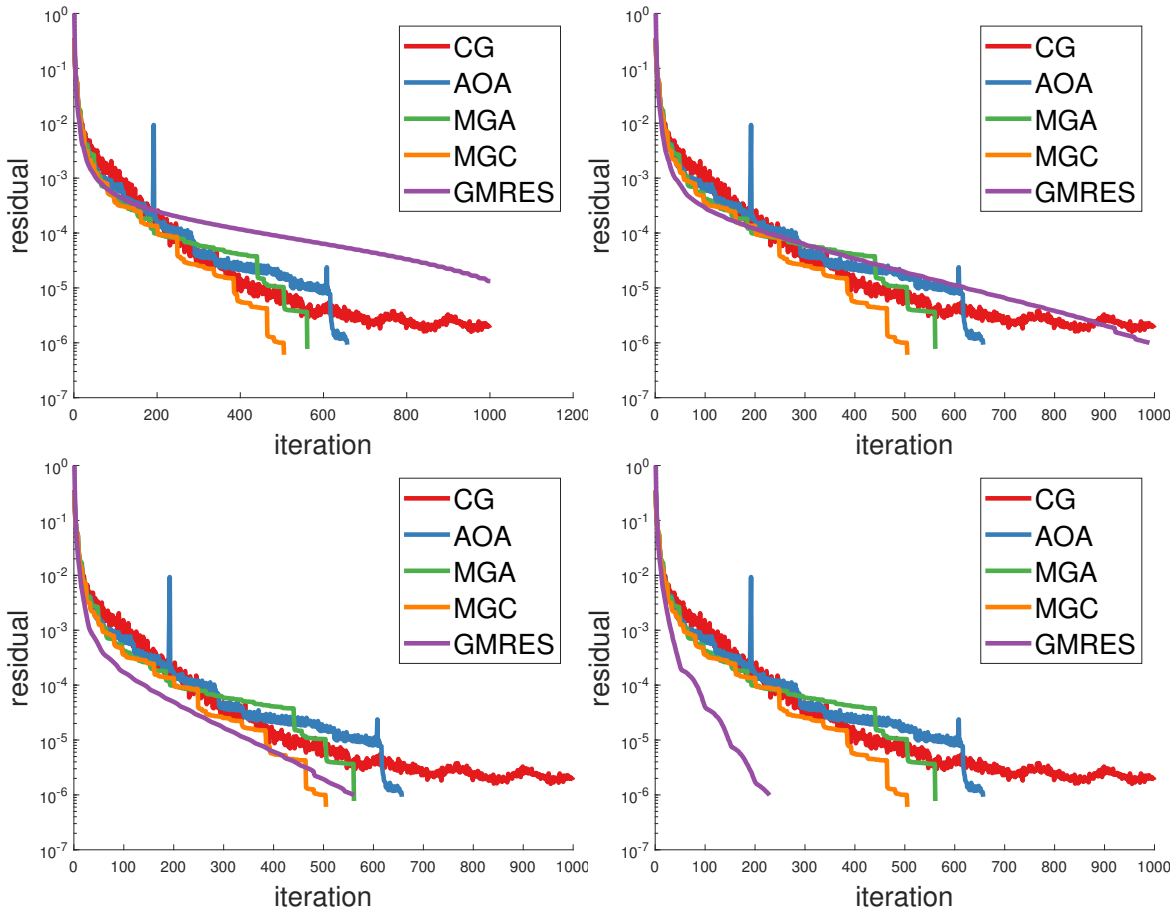

Figure 3.6 Comparison of the new methods with CG and restarted GMRES through random problems with perturbation where $N = 10^2$ and $\kappa = 10^4$. GMRES is restarted every l iterations: $l = 10$ (top-left), $l = 20$ (top-right), $l = 30$ (bottom-left), $l = 50$ (bottom-right).

every l iterations. The computational results are shown in Figure 3.6. We observe that CG curve decreases in the beginning but stagnates in the end, while our new methods are robust and resistant to perturbation. Restarted GMRES is more efficient than our methods when $l > 30$. However, it needs to store l more vectors, which means lN storage locations, and requires about l more vector updates and dot products than gradient methods. The second example is drawn from the University of Florida Sparse Matrix Collection [53] which is a large-scale system with $N = 1564794$ and $\kappa = 1.225 \times 10^8$. The matrix name is `Flan_1565` with ID 2544. This is obtained from a 3D mechanical problem discretized by hexahedral finite elements. The computational result is shown in Figure 3.7. The new methods perform better than CG in this case and the best performance is realized by MGC. Figure 3.8 shows a comparison of the performance of MGC with other efficient gradient methods through the large-scale problem. From these plots, we can see that MGC solves the large problem efficiently and its curves decrease smoothly.

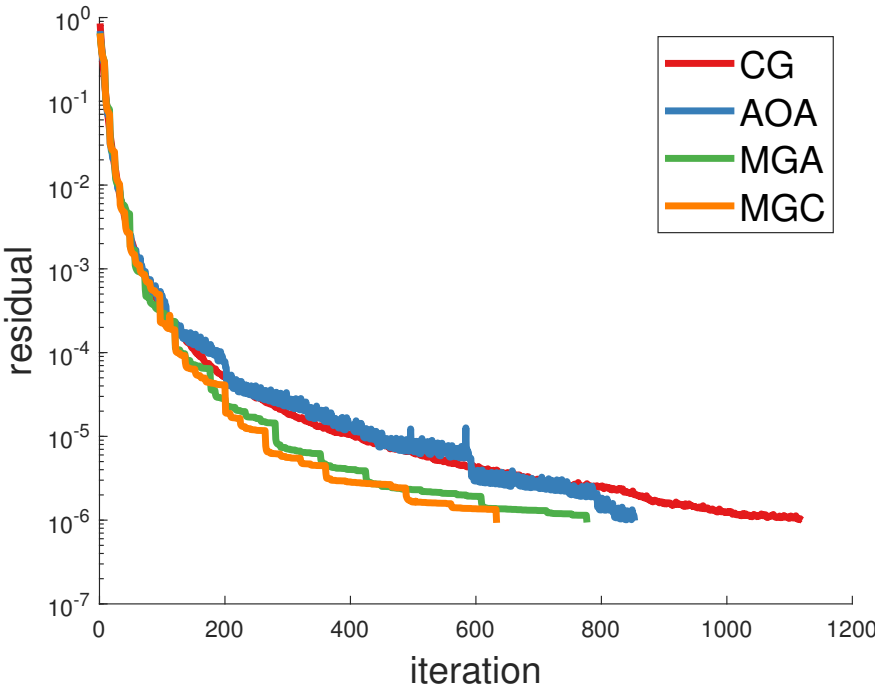

Figure 3.7 Comparison of the new methods with conjugate gradient through a large-scale problem: $N = 1564794$.

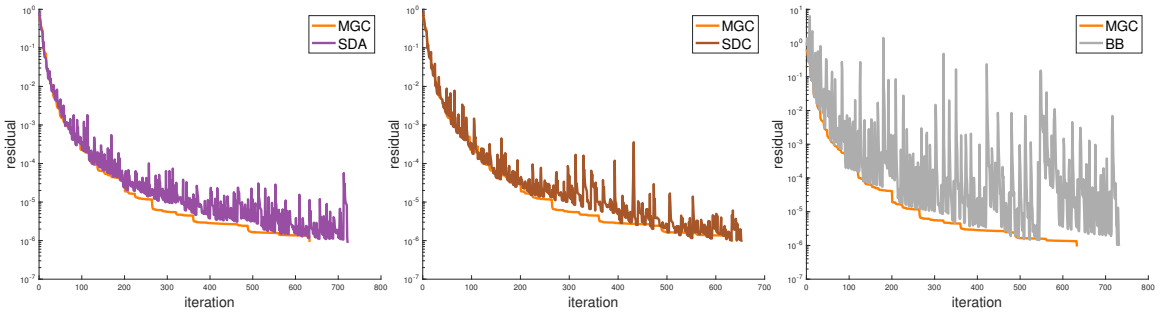

Figure 3.8 Comparison of the new method MGC with other gradient methods through a large-scale problem: $N = 1564794$.

3.5 Conclusion

We address first the spectral properties of the MG method. Our analysis effectively extends that in [117] which includes only the SD method. In fact, it is possible to further extend the current results based on the P-gradient framework as mentioned in Section 3.1. We introduce here only the MG-based properties since it is the most promising candidate for a further formulation. Additionally, our analysis shows that the Cauchy step is not an indispensable component to trigger the alignment behavior. Hence, the Cauchy-short framework proposed in [83] could be updated and generalized to our cases.

In this chapter, we propose three new gradient methods with alignment, called AOA, MGA and MGC, respectively. MGC shows great competitiveness to SDC, while SDA, AOA and MGA have been proved to be less efficient than other methods in most cases. A closer examination of AOA and MGC reveals that they are more stable than SDC. Such feature may contribute to the problem of loss of precision [100]. The new methods with alignment present several advantages over the Krylov subspace methods.

There exist two main heuristics to accelerate the gradient methods. One is to reveal the spectral property, which yields eventually the alignment methods; the other depends on the "decreasing together" behavior as presented in [51]. For example, BB and DY both possess such feature. According to our experiments, the former seems to be more effective than the latter. Further investigation of different heuristics seems to be a good research topic in the future.

Chapter 4

Parameter estimation in splitting methods

This chapter presents enhancement strategies for the Hermitian and skew-Hermitian splitting method based on gradient iterations. The spectral properties are exploited for the parameter estimation, often resulting in a better convergence compared with the arbitrary choice. In particular, gradient iterations with early stopping can generate a rough estimate of optimal parameter, which is better than the traditional choice, since the latter often causes stability problems or slow convergence. Additionally, lagged gradient methods are considered as inner solvers for the splitting method. Experiments show that they are competitive with conjugate gradient in low precision.

4.1 Hermitian and skew-Hermitian splitting

In this chapter we are interested in solving the linear system

$$Ax = b, \tag{4.1}$$

where A is a non-Hermitian positive definite matrix of size N. For non-Hermitian but symmetric systems, some algorithms such as COCG [147], modified QMR [71], CSYM [27], COCR [139], and MHSS [13] could be used. It is well known that any square matrix is similar to a complex symmetric matrix (see Theorem 4.4.24 in [92]). As a result, the eigenvalue distribution of a complex symmetric matrix does not possess any special properties.

For the solution of general non-Hermitian linear systems, one approach is to apply conjugate gradient method to the Hermitian normal equation $A^{\mathsf{H}}Ax = A^{\mathsf{H}}b$ where generally an appropriate preconditioner is needed since its condition number is the square of the

condition number of A. This method is sometimes called CGNR which minimizes the $A^{\mathsf{H}}A$-norm of the error, i.e., the 2-norm of the residual $b - Ax_n$ over the relative n-dimensional affine space. Another similar method called CGNE is possible for the system $AA^{\mathsf{H}}y = b$ with $x = A^{\mathsf{H}}y$ which minimizes the AA^{H}-norm of the error in y_n, i.e., the 2-norm of the error $x_* - x_n$ where x_* is the exact solution. We refer the reader to reference [131] for more details. These approaches are often badly affected by ill-conditioning that makes them unappealing in practice.

Another approach is to transform Equation (4.1) into real equivalent form and solve the resulting real system by a Krylov subspace method such as BiCG [64], GMRES [132], QMR [73, 72] and BiCGSTAB [145]. For good overviews, see [16, 133, 103, 138], see also Section 1.1 and the references therein. We remark that the coefficients of Equation (4.1) have the following form

$$A = \mathrm{Re}(A) + i\mathrm{Im}(A), \quad x = \mathrm{Re}(x) + i\mathrm{Im}(x), \quad b = \mathrm{Re}(b) + i\mathrm{Im}(b),$$

where $\mathrm{Re}(\cdot)$ extracts the real part and $\mathrm{Im}(\cdot)$ extracts the imaginary part, and i is the imaginary unit. It is possible to rewrite the system into real and complex parts that gives rise to the 2-by-2 block formulations

$$A_* \begin{pmatrix} \mathrm{Re}(x) \\ \mathrm{Im}(x) \end{pmatrix} = \begin{pmatrix} \mathrm{Re}(A) & -\mathrm{Im}(A) \\ \mathrm{Im}(A) & \mathrm{Re}(A) \end{pmatrix} \begin{pmatrix} \mathrm{Re}(x) \\ \mathrm{Im}(x) \end{pmatrix} = \begin{pmatrix} \mathrm{Re}(b) \\ \mathrm{Im}(b) \end{pmatrix},$$

$$A_{**} \begin{pmatrix} \mathrm{Re}(x) \\ -\mathrm{Im}(x) \end{pmatrix} = \begin{pmatrix} \mathrm{Re}(A) & \mathrm{Im}(A) \\ \mathrm{Im}(A) & -\mathrm{Re}(A) \end{pmatrix} \begin{pmatrix} \mathrm{Re}(x) \\ -\mathrm{Im}(x) \end{pmatrix} = \begin{pmatrix} \mathrm{Re}(b) \\ \mathrm{Im}(b) \end{pmatrix}.$$

While there exist several other real equivalent formulations, these two are basic and share some common properties with other special cases. In view of the spectral properties, if λ is an eigenvalue of non-Hermitian matrix A, then the complex conjugate pairs λ and $\bar{\lambda}$ are the eigenvalues of the real equivalent form A_*. Thus the spectrum $\sigma(A_*)$ is symmetric with respect to the real axis. For the second formulation, if $\lambda \in \sigma(A)$, then λ, $-\lambda$, $\bar{\lambda}$ and $-\bar{\lambda}$ are also the eigenvalues of A_{**}. Therefore, the spectrum is symmetric with respect to the origin. It is sometimes the case that the coefficient matrix has only one-sided spectrum that makes the second formulation problematic in practice. For example, if a matrix has only positive eigenvalues, then the eigenvalues in the first formulation lie in a half-plane which excludes the origin while the second one reflects the spectrum through the origin that degrades the convergence rate. Theoretically, the convergence rate for the general case tends to be the square root of the rate for the one-sided spectrum when using the same iterative method. We

mention [6, 54, 21] and the references therein for more theoretical arguments and numerical results.

It has been observed that splitting methods can be used with success. The traditional alternating direction implicit method [120] has inspired the construction of alternate two-step splittings $A = \mathscr{M}_1 - \mathscr{N}_1$ and $A = \mathscr{M}_2 - \mathscr{N}_2$, and this leads to an iteration called Hermitian and skew-Hermitian splitting (HSS) [11] in which alternately a shifted Hermitian system and a shifted skew-Hermitian system are solved. HSS has received so much attention (see, e.g., [12, 10, 20, 13–15, 90, 134, 150, 121]), possibly due to its guaranteed convergence and mathematical beauty.

Let H and S denote the Hermitian and skew-Hermitian parts of A, respectively. Let A^{H} be the conjugate transpose of matrix A. It follows that

$$H = \frac{A + A^{\mathsf{H}}}{2}, \quad S = \frac{A - A^{\mathsf{H}}}{2}.$$

Let I be the identity matrix. In short, the HSS method is defined as

$$\begin{cases} (\gamma I + H)x_{n+\frac{1}{2}} = (\gamma I - S)x_n + b, \\ (\gamma I + S)x_{n+1} = (\gamma I - H)x_{n+\frac{1}{2}} + b, \end{cases} \tag{4.2}$$

with $\gamma > 0$. It could be regarded as a stationary iterative process

$$x_{n+1} = Tx_n + p,$$

where x_0 is a given vector. Following [11] let us set

$$\mathscr{M}_1 = \gamma I + H, \quad \mathscr{N}_1 = \gamma I - S, \quad \mathscr{M}_2 = \gamma I + S, \quad \mathscr{N}_2 = \gamma I - H.$$

The coefficients T and p can be expressed as

$$T = \mathscr{M}_2^{-1}\mathscr{N}_2\mathscr{M}_1^{-1}\mathscr{N}_1, \quad p = \mathscr{M}_2^{-1}(I + \mathscr{N}_2\mathscr{M}_1^{-1})b.$$

Let $\sigma(\cdot)$ be the spectrum of a matrix and let $\rho(\cdot)$ be the spectral radius. Convergence result for (4.2) in the non-Hermitian positive definite case was established in [11]

$$\rho(T) \leq \left\|\mathscr{N}_2\mathscr{M}_1^{-1}\right\| = \max_{\lambda \in \sigma(H)} \frac{|\lambda - \gamma|}{|\lambda + \gamma|}, \tag{4.3}$$

where $\|\cdot\|$ denotes 2-norm. This shows that the spectral radius of iteration matrix T is less than 1. As a result, HSS has guaranteed convergence for which the speed depends only on the Hermitian part H. Let $\lambda_i(\cdot)$ be the ith eigenvalue of a matrix in ascending order. The key observation here is that choosing

$$\gamma = \gamma_* = \sqrt{\lambda_1(H)\lambda_N(H)} \tag{4.4}$$

leads to the well-known upper bound

$$\rho(T) \leq \frac{\sqrt{\kappa(H)}-1}{\sqrt{\kappa(H)}+1}, \tag{4.5}$$

where $\kappa(\cdot)$ denotes the condition number. It is noteworthy that inequality (4.4) is similar to the convergence result for conjugate gradient (CG) [89, 144] in terms of A-norm error. As mentioned in [11], γ_* minimizes the upper bound of $\rho(T)$ but not $\rho(T)$ itself. In some cases the right-hand side of (4.3) may not be an accurate approximation to the spectral radius. Since very little theory is available on direct minimization, we still try to approximate indirectly the optimal parameter γ_*.

4.2 Asymptotic analysis of steepest descent

In this section we consider the Hermitian positive definite (HPD) linear system

$$Hx = \hat{b} \tag{4.6}$$

of size N. The solution x_* is the unique global minimizer of convex quadratic function

$$f(x) = \frac{1}{2}x^{\mathsf{H}}Hx - \hat{b}^{\mathsf{H}}x. \tag{4.7}$$

For $n = 0, 1, \ldots$, the gradient method is of the form

$$x_{n+1} = x_n - \alpha_n g_n, \tag{4.8}$$

where $g_n = \nabla f(x_n) = Hx_n - \hat{b}$. This gives the updating formula

$$g_{n+1} = g_n - \alpha_n H g_n. \tag{4.9}$$

The steepest descent (SD) method proposed in [33] defines a sequence of steplengths as follows

$$\alpha_n^{\mathrm{SD}} = \frac{g_n^{\mathsf{H}} g_n}{g_n^{\mathsf{H}} H g_n}, \tag{4.10}$$

which is the reciprocal of Rayleigh quotient. It minimizes the quadratic function f or the A-norm error of the system (4.6) and gives theoretically an optimal result at each step

$$\alpha_n^{\mathrm{SD}} = \arg\min_{\alpha} f(x_n - \alpha g_n) = \arg\min_{\alpha} \|(I - \alpha H)e_n\|_H^2,$$

where $e_n = x_* - x_n$. This classical method is known to behave badly in practice. The directions tend to asymptotically alternate between two orthogonal directions resulting in a slow convergence [1].

The motivation for this chapter arose during the development of efficient gradient methods. We notice that generally SD converges much slower than CG for HPD systems. However, the spectral properties of the former could be beneficial to parameter estimation. Akaike [1] provided a probability distribution model for the asymptotic analysis of SD. It appears that standard techniques used in linear algebra are not very helpful in this case. The so-called two-step invariance property led to the work of Nocedal et al. [117] in which further asymptotic results are presented. Let $v_i(\cdot)$ be the eigenvector corresponding to the eigenvalue $\lambda_i(\cdot)$. Relevant properties in [117] which will be exploited in the following text can be briefly described in Lemma 4.1. Note that a symmetric positive definite real matrix was used in [117]. Therefore, we give some remarks in the proof for Hermitian case.

Lemma 4.1. *Assume that* $\lambda_1(H) < \cdots < \lambda_N(H)$. *Assume that* $v_1^{\mathsf{H}}(H)g_0 \neq 0$ *and* $v_N^{\mathsf{H}}(H)g_0 \neq 0$. *Consider the gradient method* (4.8) *with steplength* (4.10) *being used to solve* (4.6). *Then*

$$\lim_{n\to\infty} \alpha_{2n}^{\mathrm{SD}} = \frac{1+c^2}{\lambda_1(H)(1+c^2\kappa(H))}, \tag{4.11}$$

$$\lim_{n\to\infty} \alpha_{2n+1}^{\mathrm{SD}} = \frac{1+c^2}{\lambda_1(H)(c^2+\kappa(H))}, \tag{4.12}$$

and

$$\lim_{n\to\infty} \frac{\|g_{2n+1}\|^2}{\|g_{2n}\|^2} = \frac{c^2(\kappa(H)-1)^2}{(1+c^2\kappa(H))^2}, \tag{4.13}$$

$$\lim_{n\to\infty} \frac{\|g_{2n+2}\|^2}{\|g_{2n+1}\|^2} = \frac{c^2(\kappa(H)-1)^2}{(c^2+\kappa(H))^2}, \tag{4.14}$$

for some constant c.

Proof. Let $\mathrm{Re}(\cdot)$ and $\mathrm{Im}(\cdot)$ be the real and imaginary parts, respectively. The coefficients of system (4.6) have the following form

$$H = \mathrm{Re}(H) + \imath \mathrm{Im}(H), \quad x = \mathrm{Re}(x) + \imath \mathrm{Im}(x), \quad \hat{b} = \mathrm{Re}(\hat{b}) + \imath \mathrm{Im}(\hat{b}),$$

where $\imath$ denotes the imaginary unit. It is possible to rewrite system (4.6) into the real equivalent form

$$\tilde{H}\tilde{x} = \begin{pmatrix} \mathrm{Re}(H) & -\mathrm{Im}(H) \\ \mathrm{Im}(H) & \mathrm{Re}(H) \end{pmatrix} \begin{pmatrix} \mathrm{Re}(x) \\ \mathrm{Im}(x) \end{pmatrix} = \begin{pmatrix} \mathrm{Re}(\hat{b}) \\ \mathrm{Im}(\hat{b}) \end{pmatrix} = \tilde{b}. \tag{4.15}$$

By Lemma 3.3 and Theorem 5.1 in [117], it is known that results (4.11) to (4.14) hold in the real case. To prove the desired result, it suffices to show that SD applied to (4.15) is equivalent to that for (4.6), namely, they should yield the same sequences of gradient vectors and steplengths. One finds that

$$\begin{aligned} g_n &= (\mathrm{Re}(H) + \imath \mathrm{Im}(H))(\mathrm{Re}(x_n) + \imath \mathrm{Im}(x_n)) - (\mathrm{Re}(\hat{b}) + \imath \mathrm{Im}(\hat{b})) \\ &= \varphi_n + \imath\, \psi_n, \end{aligned} \tag{4.16}$$

where

$$\begin{aligned} \varphi_n &= \mathrm{Re}(H)\mathrm{Re}(x_n) - \mathrm{Im}(H)\mathrm{Im}(x_n) - \mathrm{Re}(\hat{b}), \\ \psi_n &= \mathrm{Re}(H)\mathrm{Im}(x_n) + \mathrm{Im}(H)\mathrm{Re}(x_n) - \mathrm{Im}(\hat{b}). \end{aligned}$$

Assume that the 2 blocks in $\tilde{x}_n$ is the same as the real and imaginary parts of x_n, respectively. Then, from (4.15) one obtains that

$$\tilde{g}_n = \tilde{H}\tilde{x}_n - \tilde{b} = \begin{pmatrix} \varphi_n \\ \psi_n \end{pmatrix}. \tag{4.17}$$

On the other hand, let

$$\tilde{\alpha}_n = \frac{\tilde{g}_n^{\mathsf{T}} \tilde{g}_n}{\tilde{g}_n^{\mathsf{T}} \tilde{H} \tilde{g}_n}.$$

Combining (4.16) and (4.17) implies $g_n^{\mathsf{H}} g_n = \tilde{g}_n^{\mathsf{T}} \tilde{g}_n$. Since $\mathrm{Im}(H)^{\mathsf{T}} = -\mathrm{Im}(H)$, it follows that $u^{\mathsf{T}}\mathrm{Im}(H)u = 0$ for all $u \in \mathbb{R}^N$, from which one obtains that

$$\mathrm{Re}(g_n)^{\mathsf{T}}\mathrm{Im}(H)\mathrm{Re}(g_n) = 0, \quad \mathrm{Im}(g_n)^{\mathsf{T}}\mathrm{Im}(H)\mathrm{Im}(g_n) = 0.$$

Hence, the following result holds:

$$\begin{aligned} g_n^{\mathsf{H}} H g_n &= (\mathrm{Re}(g_n) + \imath \mathrm{Im}(g_n))^{\mathsf{H}} (\mathrm{Re}(H) + \imath \mathrm{Im}(H))(\mathrm{Re}(g_n) + \imath \mathrm{Im}(g_n)) \\ &= \mathrm{Re}(g_n)^{\mathsf{T}} \mathrm{Re}(H) \mathrm{Re}(g_n) + \mathrm{Im}(g_n)^{\mathsf{T}} \mathrm{Re}(H) \mathrm{Im}(g_n) \\ &\quad + 2\mathrm{Im}(g_n)^{\mathsf{T}} \mathrm{Im}(H) \mathrm{Re}(g_n) \end{aligned}$$

Along with (4.15), this implies that $g_n^{\mathsf{H}} H g_n = \tilde{g}_n^{\mathsf{T}} \tilde{H} \tilde{g}_n$, according to which one finds that $\tilde{\alpha}_n = \alpha_n$ when the 2 blocks in $\tilde{g}_n$ are equal to the real and imaginary parts of g_n, respectively. Hence, the SD iteration for Hermitian system (4.6) and that for 2-by-2 real form yield exactly the same sequence of solutions. Since properties (4.11) to (4.14) in the real case has been proved in [117], we arrive at the desired conclusion. □

Concerning the assumption used in Lemma 4.1, if there exist repeated eigenvalues, then we can choose the eigenvectors so that the corresponding gradient components vanish [65]. If $v_1^{\mathsf{H}}(H) g_0 = 0$ or $v_N^{\mathsf{H}}(H) g_0 = 0$, then the second condition can be replaced by inner eigenvectors with no effect on the theoretical results.

It took some time before the spectral properties described in [117] were applied for solving linear systems. De Asmundis et al. [56] proposed an auxiliary steplength

$$\alpha_n^{\mathrm{A}} = \left(\frac{1}{\alpha_{n-1}^{\mathrm{SD}}} + \frac{1}{\alpha_n^{\mathrm{SD}}} \right)^{-1}, \tag{4.18}$$

which could be used for efficient implementations of gradient methods. The major result is a direct consequence of (4.11) and (4.12). We state the lemma without proof, see [56] for further discussion.

Lemma 4.2. *Under the assumptions of Lemma 4.1, the following result holds*

$$\lim_{n \to \infty} \alpha_n^{\mathrm{A}} = \frac{1}{\lambda_1(H) + \lambda_N(H)}. \tag{4.19}$$

Another direction of approach was based on a delicate derivation in [154]. Let us write $\alpha_n^{\mathrm{RA}} = \left(\alpha_n^{\mathrm{A}}\right)^{-1}$ and

$$\Gamma_n = \frac{1}{\alpha_{n-1}^{\mathrm{SD}} \alpha_n^{\mathrm{SD}}} - \frac{\|g_n\|^2}{\left(\alpha_{n-1}^{\mathrm{SD}}\right)^2 \|g_{n-1}\|^2}. \tag{4.20}$$

Yuan [154] developed a new auxiliary steplength of the form

$$\alpha_n^{\mathrm{Y}} = \frac{2}{\alpha_n^{\mathrm{RA}} + \sqrt{(\alpha_n^{\mathrm{RA}})^2 - 4\Gamma_n}}. \tag{4.21}$$

which leads to some 2-dimensional finite termination methods for solving system (4.6), see [154]. Let us now introduce an alternative steplength

$$\alpha_n^{\mathrm{Z}} = \frac{2}{\alpha_n^{\mathrm{RA}} - \sqrt{(\alpha_n^{\mathrm{RA}})^2 - 4\Gamma_n}}. \tag{4.22}$$

Let us write $\alpha_n^{\mathrm{RY}} = \left(\alpha_n^{\mathrm{Y}}\right)^{-1}$ and $\alpha_n^{\mathrm{RZ}} = \left(\alpha_n^{\mathrm{Z}}\right)^{-1}$. It follows that

$$\alpha_n^{\mathrm{RY}} + \alpha_n^{\mathrm{RZ}} = \alpha_n^{\mathrm{RA}}, \quad \alpha_n^{\mathrm{RY}}\alpha_n^{\mathrm{RZ}} = \Gamma_n.$$

The spectral properties of (4.20), (4.21) and (4.22) are shown in Lemma 4.3. Note that the consequences (4.23) and (4.24) have appeared in [57] for the real case. We prove them in full for completeness.

Lemma 4.3. *Under the assumptions of Lemma 4.1, the following limits hold*

$$\lim_{n\to\infty} \Gamma_n = \lambda_1(H)\lambda_N(H). \tag{4.23}$$

$$\lim_{n\to\infty} \alpha_n^{\mathrm{Y}} = \frac{1}{\lambda_N(H)}. \tag{4.24}$$

$$\lim_{n\to\infty} \alpha_n^{\mathrm{Z}} = \frac{1}{\lambda_1(H)}. \tag{4.25}$$

Proof. Combining (4.11) and (4.12) implies

$$\lim_{n\to\infty} \alpha_{n-1}^{\mathrm{SD}}\alpha_n^{\mathrm{SD}} = \frac{(1+c^2)^2}{\lambda_1^2(H)(c^2+\kappa(H))(1+c^2\kappa(H))}. \tag{4.26}$$

Combining (4.11) to (4.14), one could deduce that

$$\lim_{n\to\infty} \frac{1}{(\alpha_{2n}^{\mathrm{SD}})^2} \lim_{n\to\infty} \frac{\|g_{2n+1}\|^2}{\|g_{2n}\|^2} = \lim_{n\to\infty} \frac{1}{(\alpha_{2n+1}^{\mathrm{SD}})^2} \lim_{n\to\infty} \frac{\|g_{2n+2}\|^2}{\|g_{2n+1}\|^2},$$

from which one finds

$$\lim_{n\to\infty} \frac{\|g_n\|^2}{\left(\alpha_{n-1}^{\mathrm{SD}}\right)^2 \|g_{n-1}\|^2} = \frac{\lambda_1^2(H)c^2(\kappa(H)-1)^2}{(1+c^2)^2}. \tag{4.27}$$

The first equation follows by combining (4.26) and (4.27). Along with (4.19), this implies that

$$\lim_{n\to\infty} \left(\left(\alpha_n^{\mathrm{RA}}\right)^2 - 4\Gamma_n \right) = \left(\lambda_1(H) - \lambda_N(H)\right)^2,$$

which yields the desired limits (4.24) and (4.25). □

It is noteworthy that steplengths (4.21) and (4.22) could be expressed as the roots of a quadratic function

$$Q_n(\alpha) = \Gamma_n \alpha^2 - \alpha_n^{\mathrm{RA}} \alpha + 1, \tag{4.28}$$

with

$$Q_n(0) = 1, \quad Q_n(\alpha_n^{\mathrm{A}}) = \Gamma_n \left(\alpha_n^{\mathrm{A}}\right)^2,$$

$$Q_n(\alpha_{n-1}^{\mathrm{SD}}) = -\frac{\|g_n\|^2}{\|g_{n-1}\|^2}, \quad Q_n(\alpha_n^{\mathrm{SD}}) = -\frac{\left(\alpha_n^{\mathrm{SD}}\right)^2 \|g_n\|^2}{\left(\alpha_{n-1}^{\mathrm{SD}}\right)^2 \|g_{n-1}\|^2},$$

from which one could observe that $\Gamma_n > 0$ and

$$\alpha_n^{\mathrm{A}} < \alpha_n^{\mathrm{Y}} < \min\{\alpha_{n-1}^{\mathrm{SD}}, \alpha_n^{\mathrm{SD}}\}.$$

As mentioned in [154], a slightly shortened steplength would improve the efficiency of SD. This is one reason why the Yuan steplength could be fruitfully used in alternate gradient methods [51, 57].

4.3 Application to HSS iterations

4.3.1 Preliminary considerations

In this section we first try to compute estimates for parameter γ in the HSS method. One possible solution is to simply choose $\gamma = 1$ without resorting to special techniques, but experience shows that it often leads to very slow convergence, depending on the system being solved. Another approach is based on the observation that α was introduced to enable the bounded convergence, as seen in (4.3), and it is possible to express it differently. As an

example consider a positive definite diagonal matrix D such that

$$\begin{cases} (D+H)x_{n+\frac{1}{2}} = (D-S)x_n + b, \\ (D+S)x_{n+1} = (D-H)x_{n+\frac{1}{2}} + b. \end{cases} \tag{4.29}$$

As a result, the iteration matrix is of the form

$$T_D = (D+S)^{-1}(D-H)(D+H)^{-1}(D-S).$$

Notice that (4.29) is a special case of preconditioned HSS [22] when choosing $\alpha = 1$ and $P = D$. In particular, the fact that Theorem 2.1 in [22] holds for (4.29) implies $\rho(T_D) < 1$, yielding the guaranteed convergence.

By the similarity invariance of the matrix spectrum, we obtain

$$\rho(\hat{T}_D) = \rho(T_D),$$

where

$$\hat{T}_D = (D-H)(D+H)^{-1}(D-S)(D+S)^{-1}.$$

Notice that $DS = SD$ and so $(D-S)(D+S)^{-1}$ is a unitary matrix. On the basis of similar reasoning as in HSS [11], the spectral radius is bounded by

$$\rho(\hat{T}_D) \leq \left\| (D-H)(D+H)^{-1} \right\|.$$

A natural idea is to seek D so that the upper bound is small. At first glance we may choose D as the diagonal elements of H. Inspired by the diagonal weighted matrix in [71], the Euclidean norms of column vectors could also be exploited. However, the common experience is that these strategies may lead to a stagnation of convergence, and sometimes perform much worse than choosing $\gamma = 1$. We will not pursue them further in this chapter.

4.3.2 Parameter estimation based on gradient iterations

It is observed that (4.23) leads to a straightforward estimation of parameter γ_* in (4.4). Our goal now is to show that this approach is feasible. Assume that $x_0 = 0$ and

$$H = \text{diag}(1, 2, 10, 20, 100, 200, 1000, 2000). \tag{4.30}$$

Assume that $\hat{b}$ is constructed by $\hat{b} = Hx_*$ where x_* is a vector of all ones. We plot in Figure 4.1 the curves of (4.28) for a few representative iteration numbers. This figure shows that the

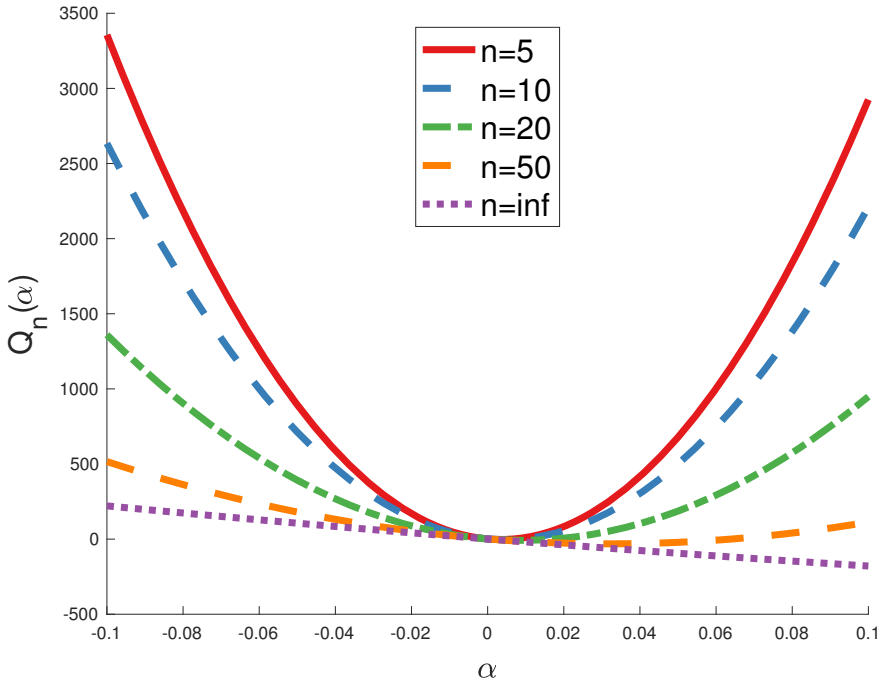

Figure 4.1 Curves of $Q_n(\alpha)$ for a few representative iteration numbers. SD is used for solving system (4.6) where H satisfies (4.30) and $\hat{b}$ is a vector of all ones

curves of $Q_n(\alpha)$ corresponding to SD converges to the limit, as proved in Lemma 4.2 and Lemma 4.3.

Hence, the optimal parameter in HSS could be actually approximated by SD iterations, which is shown in the following theorem.

Theorem 4.4. *Assume that the matrix H in system* (4.6) *is the Hermitian part of A in system* (4.1)*. If SD is used for solving* (4.6)*, then the following limit holds*

$$\lim_{n\to\infty} \sqrt{\Gamma_n} = \gamma_*. \tag{4.31}$$

Proof. Combining (4.4), (4.23) and the fact that $\Gamma_n > 0$ observed from (4.28), the desired conclusion follows. □

Another approach is to compute the approximation by combining Lemmas 4.2 and 4.3, in which case γ_* could be estimated without explicit access to operator H. This approach is shown in Theorem 4.5.

Theorem 4.5. *Assume that the matrix H in system* (4.6) *is the Hermitian part of A in system* (4.1)*. If SD is used for solving $\mathscr{M}_1 x = \hat{b}$, then the following limit holds*

$$\lim_{n\to\infty} \sqrt{\Gamma_n - \gamma\alpha_n^{\mathrm{RA}} + \gamma} = \gamma_*. \tag{4.32}$$

Proof. Recall that $\mathscr{M}_1 = \gamma I + H$. Since

$$\lambda_i(\mathscr{M}_1) = \gamma + \lambda_i(H)$$

for $i = 1, \ldots, N$, it follows that

$$\begin{aligned}\gamma_* &= \sqrt{\lambda_1(H)\lambda_N(H)} = \sqrt{(\lambda_1(\mathscr{M}_1) - \gamma)(\lambda_N(\mathscr{M}_1) - \gamma)} \\ &= \sqrt{\lambda_1(\mathscr{M}_1)\lambda_N(\mathscr{M}_1) - \gamma(\lambda_1(\mathscr{M}_1) + \lambda_N(\mathscr{M}_1)) + \gamma^2}\end{aligned}$$

Combining (4.19) and (4.23) implies

$$\begin{aligned}\gamma_* &= \sqrt{\lim_{n\to\infty}\Gamma_n - \gamma\lim_{n\to\infty}\alpha_n^{\mathrm{RA}} + \gamma^2} \\ &= \lim_{n\to\infty}\sqrt{\Gamma_n - \gamma\alpha_n^{\mathrm{RA}} + \gamma^2}.\end{aligned}$$

This completes out proof. □

Practically, obtaining γ_* by (4.32) requires a predetermined parameter γ. One could choose $\gamma = 1$ and give an integer k as the maximum number of iterations such that

$$\gamma_* \approx \sqrt{\Gamma_k - \alpha_k^{\mathrm{RA}} + 1},$$

in which case the HSS algorithm might be executed at reduced costs.

Another direction of approach is based on the minimal gradient (MG) steplength

$$\alpha_n^{\mathrm{MG}} = \frac{g_n^{\mathsf{H}} H g_n}{g_n^{\mathsf{H}} H^2 g_n},$$

the spectral properties of which have been discussed in Chapter 3. Let

$$\alpha_n^{\mathrm{A2}} = \left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}}} + \frac{1}{\alpha_n^{\mathrm{MG}}}\right)^{-1}, \quad \tilde{\Gamma}_n = \frac{1}{\alpha_{n-1}^{\mathrm{MG}}\alpha_n^{\mathrm{MG}}} - \frac{g_n^{\mathsf{H}} H g_n}{\left(\alpha_{n-1}^{\mathrm{MG}}\right)^2 g_{n-1}^{\mathsf{H}} H g_{n-1}}.$$

Let $\alpha_n^{\mathrm{RA2}} = \left(\alpha_n^{\mathrm{A2}}\right)^{-1}$. We state the following theorems without proof.

Theorem 4.6. *Assume that the matrix H in system* (4.6) *is the Hermitian part of A in system* (4.1)*. If MG is used for solving* (4.6)*, then the following limit holds*

$$\lim_{n\to\infty} \sqrt{\tilde{\Gamma}_n} = \gamma_*. \tag{4.33}$$

Theorem 4.7. *Assume that the matrix H in system* (4.6) *is the Hermitian part of A in system* (4.1)*. If MG is used for solving $\mathscr{M}_1 x = \hat{b}$, then the following limit holds*

$$\lim_{n\to\infty} \sqrt{\tilde{\Gamma}_n - \gamma\alpha_n^{\mathrm{RA2}} + \gamma} = \gamma_*. \tag{4.34}$$

4.3.3 Lagged variants in low precision

Although SD has remarkable spectral properties, as an iterative method, its popularity has been overshadowed by CG. Akaike [1] exploited the fact that the zig-zag behavior nearly always leads to slow convergence, except when initial gradient approaches an eigenvector. This drawback can be cured with a lagged strategy, first proposed in [17], which was later called Barzilai-Borwein (BB) method. Recall that BB is of the form

$$\alpha_n^{\mathrm{BB}} = \frac{g_{n-1}^{\mathsf{H}} g_{n-1}}{g_{n-1}^{\mathsf{H}} H g_{n-1}}.$$

The convergence analysis was given in [126, 48]. For the Q-linear result, however, has never been proved due to its nonmonotone convergence. It seems overall that the effect of this irregular behavior is beneficial.

For the HSS method, two iterative procedures are needed at each iteration. Since the solution of subproblems in (4.2) is sometimes as difficult as that of the original system (4.1), the inexact solvers with rather low precision are often considered, especially for ill-conditioned problems. Friedlander et al. [74] made the observation that BB could often be competitive with CG when low precision is required. An example is illustrated in Figure 4.2. We solve (4.6) with different residual thresholds ε, where H is chosen as a diagonal matrix of size 10^3 and $\hat{b}$ is a vector of all ones. The diagonal entries have values logarithmically distributed between 10^{-3} and 1 in ascending order, with the first and the last entries equal to the limits, respectively, such that $\kappa(H) = 10^3$. The plot shows a fairly efficient behavior of BB. It is known that CG is sensitive to rounding errors, while lagged gradient methods can remedy this issue [65, 143] with less computational costs per iteration. Additionally, although BB sometimes suffers from the disadvantage of requiring increasing number of iterations for increasing condition numbers, its low-precision behavior tends to be less sensitive to the ill-conditioning.

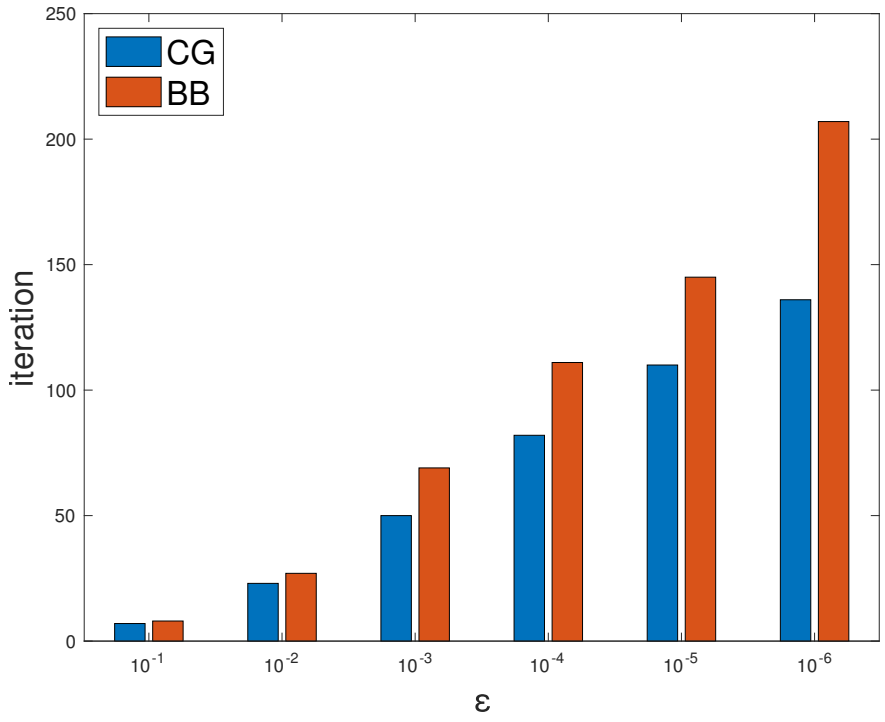

Figure 4.2 Comparison of CG and BB for solving system (4.6) where H is a diagonal matrix of size 10^3 with $\kappa(H) = 10^3$ and $\hat{b}$ is a vector of all ones

4.4 Numerical experiments

In this section we perform some numerical tests with MATLAB R2018b. Assume that iterative algorithms are started from zero vectors. The global stopping criterion in HSS is determined by the threshold $\varepsilon = \|b - Ax_n\| / \|b\|$ with a fixed convergence tolerance 10^{-6}. The inner stopping thresholds ε_1 and ε_2 for the two half-steps of (4.2) are defined in the same way. For gradient iterations applied to system (4.6), similarly, the stopping criterion is defined by the threshold $\varepsilon = \|\hat{b} - Hx_n\|/\|\hat{b}\|$ with the same tolerance. All tests are run in double precision on the computer with 2.8GHz Intel Core i7 central processing unit.

4.4.1 Asymptotic results of gradient iterations

The goal of the first experiment is to illustrate how the spectral properties described earlier can be used for providing a rough estimate of parameter γ_*. We have implemented SD and MG iterations for several real matrices of size 1000 generated by MATLAB routine `sprandsym`. The right-hand side is chosen to be a vector of ones. In Figure 4.3, parameter γ is plotted versus iteration number, under which a red dotted line marks out the position of γ_*. As can be seen, SD with limits (4.31) and (4.32) turns out to be a better strategy than MG in all cases.

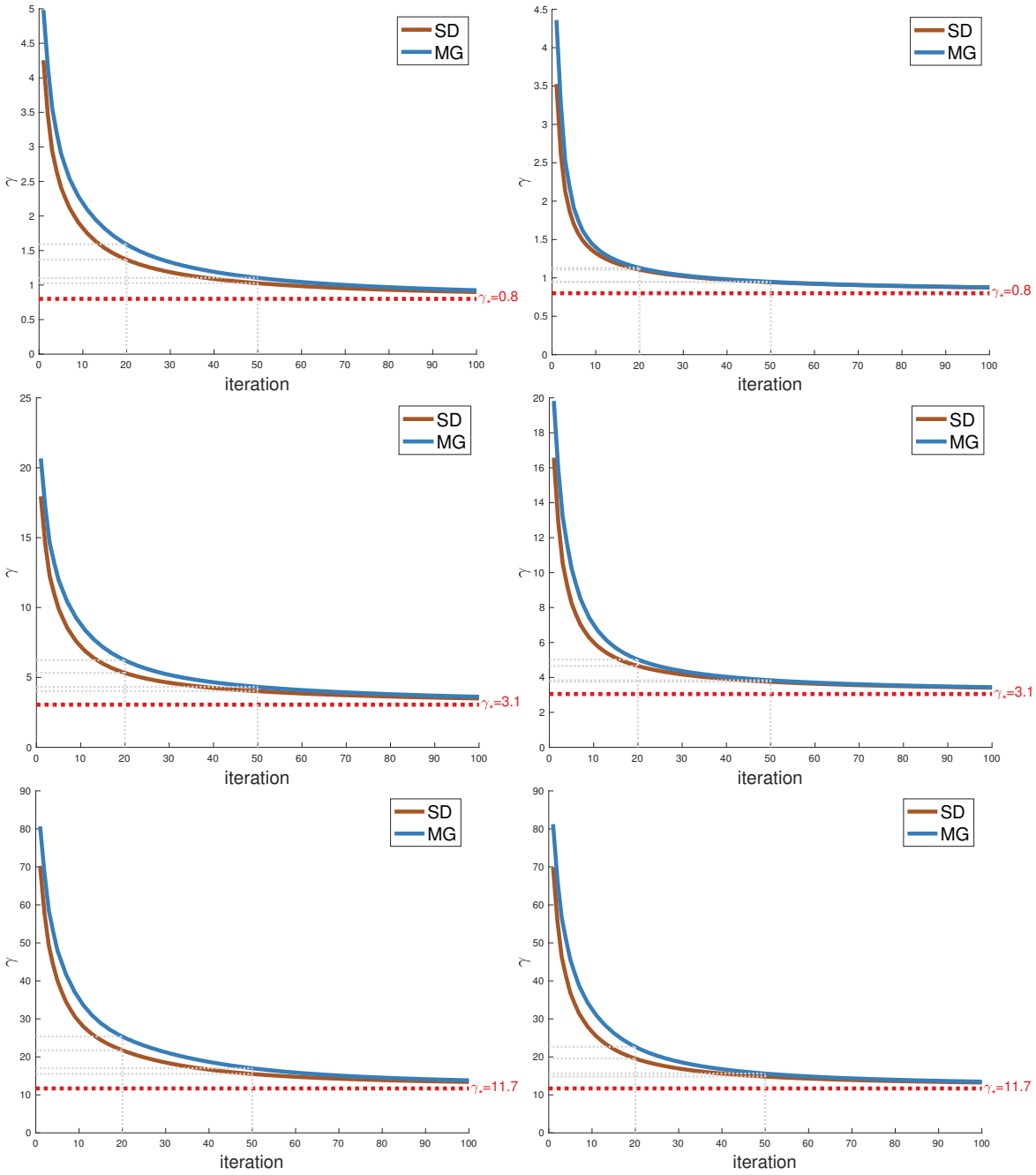

Figure 4.3 Parameter estimation with different matrix H generated randomly by MATLAB: $\gamma_* = 0.8$ (top), $\gamma_* = 3.1$ (middle), $\gamma_* = 11.7$ (bottom). Parameter γ is computed by two approaches: Theorems 4.4 and 4.6 (left), Theorems 4.5 and 4.7 (right)

The indirect approximations based on (4.32) and (4.34) yield faster convergence for both SD and MG.

This test confirms Theorems 4.4 to 4.7. Recall that choosing γ_* as parameter leads to an upper bound of $\rho(T)$, for which it is not necessary to obtain an exact estimate. Experience shows that this choice may sometimes cause overfitting, resulting in slow convergence or even divergence, especially when γ_* is small. One simple measure is to use early stopping in gradient iterations. In the following, it is assumed that SD is used for parameter estimation in HSS, called preadaptive iterations, and we consider only the direct approach (4.31).

4.4.2 HSS with different parameters

In this test we generate some matrices obtained from a classical problem in order to understand the convergence behavior of HSS enhanced by SD iterations.

Example. Consider system (2.1) where A arises from the discretization of differential equation

$$-\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}\right)+\theta\left(\frac{\partial u}{\partial x}+\frac{\partial u}{\partial y}+\frac{\partial u}{\partial z}\right)=q \tag{4.35}$$

on the unit cube $\Omega=[0,1]^3$ with θ a positive constant. Assume that u satisfies homogeneous Dirichlet boundary conditions. The finite difference discretization on a uniform $m\times m\times m$ grid with mesh size $h=1/(m+1)$ is applied to the above model yielding a linear system with $N=m^3$.

In the following we use the centered difference scheme for discretization. The right-hand side b is generated with random components in $[-10,10]+\imath[-10,10]$. As thresholds for inner iterations, $\varepsilon_1=10^{-4}$ and $\varepsilon_2=10^{-4}$ are chosen. CG is exploited for solving the Hermitian inner system, while CGNE (see, e.g., [131]) is for the skew-Hermitian part. Figure 4.4 shows the convergence behavior of HSS with different parameters. Here, we set $\gamma\in[0.5,3.5]$ and $m\in[9,21]$. The optimal parameters γ_* with $m=9,12,15,18,21$ are located by red lines. Notice that a path that zigzags through the bottom of the valley corresponds to the best parameters. As already noted that the parameter estimates need not be accurate, and thus the red lines are good enough in practice.

Then approximating γ_* by inexact SD iterations yields the preadaptive HSS method (PAHSS). Let η denote the number of preadaptive iterations. The convergence behaviors and total computing times are illustrated in Figure 4.5. The left three show the residual curves with several typical choices of η when $m=16,32,64,128$, namely, $N=4096,32768,262144,2097152$. Two observations can be made for all dimensions: the first is that larger η yields faster convergence; the second is that $\eta=100$ does not lead to significant gains in efficiency compared with $\eta=50$. The right three map total wall-clock times of PAHSS iterations, measured in seconds, on to η from 10 to 160. It can be seen that substantial

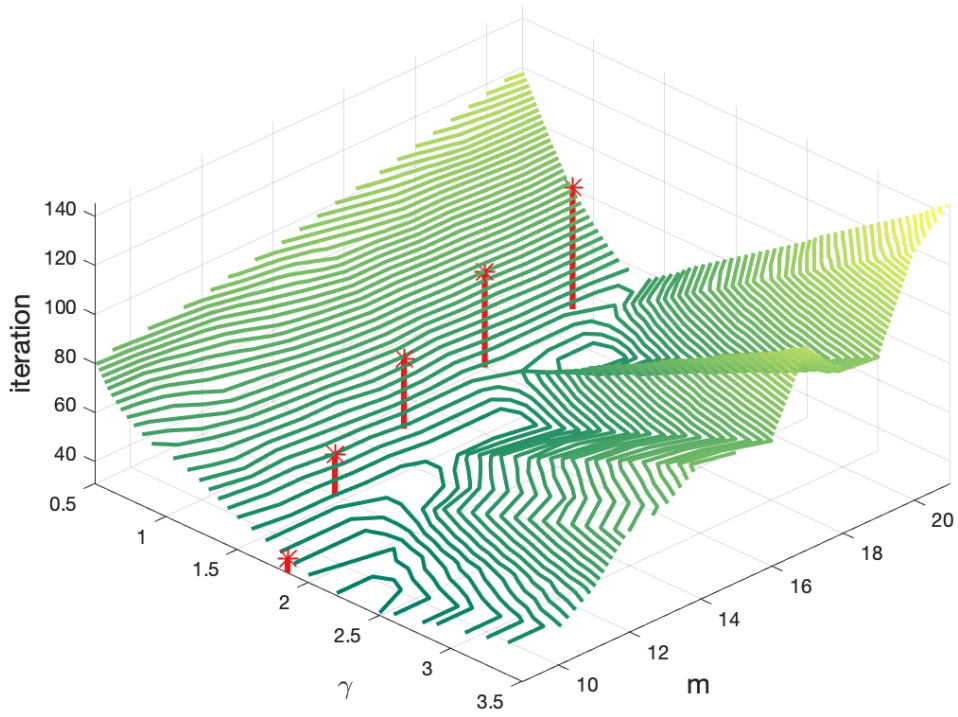

Figure 4.4 Solving problem (4.35) by HSS with $\gamma \in [0.5, 3.5]$ and $m \in [9, 21]$. The optimal parameters γ_* are located by red lines

gains are made in the beginning, following a long period of stagnation. Experience shows that a small number of SD iterations is sufficient and it is therefore appropriate to use early stopping.

4.4.3 CG and BB as low-precision inner solvers

In order to verify that BB can be an alternative to CG as low-precision inner solver for HSS, some tests proceed along the same lines as above but consider both CG and BB as inner solvers for the Hermitian part. Results are shown in Tables 4.1 to 4.3. These three tables illustrate: (1) numbers of outer iterations; (2) numbers of total iterations including the inner ones; (3) wall-clock times, measured in seconds. In Table 4.3, we conduct 10 repeated experiments and print only the average results. As expected, BB is competitive with CG in terms of computing times. As was seen, the optimal parameters γ_* for (4.35) with $m = 10, 20, 30, 40$ are between 1 and 2. BB performs slightly better than CG when $\gamma < \gamma_*$, while the opposite is true when $\gamma > \gamma_*$. On the other hand, large γ leads to slow convergence

Table 4.1 Number of outer HSS iterations problem (4.35) with $\varepsilon_1 = 10^{-1}$ and $\varepsilon_2 = 10^{-4}$.

	$\gamma = 1$				$\gamma = 2$				$\gamma = 5$			
m	10	20	30	40	10	20	30	40	10	20	30	40
CG	85	105	109	117	37	64	111	152	71	143	257	362
BB	72	62	84	91	40	75	117	156	71	143	257	362

Table 4.2 Number of total HSS iterations for problem (4.35) with $\varepsilon_1 = 10^{-1}$ and $\varepsilon_2 = 10^{-4}$.

	$\gamma = 1$				$\gamma = 2$				$\gamma = 5$			
m	10	20	30	40	10	20	30	40	10	20	30	40
CG	386	396	431	496	159	245	399	539	199	359	632	888
BB	410	298	533	578	196	450	776	889	212	391	700	986

Table 4.3 Average execution time (s) of HSS among 10 repeated experiments for problem (4.35) with $\varepsilon_1 = 10^{-1}$ and $\varepsilon_2 = 10^{-4}$.

	$\gamma = 1$				$\gamma = 2$				$\gamma = 5$			
m	10	20	30	40	10	20	30	40	10	20	30	40
CG	0.58	2.16	3.42	5.61	0.25	1.31	3.14	6.51	0.44	2.37	6.80	13.61
BB	0.51	1.39	2.84	4.94	0.28	1.71	3.85	7.91	0.44	2.35	6.98	13.70

in our case. It could smooth out the differences among inner solvers and ensure similar outer iterations as seen in Table 4.1.

4.5 Conclusion

Gradient iterations provide a versatile tool in linear algebra. Apart from parameter estimates related to the spectral properties, SD variants have also been tried recently with success as iterative methods [56, 57, 83]. This chapter extends the spectral properties of gradient iterations and gives an application in HSS. Note that this approach can be extended to other splitting methods (see Section 1 and the references therein) where a parameter γ is to be computed.

Lagged gradient methods play a significant role when nonsymmetric error occurs in the Hessian matrix [143]. This chapter reveals that they are also appealing in situations where low precision is required. The strategy discussed above can be also applied to the skew-Hermitian system. Note that low-precision iterations may suffer from a lack of stability. The common

experience is that it is desirable to choose $\gamma > 1$. Another direction of approach is based on the nonstationary iterative scheme as discussed in [11] in which the inner stopping thresholds satisfy $\varepsilon_j = \max\left\{0.1\delta^n, 10^{-6}\right\}$ where $0 < \delta < 1$ and $j = 1, 2$. On the other hand, CG is practically better than BB as well as other lagged gradient methods when $\varepsilon_1 < 10^{-2}$. For the high-precision solutions of ill-conditioned HPD systems, more precisely, preconditioned CG is still the method of choice.

Our experiments confirm that the gradient-enhanced HSS method can be an attractive alternative to the original one. The stability issue can often be remedied by setting a threshold for the parameter γ. If the problem has large condition number, then a preconditioner should be used for efficiency purposes (see, e.g., [19, 149]). The choice of good preconditioners for non-Hermitian systems remains an open question.

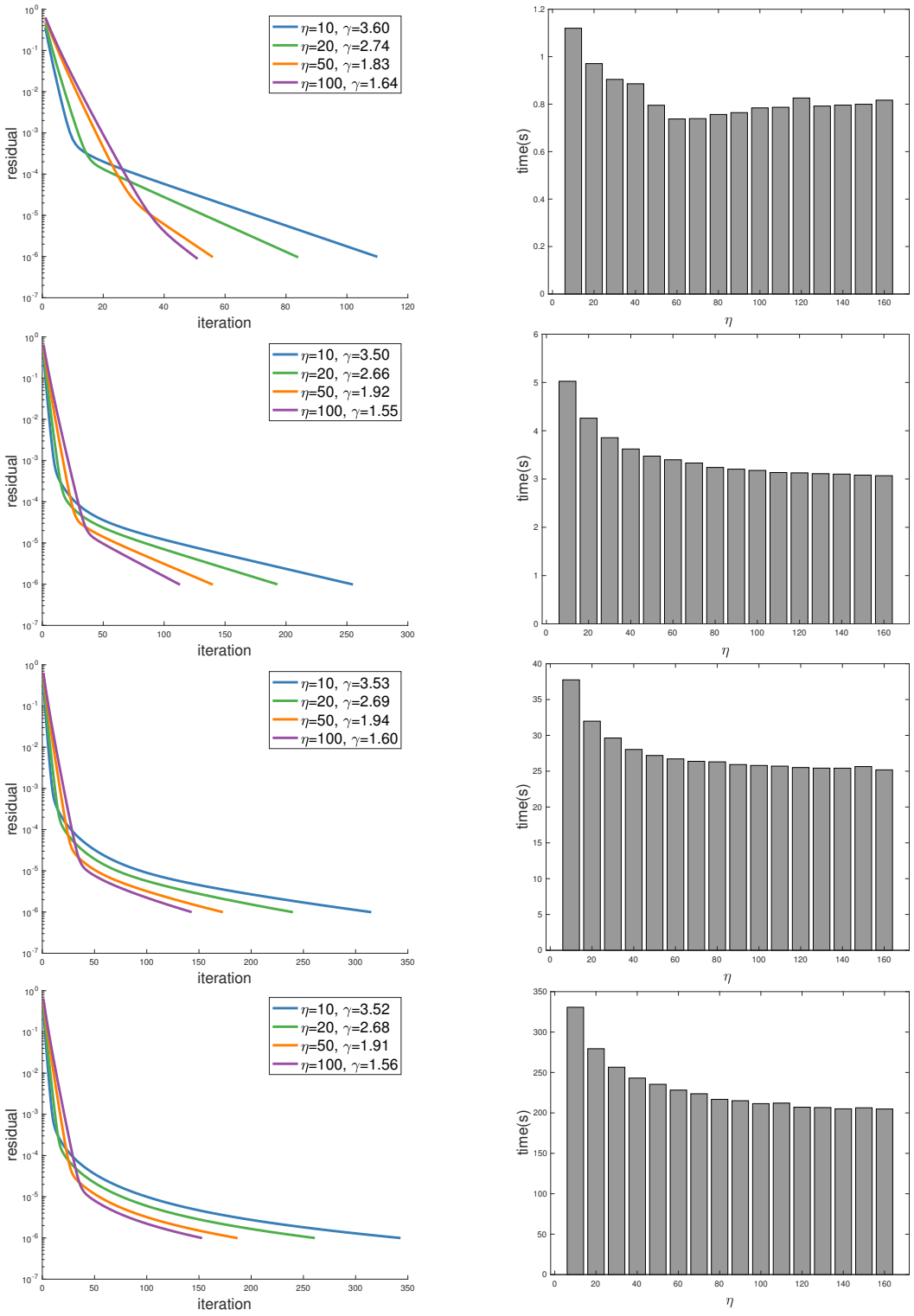

Figure 4.5 Parameter estimation for problem (4.35) with different mesh densities: $m = 16$ (first), $m = 32$ (second), $m = 64$ (third), $m = 128$ (fourth), namely, $N = 4096, 32768, 262144, 2097152$, respectively. Left: convergence curves for different η. Right: average wall-clock times for different η including that of SD iterations

Chapter 5

Reducing communication in lagged gradient methods

Compared with arithmetic operation, communication cost is often the bottleneck on modern computers, and thus should be paid increasing attention when choosing algorithms. Lagged gradient methods are known for their error tolerance and fast convergence. It appears that the parallel behavior of a number of variants is not well understood. In this chapter, we explore the cyclic formulations of lagged gradient methods and s-dimensional methods for reducing global synchronizations. We provide parallel implementations for these methods and propose some new variants. A comparison is then reported for different gradient iterative schemes. To illustrate the performance, we run a number of experiments, from which we conclude that our formulations perform better than traditional methods in terms of both iteration count and computing time.

5.1 Parallel lagged gradient methods

Real-life industrial applications often lead to large-scale linear systems, for which parallel iterative solvers have been studied for decades. The performance of a parallel algorithm is impacted by both floating point operations and communications performed during its execution. On modern computers, however, communication is generally much slower than computation, and this trend is likely to accelerate in future systems.

One way to remedy this issue is to recast parallel synchronous algorithms into asynchronous formulations by breaking up the data dependencies, in which iterations and communications are no longer synchronized. Asynchronous iterations were formally proposed by Chazan and Miranker [35] and generalized by many later authors [113, 18, 62, 23]. Re-

cently, much interest has been focused on programming libraries [7, 43, 105, 107], GPU implementations [3, 36], space domain decomposition methods [109, 110] and time domain decomposition methods [111, 108]. Another direction to reduce communication is based on the s-step Krylov methods, which can often reduce the amount of communications per iteration by a factor of $\mathscr{O}(s)$ (see, e.g., [91]). Chronopoulos and Gear [38] suggested an early specimen for the s-step formulation of the conjugate gradient (CG) method [89], see [37, 96, 97, 41, 39, 40] for the follow-on work. Hoemmen [91] gave a complete treatise of the relevant theory and proposed several new methods, which were later generalized by Carson et al. [31], see also [2, 32, 30, 28, 93, 29]. There are other strategies for reducing communication in iterative methods that can be found in much recent literature, see, e.g. [80, 86, 157, 42]. In this chapter, we suggest some variants of gradient methods for the purpose of reducing synchronization cost, for which parallel algorithms are provided by complying the message passing interface (MPI) specification.

Concerning numerical implementation, one observes that SD and BB have similar operations: matrix-vector multiplications, dot products, and vector updates. In parallel environment, dot products require a combination of local dot products and a reduction over the processors, which incur a costly global synchronization. Let ν be the number of processors and $i \in \{1, \ldots, \nu\}$. Let $x_{i,n}$ and $g_{i,n}$ be the subvectors updated in the ith processors. Assume that the matrix A and the vector b are partitioned into the banded submatrices A_i and the subvectors b_i, respectively. Assume that $x_0 = 0$. This yields the parallel SD method described in Algorithm 5.1. The potential bottleneck is in the reduction and the gather operations. The parallel BB method can be similarly described. If one combines SD and BB, an alternate step (AS) method can be derived (see [45]).

A famous publication by Friedlander et al. [74] considered a framework called gradient methods with retards (GMR) and proved its global convergence. Let m be a positive integer. Let $\bar{n} = \max\{0,\ n-m\}$. Recall that this framework is of the form

$$\alpha_n^{\mathrm{GMR}} = \frac{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)+1} g_{\tau(n)}},$$

with

$$\tau(n) \in \{\bar{n}, \bar{n}+1, \ldots, n-1, n\}, \quad \rho(n) \in \{q_1, \ldots, q_m\}, \quad q_j \geq 0.$$

Notice that choosing $\tau(n) = n$ and $\rho(n) = 0$ yields SD, while $\tau(n) = n-1$ yields BB. Assume that m is the maximum delay. Let $d = m+1$. A communication-avoiding gradient method can be derived from this framework by periodically fixing and recovering $\tau(n) = n$ within d

Algorithm 5.1 Parallel SD method.

1: set $g_{i,0} = -b_i$
2: $\gamma = g_{i,0}^{\mathsf{T}} g_{i,0}$
3: `Allreduce`(γ, `SUM`)
4: compute initial residual
5: **for** $n = 0, 1, \ldots$ **do**
6: `Allgather`(g_n)
7: $p_{i,n} = A_i g_n$
8: $\delta = p_{i,n}^{\mathsf{T}} g_{i,n}$
9: `Allreduce`(δ, `SUM`)
10: $\alpha_n = \gamma / \delta$
11: $x_{i,n+1} = x_{i,n} - \alpha_n g_{i,n}$
12: $g_{i,n+1} = g_{i,n} - \alpha_n p_{i,n}$
13: $\gamma = g_{i,n+1}^{\mathsf{T}} g_{i,n+1}$
14: `Allreduce`(γ, `SUM`)
15: compute residual
16: **end for**

iterations, leading to the cyclic steepest descent (CSD) method

$$\alpha_n^{\mathrm{CSD}} = \alpha_{\tau(n)}^{\mathrm{SD}}, \quad \tau(n) = \max\{j \le n : j \bmod d = 0\},$$

with $d \ge 1$. AS is indeed a special case of CSD when choosing $d = 2$. Further experiments in Friedlander et al. [74] confirmed that CSD is competitive with BB in the sequential case. The parallel implementation is described in Algorithm 5.2. In the parallel case, CSD reduces $d-1$ or $2(d-1)$ dot product operations in every d iterations depending on the implementation. If one computes only local residual in each iteration, then the algorithm could be further improved by moving line 14 into the code block starting from line 5 in Algorithm 5.2.

It was De Asmundis et al. [56] who suggested to use the second order information based on some spectral properties of SD to accelerate the gradient iterations. They proposed in [56] and [57] two new methods called SD with alignment (SDA) and SD with constant steplength (SDC), respectively. We only discuss the latter since they share similar properties, and it is the experience of the present authors that SDC generally performs better. The step is of the

Algorithm 5.2 Parallel CSD method.

same as lines 1 to 4 in Algorithm 5.1
for $n = 0, 1, \ldots$ **do**
    `Allgather`(g_n)
    $p_{i,n} = A_i g_n$
    if $n \bmod d = 0$ **then**
        $\delta = p_{i,n}^{\mathsf{T}} g_{i,n}$
        `Allreduce`(δ, `SUM`)
        $\alpha_n = \gamma/\delta$
        $r = 0$
    end if
    $x_{i,n+1} = x_{i,n} - \alpha_{n-r} g_{i,n}$
    $g_{i,n+1} = g_{i,n} - \alpha_{n-r} p_{i,n}$
    $r = r + 1$
    $\gamma = g_{i,n+1}^{\mathsf{T}} g_{i,n+1}$
    `Allreduce`(γ, `SUM`)
    compute residual
end for

form

$$\alpha_n^{\mathrm{SDC}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod (d_1 + d_2) < d_1, \\ \alpha_{\tau(n)}^{\mathrm{Y}}, & \tau(n) = \max\{j \le n : j \bmod (d_1 + d_2) = d_1\}, \end{cases}$$

where $d_1, d_2 \ge 1$ and α_n^{Y} has been defined in Section 2.3. Here, the Yuan step was introduced in Yuan [154]. The main feature of SDC is to foster the reduction of gradient components along the eigenvectors of A selectively, and thus reduce the search space into smaller and smaller dimensions. As a result, the problem tends to have a better and better condition number (see [57]). The parallel algorithm is illustrated in Algorithm 5.3. In parallel case, this method reduces $d_2 - 1$ or $2(d_2 - 1)$ dot product operations in every $d_1 + d_2$ iterations.

One finds that both CSD and SDC are cyclic gradient methods. A cycle equals d iterations in CSD and $d_1 + d_2$ in SDC. It is also possible to define minimal gradient (MG) approaches (see, e.g., [50]) in a cyclic form. We will not investigate these variants further since MG and SD share similar properties in both sequential and parallel views.

Algorithm 5.3 Parallel SDC method.

1: same as lines 1 to 4 in Algorithm 5.1
2: **for** $n = 0, 1, \ldots$ **do**
3: `Allgather`(g_n)
4: $p_{i,n} = A_i g_n$
5: **if** $n \bmod (d_1 + d_2) < d_1$ **then**
6: $\delta = p_{i,n}^{\mathsf{T}} g_{i,n}$
7: `Allreduce`$(\delta,$ `SUM`$)$
8: $\alpha_n = \gamma/\delta$
9: $\theta = \gamma$
10: $r = 0$
11: **else if** $n \bmod (d_1 + d_2) = d_1$ **then**
12: $\delta = p_{i,n}^{\mathsf{T}} g_{i,n}$
13: `Allreduce`$(\delta,$ `SUM`$)$
14: $\hat{\alpha} = \gamma/\delta$
15: $\alpha_n = 2/(\sqrt{(1/\alpha_{n-1} - 1/\hat{\alpha})^2 + 4\gamma/(\alpha_{n-1}^2\theta)} + 1/\alpha_{n-1} + 1/\hat{\alpha})$
16: $r = 0$
17: **end if**
18: same as lines 11 to 16 in Algorithm 5.2
19: **end for**

5.2 Parallel s-dimensional steepest descent

From Section 2.1, it is known that there exist real values $\zeta_{j,n}$ such that

$$g_n = \sum_{j=1}^{N} \zeta_{j,n} v_j.$$

Using Equation (2.4), we get

$$\zeta_{j,n+1} = (1 - \alpha_n \lambda_j)\zeta_{j,n} = \zeta_{j,0} \prod_{k=0}^{n} (1 - \alpha_k \lambda_j),$$

where $j \in \{1, \ldots, N\}$. It is clear that choosing $\alpha_n = 1/\lambda_j$ leads to the fact that the corresponding component of the gradient vector will vanish and remain zero throughout the

iteration. Now we rewrite the solution vector recurrence in the following form

$$x_{n+1} = x_n - P(A)g_n.$$

By the Cayley-Hamilton theorem, one finds that A^{-1} could be described as a matrix polynomial of degree $s-1$. Let $P(A)$ be a polynomial of the form

$$P(A) = \alpha_n^{(1)} I + \alpha_n^{(2)} A + \cdots + \alpha_n^{(s)} A^{s-1},$$

where I is an identity matrix of size N. We consider the s-dimensional plane

$$L_n^{(s)} = \left\{ x_n - \sum_{j=1}^{s} \alpha_n^{(j)} A^{j-1} g_n : \alpha_n^{(j)} \in \mathbb{R} \right\}. \tag{5.1}$$

We remark that choosing $s = 1$ will recover the search spaces of gradient methods. The s-dimensional steepest descent (s-SD) method is of the form

$$x_{n+1} = x_n - \alpha_n^{(1)} g_n - \cdots - \alpha_n^{(s)} A^{s-1} g_n,$$

where $\alpha_n^{(1)}, \ldots, \alpha_n^{(s)}$ are selected to minimize the quadratic function in Equation (2.2) or the A-norm error $\|e_{n+1}\|_A$ over $L_n^{(s)}$. The recurrence of gradient vector can be written as

$$g_{n+1} = g_n - \alpha_n^{(1)} A g_n - \cdots - \alpha_n^{(s)} A^s g_n.$$

It is clear that g_{n+1} should be orthogonal to the s-dimensional space $L_n^{(s)}$. As a result, one finds the following system of equations

$$\begin{array}{ccccccc}
\alpha_n^{(1)} g_n^\intercal A g_n & + & \ldots & + & \alpha_n^{(s)} g_n^\intercal A^s g_n & = & g_n^\intercal g_n, \\
\alpha_n^{(1)} (Ag_n)^\intercal A g_n & + & \ldots & + & \alpha_n^{(s)} (Ag_n)^\intercal A^s g_n & = & (Ag_n)^\intercal g_n, \\
\vdots & & & & \vdots & & \vdots \\
\alpha_n^{(1)} (A^{s-1} g_n)^\intercal A g_n & + & \ldots & + & \alpha_n^{(s)} (A^{s-1} g_n)^\intercal A^s g_n & = & (A^{s-1} g_n)^\intercal g_n.
\end{array}$$

Let $\omega_n^{(j)} = g_n^{\intercal} A^j g_n$. Since $(A^k g_n)^{\intercal} A^l g_n = g_n^{\intercal} A^{k+l} g_n$, it follows that

$$\begin{pmatrix} \omega_n^{(1)} & \omega_n^{(2)} & \cdots & \omega_n^{(s)} \\ \omega_n^{(2)} & \omega_n^{(3)} & \cdots & \omega_n^{(s+1)} \\ \vdots & \vdots & & \vdots \\ \omega_n^{(s)} & \omega_n^{(s+1)} & \cdots & \omega_n^{(2s-1)} \end{pmatrix} \begin{pmatrix} \alpha_n^{(1)} \\ \alpha_n^{(2)} \\ \vdots \\ \alpha_n^{(s)} \end{pmatrix} = \begin{pmatrix} \omega_n^{(0)} \\ \omega_n^{(1)} \\ \vdots \\ \omega_n^{(s-1)} \end{pmatrix}. \tag{5.2}$$

The matrix of the system (5.2) is denoted by Ω_n.

Some early works of the s-dimensional steepest descent (s-SD) method can be found in Birman [26], Khabaza [95] and Forsythe [68]. In 1950, Birman [26] gave a proof for the convergence by virtue of the orthogonal polynomial. The original literature was written in Russian, see Equation (2.14) in [68] for an English discussion, where the asymptotic behavior is also shown therein. The following theorem gives a lower bound for the parameter which has the largest magnitude, acting as a complement to the existing theories.

Theorem 5.1. *Consider the linear system $Ax = b$ where A is SPD. Let $\Gamma_n = \{\alpha_n^{(1)}, \ldots, \alpha_n^{(s)}\}$. If the sequence of solution vectors $\{x_n\}$ is generated by the s-dimensional SD method, then*

$$\max_{\alpha \in \Gamma_n} |\alpha| \geq \frac{1}{\lambda_N + \cdots + \lambda_N^s}$$

Proof. Let

$$\hat{\alpha} = \max_{\alpha \in \Gamma_n} |\alpha|.$$

By assumption,

$$\omega_n^{(0)} = \omega_n^{(1)} \alpha_n^{(1)} + \cdots + \omega_n^{(s)} \alpha_n^{(s)} \leq \omega_n^{(1)} \hat{\alpha} + \cdots + \omega_n^{(s)} \hat{\alpha}.$$

Since

$$\frac{\omega_n^{(j)}}{\omega_n^{(0)}} = \frac{\omega_n^{(j)}}{\omega_n^{(j-1)}} \cdot \cdots \cdot \frac{\omega_n^{(1)}}{\omega_n^{(0)}} \leq \lambda_N^j,$$

we get

$$1 \leq \lambda_N \hat{\alpha} + \cdots + \lambda_N^s \hat{\alpha},$$

which yields the desired result. □

Let $u_n^{(j)} = A^{(2j-1)/2} g_n$. One finds that

$$\Omega_n = \begin{pmatrix} (u_n^{(1)})^{\intercal} u_n^{(1)} & (u_n^{(1)})^{\intercal} u_n^{(2)} & \cdots & (u_n^{(1)})^{\intercal} u_n^{(s)} \\ (u_n^{(2)})^{\intercal} u_n^{(1)} & (u_n^{(2)})^{\intercal} u_n^{(2)} & \cdots & (u_n^{(2)})^{\intercal} u_n^{(s)} \\ \vdots & \vdots & & \vdots \\ (u_n^{(s)})^{\intercal} u_n^{(1)} & (u_n^{(s)})^{\intercal} u_n^{(2)} & \cdots & (u_n^{(s)})^{\intercal} u_n^{(s)} \end{pmatrix}.$$

Therefore, Ω_n is a Gram matrix, and thus a positive definite Hankel matrix, which can be solved by Cholesky factorization. This variant involves the computational work of $4sN$ multiplications, $4sN$ additions, s matrix-vector operations and the cost of solving a positive definite Hankel system, for which the Cholesky factorization requires approximately $(1/3)s^3 + (1/2)s^2$ arithmetic operations. On the other hand, a simple SD algorithm over s iterations requires $4sN$ multiplications, $4sN$ additions, s matrix-vector operations and s divisions. It is well-known that the Hankel matrix is ill-conditioned for which the condition number has a lower bound $\kappa \geq 3 \cdot 2^{s-6}$ (see, e.g., [142]). In the finite precision case, the monomial basis may become linearly dependent asymptotically when s is large. As a result, the ill-conditioning of Ω_n makes the s-SD algorithm unstable. Concerning parallel implementation, however, breaking the data dependency between matrix-vector multiplications and dot products makes the s-dimensional method attractive. Now we design the parallel s-SD method, see Algorithm 5.4. In the parallel case, s-SD requires 2 reduction operations and s gather operations, while SD requires $2s$ reduction operations and s gather operations over s iterations. Under the reasonable assumption that the computational environment is latency-bound, the redundant operations of s-SD will not be a limiting factor. Additionally, since $N \gg s$ for large sparse matrix A, the computational cost of Cholesky factorization is unlikely to be a bottleneck.

5.3 New lagged methods

According to Friedlander et al. [74], the gradient methods with retards usually perform much better than the traditional optimal gradient methods. We have the following observations:

- in Forsythe [68], we know that both SD and s-SD have two-step invariance property, which makes the directions sink into lower subspaces, while lagged gradient steps could remedy such problem (see, e.g., [51]);

Algorithm 5.4 Parallel s-SD method.

set $g_{i,0} = -b_i$
$\omega_0^{(0)} = g_{i,0}^{\mathsf{T}} g_{i,0}$
`Allreduce`($\omega_0^{(0)}$, `SUM`)
compute initial residual
for $n = 0, 1, \ldots$ **do**
    set $u_{i,n} = g_{i,n}$
    for $j = 1, \ldots, s$ **do**
        `Allgather`(u_n)
        $p_{i,n}^{(j)} = A_i u_n$
        set $u_{i,n} = p_{i,n}^{(j)}$
    end for
    compute $\omega_n^{(1)}, \ldots, \omega_n^{(2s-1)}$
    `Allreduce`($[\omega_n^{(1)} \ldots \omega_n^{(2s-1)}]$, `SUM`)
    solve system (5.2)
    $x_{i,n+1} = x_{i,n} - \alpha_n^{(1)} g_{i,n} - \alpha_n^{(2)} p_{i,n}^{(1)} - \cdots - \alpha_n^{(s)} p_{i,n}^{(s-1)}$
    $g_{i,n+1} = g_{i,n} - \alpha_n^{(1)} p_{i,n}^{(1)} - \alpha_n^{(2)} p_{i,n}^{(2)} - \cdots - \alpha_n^{(s)} p_{i,n}^{(s)}$
    $\omega_{n+1}^{(0)} = g_{i,n+1}^{\mathsf{T}} g_{i,n+1}$
    `Allreduce`($\omega_{n+1}^{(0)}$, `SUM`)
    compute residual
end for

- unlike CG, gradient methods are less sensitive to the rounding error since no sequence of conjugate directions is constructed;
- both computation and communication for dot products can be cyclically curtailed by imposing lagged steps.

It may be possible to add a slight delay in successive s-SD iterations, from which the induced rounding error is generally acceptable. Recall that a scalar matrix is a diagonal matrix where all elements are equal. The following theorem reveals the signs of parameters in 2-dimensional case.

Theorem 5.2. *Consider the linear system $Ax = b$ where A is SPD. Assume that A is not a scalar matrix. If the sequence of solution vectors $\{x_n\}$ is generated by the 2-dimensional SD method, then $\alpha_n^{(1)} > 0$ and $\alpha_n^{(2)} < 0$.*

Proof. The system (5.2) is reduced to

$$\begin{pmatrix} \omega_n^{(1)} & \omega_n^{(2)} \\ \omega_n^{(2)} & \omega_n^{(3)} \end{pmatrix} \begin{pmatrix} \alpha_n^{(1)} \\ \alpha_n^{(2)} \end{pmatrix} = \begin{pmatrix} \omega_n^{(0)} \\ \omega_n^{(1)} \end{pmatrix}.$$

Il follows that

$$\alpha_n^{(1)} = \frac{\omega_n^{(0)}\omega_n^{(3)} - \omega_n^{(1)}\omega_n^{(2)}}{\omega_n^{(1)}\omega_n^{(3)} - \left(\omega_n^{(2)}\right)^2}.$$

$$\alpha_n^{(2)} = \frac{\left(\omega_n^{(1)}\right)^2 - \omega_n^{(0)}\omega_n^{(2)}}{\omega_n^{(1)}\omega_n^{(3)} - \left(\omega_n^{(2)}\right)^2}.$$

By Cauchy-Schwarz inequality, we observe that

$$\begin{aligned} \frac{g_n^\intercal A^{j+1} g_n}{g_n^\intercal A^{j+2} g_n} &\leq \frac{\left\|A^{j/2} g_n\right\| \left\|A^{(j+2)/2} g_n\right\|}{\left\|A^{(j+2)/2} g_n\right\|^2} \\ &= \frac{\left\|A^{j/2} g_n\right\|^2}{\left\|A^{(j+2)/2} g_n\right\| \left\|A^{j/2} g_n\right\|} \leq \frac{g_n^\intercal A^j g_n}{g_n^\intercal A^{j+1} g_n}. \end{aligned} \tag{5.3}$$

Moreover, by assumption we know that A is not a scalar matrix, yielding that $A^{j/2} g_n$ and $A^{(j+2)/2} g_n$ are linearly independent. Thus, the strict inequality holds.

Now consider the denominators

$$\omega_n^{(1)}\omega_n^{(3)} = \frac{\omega_n^{(1)}}{\omega_n^{(2)}} \cdot \frac{\omega_n^{(2)}}{\omega_n^{(3)}} \cdot \left(\omega_n^{(3)}\right)^2 > \left(\frac{\omega_n^{(2)}}{\omega_n^{(3)}}\right)^2 \cdot \left(\omega_n^{(3)}\right)^2 = \left(\omega_n^{(2)}\right)^2.$$

Similarly, consider the numerators

$$\omega_n^{(0)}\omega_n^{(3)} = \frac{\omega_n^{(0)}}{\omega_n^{(1)}} \cdot \omega_n^{(1)}\omega_n^{(2)} \cdot \frac{\omega_n^{(3)}}{\omega_n^{(2)}} > \frac{\omega_n^{(2)}}{\omega_n^{(3)}} \cdot \frac{\omega_n^{(3)}}{\omega_n^{(2)}} \cdot \omega_n^{(1)}\omega_n^{(2)} = \omega_n^{(1)}\omega_n^{(2)},$$

and

$$\omega_n^{(0)}\omega_n^{(2)} > \left(\omega_n^{(1)}\right)^2.$$

This completes our proof. □

From Theorem 5.2, one finds that the components obtained by system (5.2) may be less than zero, and thus could not serve as lagged steps. Another direction of approach is based on

Algorithm 5.5 Parallel Cs-SD method.

1: same as lines 1 to 4 in Algorithm 5.4
2: **for** $n = 0, 1, \ldots$ **do**
3: **if** n mod $d = 0$ **then**
4: same as lines 6 to 16 in Algorithm 5.4
5: $r = 1$
6: **else**
7: `Allgather`(g_n)
8: $p_{i,n} = A_i g_n$
9: compute $\hat{\alpha}_n$ by $\omega_{n-r}^{(1)}, \ldots, \omega_{n-r}^{(2s-1)}$
10: $x_{i,n+1} = x_{i,n} - \hat{\alpha}_n g_{i,n}$
11: $g_{i,n+1} = g_{i,n} - \hat{\alpha}_n p_{i,n}$
12: $r = r + 1$
13: **end if**
14: same as lines 17 to 19 in Algorithm 5.4
15: **end for**

the steps of the traditional gradient methods, fortunately, the parameters $\omega_n^{(j)}$ being available. The real scalars

$$\hat{\alpha}_n = \frac{\omega_{n-r}^{(j)}}{\omega_{n-r}^{(j+1)}} = \frac{g_{n-r}^{\mathsf{T}} A^j g_{n-r}}{g_{n-r}^{\mathsf{T}} A^{j+1} g_{n-r}}$$

can be constructed, where $1 \leq r < d$ and $0 \leq j < 2s-1$, and embedded in the iterative process. It is noteworthy that such kind of steps is the inverse of Rayleigh quotient and satisfies the form of Equation (2.6). Like the previously described methods, we illustrate the parallel implementation of this variant in Algorithm 5.5, called cyclic s-SD (Cs-SD) method. We observe that a cycle involves d iterations in which an s-SD process is performed at the beginning of the cycle and $d-1$ lagged gradient iterations are performed thereafter.

There exist several versions of the algorithm that can be effectively used for updating the solution and the gradient vectors due to the redundant elements of $\omega_n^{(j)}$. We give examples for the CSD-like version and the damped version. The CSD-like step is defined by choosing $j = 0$ for all $r \in \{1, \ldots, d-1\}$, which is equivalent to the CSD method except when n mod $d = 0$. The damped step means that the steplengths become smaller and smaller during lagged

iterations. From Equation (5.3) and by the fact that A is not a scalar matrix,

$$\frac{\omega_n^{(2s-2)}}{\omega_n^{(2s-1)}} < \cdots < \frac{\omega_n^{(0)}}{\omega_n^{(1)}},$$

and thus choosing $j = r-1$ yields the damped Cs-SD method.

In average, the computational work in each iteration consists of $2(d+1)sN/d$ multiplications, $2(d+1)sN/d$ additions, s matrix-vector products and $\mathscr{O}(s^3/d)$ operations to solve the positive definite Hankel system. The construction of scalars in lagged iterations also brings the computational cost of divisions. Nevertheless, we are more concerned with the communication cost in the parallel case, which on average consists of $(d+1)/d$ reduction operations and $(s+d-1)/d$ gather operations. We remark that $d = 1$ leads to exactly the case of s-SD in terms of the communication cost. If we move the line 14 into the code block starting from line 3, merge the reduction operations and compute local residual instead in Algorithm 5.5, Cs-SD requires $1/d$ reduction operations and $(s+d-1)/d$ gather operations, while 1 and s operations for s-SD, respectively.

It is challenging to calculate the potential speedup. For example, under the assumption that both reduction and gather operations have the same latency, which is assumed as the major bottleneck that impacts the performance of algorithms, then we expect to see a $(sd+d)/(s+d)$ times less communication of Algorithm 5.5 with respect to Algorithm 5.4 when computing local residual per iteration. The drawback of this analysis is that an s-SD iteration is generally not equivalent to a lagged step iteration. A lagged iteration usually gives some orders of magnitude more contributions to the convergence speed than an s-SD iteration when n is large. On the other hand, the comparison of classical gradient methods is obvious. Here, we only discuss the most communication-hiding situation. If we merge the reduction operations, then both SD and BB require 1 reduction operations and 1 gather operation. For the CSD method, on average, the communication cost consists of $1/d$ reduction operations and 1 gather operation. Under the above assumption, choosing $d = 2$ yields a 1.3 times less communication.

Similar to classical CSD, classical SDC generates cyclic iterations which allow a reduction in communication cost. The following inequality is easy to show

$$\alpha_n^{\mathrm{Y}} < \min\left\{\alpha_{n-1}^{\mathrm{SD}}, \alpha_n^{\mathrm{SD}}\right\}.$$

This leads to a rather smooth convergence. The idea of the s-SD with constant steplength (s-SDC) method can be interpreted as a process in which s-SD takes place at the beginning of a cycle, and then one proceeds with the cyclic Yuan step over $d-1$ iterations. We illustrated

Algorithm 5.6 Parallel s-SDC method.

1: same as lines 1 to 4 in Algorithm 5.4
2: **for** $n = 0, 1, \ldots$ **do**
3: **if** $n \bmod d = 0$ **then**
4: same as lines 6 to 16 in Algorithm 5.4
5: $\tilde{\alpha} = \omega_n^{(0)} / \omega_n^{(1)}$
6: $\theta = \omega_n^{(0)}$
7: **else**
8: `Allgather`(g_n)
9: $p_{i,n} = A_i g_n$
10: **if** $n \bmod d = 1$ **then**
11: $\delta = p_{i,n}^{\mathsf{T}} g_{i,n}$
12: `Allreduce`$(\delta,$ `SUM`$)$
13: $\hat{\alpha} = \omega_n^{(0)} / \delta$
14: $\alpha_n = 2/(\sqrt{(1/\tilde{\alpha} - 1/\hat{\alpha})^2 + 4\omega_n^{(0)}/(\tilde{\alpha}^2\theta)} + 1/\tilde{\alpha} + 1/\hat{\alpha})$
15: $r = 0$
16: **end if**
17: $x_{i,n+1} = x_{i,n} - \alpha_{n-r} g_{i,n}$
18: $g_{i,n+1} = g_{i,n} - \alpha_{n-r} p_{i,n}$
19: $r = r + 1$
20: **end if**
21: same as lines 17 to 19 in Algorithm 5.4
22: **end for**

the parallel algorithm of s-SDC in Algorithm 5.6. An obvious drawback of this method is that one more dot product operation is required over a cycle. This will increase the average cost in terms of both computation and communication. Since Yuan step plays an important role in the alignment methods, we expect to see that the superior convergence behavior can remedy the additional cost per cycle.

Now we make the following simplifying assumptions about algorithms:

- communication is the major bottleneck compared with computation, in which latency plays a dominant role compared with bandwidth;
- we count reduction and gather operations as yielding the same communication cost;

Table 5.1 Average number of reduction and gather operations (ro and go). 1 iteration here equals 1 iteration in CSD process or $1/s$ iteration in s-SD process.

	SD	s-SD	CSD	SDC	Cs-SD	s-SDC
ro	1	$1/s$	$1/d$	$(d_1+1)/(d_1+d_2)$	$1/(s+d-1)$	$2/(s+d-1)$
go	1	1	1	1	1	1

- we do not use blocking synchronization for the computation of residual;
- $1/s$ iteration in s-SD process is regarded as equivalent to 1 iteration in SD or CSD process.

Although the methods discussed above are pairwise distinct in a mathematical point of view, these assumptions enable a comparison of them, which is shown in Table 5.1. Since we have not exploited specific properties of A, it is necessary to perform a gather operation in each step explicitly. Concerning reduction operations, both parameters s and d lead to gains in communication avoidance. We observe that SDC is special because it was motivated by the spectral properties of SD, involving d_1 iterations to foster alignment (see [57]), and thus has a parameter in the numerator. It appears that this approach could be more efficient, albeit at the cost of increasing communication with respect to CSD.

5.4 Comparison of parallel gradient schemes

The straightforward parallel gradient scheme, illustrated in Algorithm 5.7, consists of two global synchronizations: one before the calculation of α_n and the other for g_n. Unlike previous algorithms, we show here explicitly the nonblocking version, resulting in limited performance gains by overlapping communication with computation. It is known that the data dependencies in gradient methods preclude further improvement of performance. In one direction, the lagged iterative scheme is a communication-avoiding strategy allowing an $\mathscr{O}(d)$ reduction in communication cost. On the other hand, it does not verify the optimality properties in gradient methods, and lead to drastic improvements in convergence. The parallel lagged gradient scheme is shown in Algorithm 5.8.

If we remove lines 2 and 3 in Algorithm 5.7 or lines 2 to 5 in Algorithm 5.5, i.e., a constant parameter $\hat{\alpha} = \alpha_n$ is used for updating $x_{i,n+1}$ and $g_{i,n+1}$, then we get the parallel constant gradient scheme. The sequential case was first considered by Richardson [130] in a more general form, and thus could be called stationary Richardson iteration (see, e.g., [152]). This scheme can be further improved by removing the waiting statement, yielding a so-called

Algorithm 5.7 Parallel gradient scheme.

1: compute $A_i g_n$
2: compute and synchronize coefficients
3: compute α_n
4: compute $x_{i,n+1}$, $g_{i,n+1}$
5: **for** $j \in$ dependent neighbors **do**
6: send g_i to processor j
7: **end for**
8: **for** $j \in$ essential neighbors **do**
9: receive g_j from processor j
10: **end for**
11: compute residual
12: wait for requests to finish

asynchronous gradient scheme, as described in Algorithm 5.9. Asynchronous iterations were first proposed by Chazan and Miranker [35]. Let $\mathscr{F}_i$ be a mapping in processor i. Assume that $\mathscr{P}_n \subset \{1, \ldots, v\}$ is a subset of processors in the nth iteration. Such iterative scheme can be expressed in the following form

$$x_{i,n+1} = \begin{cases} \mathscr{F}_i\left(x_{1,\hat{\tau}_{i,1}(n)}, \ldots, x_{v,\hat{\tau}_{i,v}(n)}\right), & i \in \mathscr{P}_n, \\ x_{i,n}, & \text{otherwise,} \end{cases}$$

where $\hat{\tau}_{i,j}(n)$ is a positive integer satisfying $\hat{\tau}_{i,j}(n) \leq n$ for each element j in each processor i. It avoids communication by breaking up the data dependencies in iterative methods but still allows updates when messages arrive. In order to hide global synchronization, a well-designed non-blocking technique is required to evaluate residual, see [135, 8, 106] for a discussion.

Although the asynchronous gradient scheme can minimize waiting time among v processors, the convergence conditions may be more stringent than those for the lagged gradient scheme. The basic conditions for A at the beginning of Section 2.1 should suffice for the latter in exact arithmetic, while the condition $\rho(|I - \hat{\alpha}A|) < 1$ is required for the former (see Section 6.3.2 in [24] for a discussion). Another drawback is that the asynchronous formulation is based on the stationary Richardson method, which is generally far less efficient than the lagged gradient methods. Hoemmen [91] argued that asynchronous methods tend to be based on slow relaxation-type iterations. In other words, synchronous advanced

Algorithm 5.8 Parallel cyclic gradient scheme.

compute $A_i g_n$
if $n \bmod d = 0$ **then**
    compute and synchronize coefficients
    compute α_n
end if
compute $x_{i,n+1}$, $g_{i,n+1}$
for $j \in$ dependent neighbors **do**
    send g_i to processor j
end for
for $j \in$ essential neighbors **do**
    receive g_j from processor j
end for
compute residual
wait for requests to finish

methods tend to perform better than asynchronous basic methods. However, as mentioned in Section 5.1, asynchronous iterations are still viewed as promising, especially when communication costs become prohibitive. We will not pursue this scheme further in this chapter. For good overviews, see [77, 9, 78].

5.5 Some practical considerations

The algorithms mentioned in Sections 5.1 to 5.4 are described in an MPI-like manner. If the matrix size is not a power of two, then one should invoke `Allgatherv`, a routine defined in the MPI specification for assembling data with displacement of indices, to collect matrix-vector multiplication results. For the global synchronization before α_n, it appears that the use of nonblocking communication does not lead to performance gains due to the data dependencies. If the MPI programming libraries are optimally implemented, then the collective routine `Allreduce` is somewhat preferred to the point-to-point operations. Thakur et al. [141] has shown how to improve the performance of collective communication operations. On the other hand, 2 reduction operations result in 2 times latency costs, and thus could be avoided by constructing 1 message for 2 values. The termination detection may become complicated. For example, global convergence could be evaluated by a nonblocking reduction of the local residual. In this case, however, a delayed snapshot of global state without redundant solution

Algorithm 5.9 Asynchronous gradient scheme.

1: compute $A_i g_n$
2: compute $x_{i,n+1}$, $g_{i,n+1}$
3: **for** $j \in$ dependent neighbors **do**
4: send g_i to processor j
5: **end for**
6: **for** $j \in$ essential neighbors **do**
7: receive g_j from processor j
8: **end for**
9: compute residual

vector being stored may lead to a suspicion about the final residual, as well as a delay in the termination detection. A more robust solution is based on the blocking reduction of residual required by α_j where $j > n$. One could certainly make a snapshot of solution vector at the cost of increasing computer storage. For the gather operation, overlapping is an effective technique for improving performance.

A concern with lagged gradient methods is the stability. The choice of d between 1 and 2 can be viewed as a compromise between stability and convergence speed. Choosing larger d may however yield no gain in convergence speed but a heavy price in instability, which may decrease the maximum attainable accuracy. For the CSD-like methods, from the experience of the present authors, choosing $d > 5$ can occasionally lead to serious accuracy problems when $\kappa = 10^4$. For SDC, however, it depends on the choice of both d_1 and d_2. Choosing $d_1 = 1$ and $d_2 > 8$ sometimes yields an oscillation or a divergence, while choosing $d_1 = 5$ and $d_2 = 15$ seems to be stable when $\kappa = 10^4$. Raydan and Svaiter [128] studied the relaxed SD step and observed its effectiveness. Yuan [155] suggested that a good gradient method should use at least one SD step in every few iterations. van den Doel and Ascher [143] argued that faster gradient methods should use occasionally larger steplengths. It appears that SDC satisfies all the arguments given above.

Another serious concern is the conditioning of Hankel matrices. As mentioned in Section 5.2, Hankel matrices are ill-conditioned even for small orders. Hoemmen [91] discussed a technique called equilibration which entails applying a transformation $\hat{A} = D^{-1/2} A D^{-1/2}$, where the diagonal elements of D equal the diagonal elements of A. This technique is indeed a special case of preconditioning and has been successfully applied to the communication-avoiding Krylov subspace methods (see, e.g., [91, 31]). We will discuss the impact of equilibration for Hankel matrices in the next section. On the other hand, there

exist faster solvers for Hankel systems with a complexity of $\mathcal{O}(s^2)$ compared with $\mathcal{O}(s^3)$ for the Cholesky factorization (see, e.g., [88]). Nevertheless, the limiting behavior ignores the importance of large constant terms. Since s must be chosen small enough to maintain the conditioning, a traditional $\mathcal{O}(s^3)$ solver seems to be preferred. Additionally, advanced methods are more complex to implement.

5.6 Numerical experiments

The goal of this section is to show the numerical behaviors of the methods discussed in this chapter. Among the experiments we have run, we would like to illustrate a few that are representative. Concerning parallel experiments, we have implemented the algorithms by using Alinea [104], an advanced linear algebra library developed for massively parallel computations, on a cluster of nodes comprising Intel Xeon CPU E5-2670 v3 2.30 GHz connected with FDR Infiniband network 56 Gbit/s. The MPI environment is supported by SGI MPT 2.12. Other experiments have been performed in MATLAB R2018b.

Some of the linear systems that we attempt to solve are randomly generated by MATLAB, while others are collected from the University of Florida Sparse Matrix Collection [53]. In all tests, the elements of initial vector x_0 are randomly selected from $(-1, 1)$. The right-hand side is fixed with $b = 0$ and thus $x_* = 0$. The starting vector is fixed with $x_0 = 0$. The iteration is stopped whenever $\|g_n\| < 10^{-6}\|g_0\|$.

In the first test, we give a general comparison of the aforementioned methods, shown in Figure 5.1. The first plot (top-left) shows that SD and s-SD converge much slower than other methods. As expected, monotone methods are not so effective, even for a moderately well-conditioned system. The second plot (top-right) gives an example for nonmonotone methods. Along with other results, we conclude that SDC is a robust and fast solver, while s-SDC is generally less efficient. For CSD and Cs-SD, we perform more tests with different parameters (e.g., the third and the fourth plots) but could not come to a general conclusion. What seems to be clear is that lagged gradient methods are highly sensitive to initial conditions, of which a fair comparison could not be outlined by several curves.

In Tables 5.2 to 5.5, we perform a set of parallel experiments for two structural problems. The first one is ill-conditioned with $N = 10848$ and $\kappa = 9.967 \times 10^9$. The matrix name is `msc10848` and the matrix ID is 361. The second one has a larger size $N = 141347$. The matrix name is `bmw7st_1` and the matrix ID is 1253. We can see in Tables 5.2 and 5.3 that in all cases actual convergence does not occur within 800 iterations. By fixing the number of iterations, we observe that the lagged strategy can significantly reduce the communication and computation costs. In Tables 5.4 and 5.5, most of the procedures are stopped before

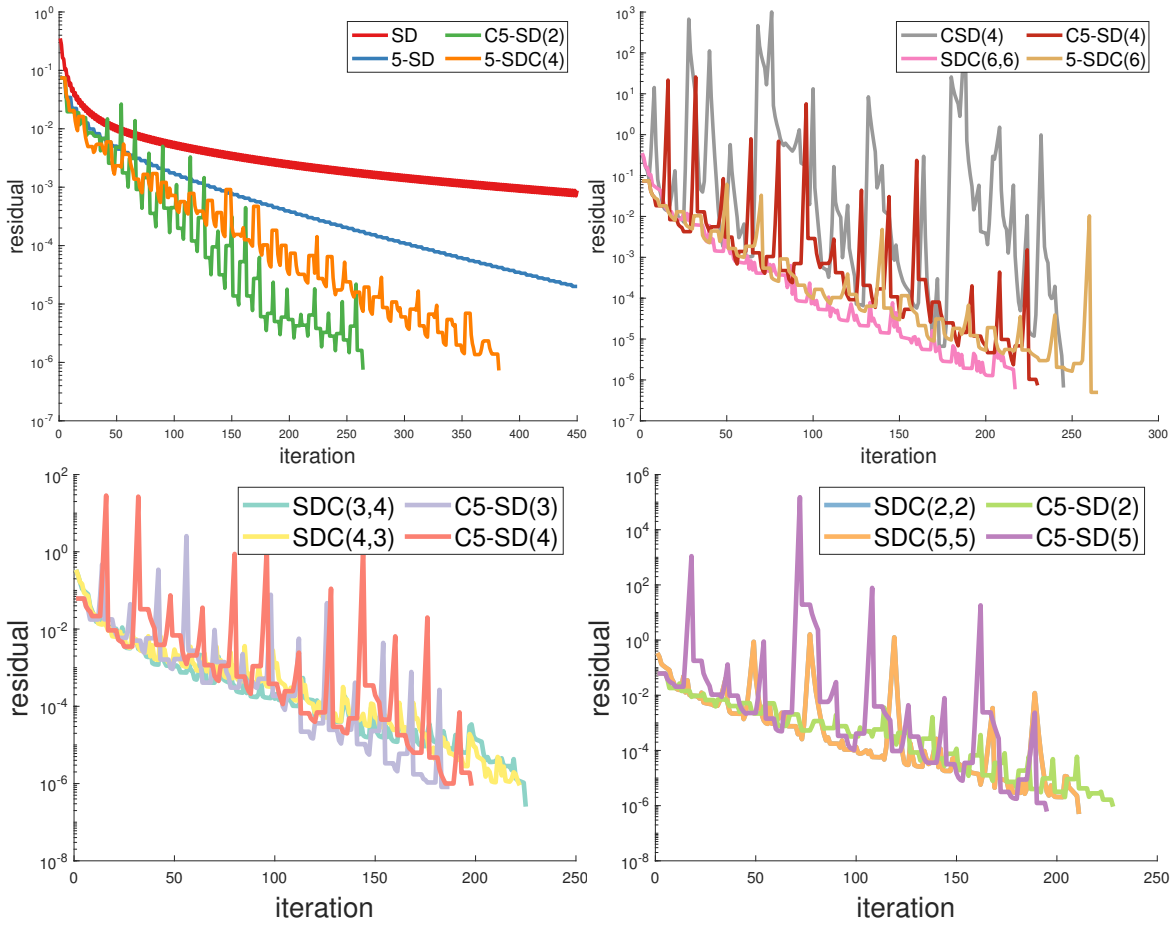

Figure 5.1 Experiments with random matrices: $N = 10^2$, $\kappa = 10^3$ (top), $N = 10^2$, $\kappa = 10^2$ (bottom).

Table 5.2 Average results among 10 tests with 16 and 32 processors. The matrix is ill-conditioned with $N = 10848$. The threshold of iteration is 800 and the stopping criterion is $\|g_n\| < 10^{-6}\|g_0\|$.

	16 processors			32 processors		
method	iter	time(s)	residual	iter	time(s)	residual
SD	800	0.616	5.6×10^{-5}	800	0.462	5.6×10^{-5}
BB	800	0.515	2.8×10^{-6}	800	0.364	4.6×10^{-5}
CSD(4)	800	0.468	3.2×10^{-4}	800	0.311	2.5×10^{-2}
SDC(4,4)	800	0.507	1.1×10^{-5}	800	0.350	2.9×10^{-6}
C5-SD(4)	800	0.495	6.1×10^{-6}	800	0.341	4.4×10^{-6}

reaching the specified iteration limit. Although BB is substantially faster than SD, cyclic formulations are generally superior to BB in terms of the convergence behavior. It appears

Table 5.3 Average results among 10 tests with 64 and 128 processors. The matrix is ill-conditioned with $N = 10848$. The threshold of iteration is 800 and the stopping criterion is $\|g_n\| < 10^{-6}\|g_0\|$.

	64 processors			128 processors		
method	iter	time(s)	residual	iter	time(s)	residual
SD	800	0.402	5.6×10^{-5}	800	0.408	5.6×10^{-5}
BB	800	0.304	1.1×10^{-3}	800	0.305	1.8×10^{-4}
CSD(4)	800	0.256	4.2×10^{-3}	800	0.252	5.4×10^{-6}
SDC(4,4)	800	0.301	2.2×10^{-6}	800	0.301	2.3×10^{-6}
C5-SD(4)	800	0.298	1.8×10^{-4}	800	0.300	8.1×10^{-5}

Table 5.4 Average results among 10 tests with 16 and 32 processors. The matrix is larger with $N = 141347$. The threshold of iteration is 10^4 and the stopping criterion is $\|g_n\| < 10^{-6}\|g_0\|$.

	16 processors			32 processors		
method	iter	time(s)	residual	iter	time(s)	residual
SD	$> 10^4$	\	\	$> 10^4$	\	\
BB	567	2.354	8.3×10^{-7}	1034	3.168	9.5×10^{-7}
CSD(4)	385	1.355	3.4×10^{-7}	365	0.850	1.1×10^{-6}
SDC(4,4)	468	1.927	8.7×10^{-7}	378	1.144	8.4×10^{-7}
C5-SD(4)	509	2.163	7.8×10^{-7}	405	1.153	9.9×10^{-7}

Table 5.5 Average results among 10 tests with 64 and 128 processors. The matrix is larger with $N = 141347$. The threshold of iteration is 10^4 and the stopping criterion is $\|g_n\| < 10^{-6}\|g_0\|$.

	64 processors			128 processors		
method	iter	time(s)	residual	iter	time(s)	residual
SD	$> 10^4$	\	\	$> 10^4$	\	\
BB	779	1.903	8.0×10^{-7}	788	1.723	7.1×10^{-7}
CSD(4)	590	1.017	9.4×10^{-7}	378	0.523	5.8×10^{-6}
SDC(4,4)	421	1.016	8.6×10^{-7}	301	0.652	9.9×10^{-7}
C5-SD(4)	605	1.392	9.0×10^{-7}	525	1.119	9.2×10^{-7}

that Cs-SD in Tables 5.4 and 5.5 requires more iterations than other competing methods. However, Cs-SD is still better than BB, both from the point of view of convergence speed

Table 5.6 Average results among 10 tests with 64 and 128 processors. The matrix is very large with $N = 1564794$. The threshold of iteration is 10^4 and the stopping criterion is $\|g_n\| < 10^{-6}\|g_0\|$.

	64 processors			128 processors		
method	iter	time(s)	residual	iter	time(s)	residual
BB	720	23.649	7.9×10^{-7}	625	16.246	8.9×10^{-7}
CSD(4)	818	19.672	8.5×10^{-7}	709	13.594	8.4×10^{-7}
SDC(4,4)	453	14.503	9.4×10^{-7}	445	11.796	9.1×10^{-7}
C5-SD(4)	768	22.925	9.2×10^{-7}	545	14.002	9.7×10^{-7}

Table 5.7 Average results among 10 tests with 256 and 512 processors. The matrix is very large with $N = 1564794$. The threshold of iteration is 10^4 and the stopping criterion is $\|g_n\| < 10^{-6}\|g_0\|$.

	256 processors			512 processors		
method	iter	time(s)	residual	iter	time(s)	residual
BB	601	14.179	9.4×10^{-7}	762	18.017	8.0×10^{-7}
CSD(4)	569	9.591	8.9×10^{-7}	598	9.844	8.0×10^{-7}
SDC(4,4)	429	10.505	8.9×10^{-7}	429	10.269	9.8×10^{-7}
C5-SD(4)	671	14.998	8.0×10^{-7}	700	15.522	9.9×10^{-7}

and robustness. In Tables 5.2 and 5.3, we find that Cs-SD and SDC are indistinguishable in terms of computing time. Additionally, SDC can ensure a good balance between efficiency and stability, while CSD requires less arithmetic operations but may decrease accuracy. Each of the two has its own dynamics and we could not provide a general recommendation. It is important to note that SD is much more robust than the nonmonotone methods in terms of residual, while CSD seems to be the most unstable one from Tables 5.4 and 5.5, in which the gap of iteration count among different situations is fairly large.

The next test proceeds along the same lines but considers a large-scale problem. We draw another matrix from the University of Florida Sparse Matrix Collection with $N = 1564794$, which is obtained from a 3D mechanical problem and discretized with the finite element method. The matrix name is `Flan_1565` and the matrix ID is 2544. We perform several experiments on a cluster using 64, 128, 256 and 512 cores. Tables 5.6 and 5.7 illustrate the results. Note that we will not show the results of SD since it requires always more than 10^4 iterations. From these tables, we find that SDC is still efficient in the large-scale case, while BB generally takes a long time to finish the job. It is noteworthy that there is almost

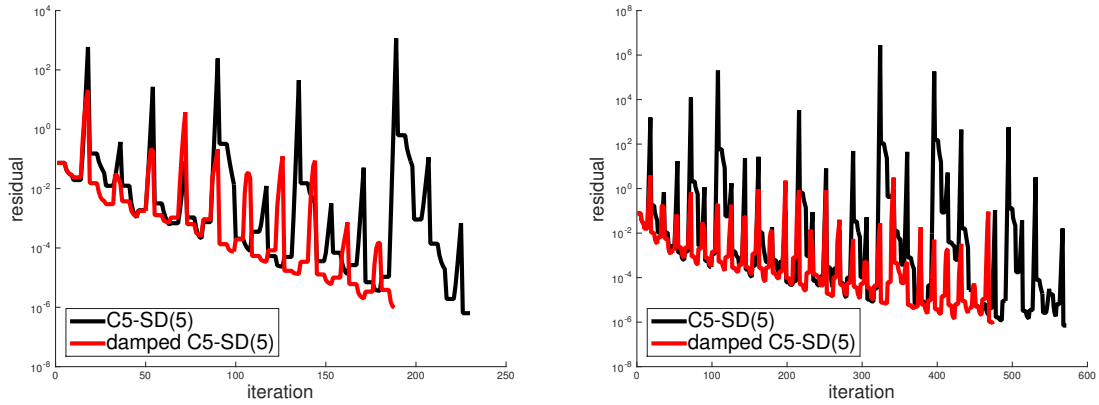

Figure 5.2 A comparison of Cs-SD and damped Cs-SD with $s = 5$ and $d = 5$ for random problems: $N = 10^2$, $\kappa = 10^3$ (left), $N = 10^2$, $\kappa = 10^4$ (right).

no gain in using 512 cores compared with the case of 256 cores. This may be due to the communication costs and the limit of problem size.

One may wonder if Cs-SD could be further improved. One possible way is the damped formulation as mentioned in Section 5.3. It is often observed that the iteration count of damped Cs-SD is smaller than the original method. Two examples are shown in Figure 5.2, in which we use two 100×100 random matrices with $\kappa = 10^3$ and $\kappa = 10^4$, respectively. Note that the opposite results can also be obtained, which depends on the initial condition and the implementation. In contrast, if we only select the smallest possible step $\omega_n^{(2s-2)}/\omega_n^{(2s-1)}$, then a worse convergence result will generally be obtained.

In the next case, we discuss the use of equilibration for managing the conditioning of Hankel matrices. Numerical experiments in Figure 5.3 indicate a marked improvement. The matrix size is 7102 with a condition number of 1.6×10^4. The left plot shows how the condition number changes over iteration, in which a two-step invariance property can be clearly confirmed. The right plot shows the changes of conditioning over s. For each s, we choose the 49th and the 50th Hankel matrices and calculate their condition numbers. The larger one is illustrated in the right plot. We observe that the equilibration technique improves the quality of Hankel matrices.

5.7 Conclusion

Lagged gradient method is a useful tool for large-scale linear systems. For well-conditioned problems, for example, SDC can often be competitive with the conjugate gradient method. Lagged methods are most appealing when initial perturbations occur in the original matrices.

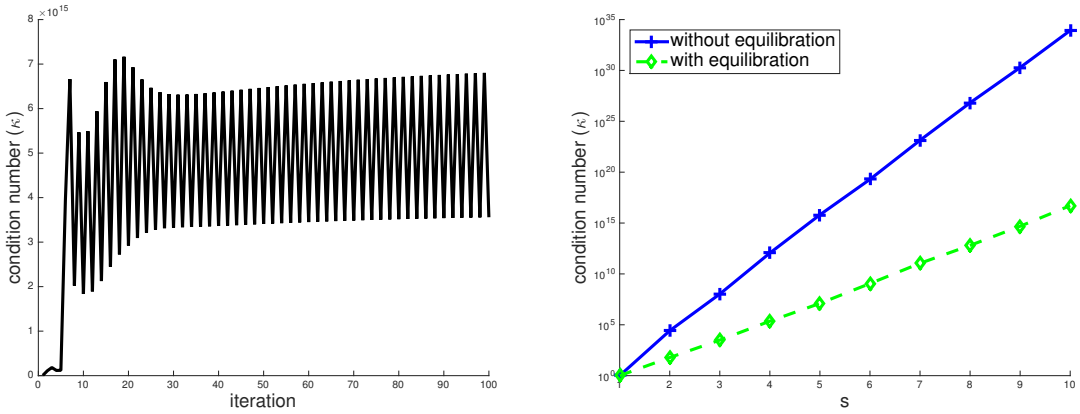

Figure 5.3 Experiments for the conditioning of Hankel systems. The matrix is generated from a structural problem with $N = 7102$ and $\kappa = 1.6 \times 10^4$. The left plot shows the condition number when $s = 5$. The right plot shows the impact of equilibration.

Concerning parallel performance, SDC is still viewed as the most robust method. The behavior of our new cyclic formulation Cs-SD is indistinguishable from that of CSD. On the other hand, SD, s-SD, BB and s-SDC are generally less or far less efficient than these methods. The damped formulation of Cs-SD and the equilibration technique could be used with success in most cases. We have also shown parallel algorithms of all these methods, as well as some simplified schemes for the purpose of comparison. A comparison of parallel lagged gradient methods and the communication-avoiding Krylov methods seems to be an interesting point and needs to be tackled.

Chapter 6

Conclusion

This thesis focuses on the practical use of gradient methods, especially the spectral properties of optimal methods and the applications of lagged methods. Chapter 2 has presented an overview of gradient methods, as well as the relevant theories. Chapter 3 has given the spectral properties of minimal gradient and proposed some new alignment methods. Chapter 4 has discussed the application of gradient iterations to the splitting methods. Chapter 5 has listed the parallel algorithms and proposed new formulations for the purposes of communication avoidance. Experiments have been conducted at the end of each chapter.

Monotone gradient methods face challenges in solving ill-conditioned problems even with preconditioners, in which case preconditioned conjugate gradient [112] is often the method of choice. Efficient gradient solvers like classical gradient methods with retards and alignment methods (see Section 3.2) are useful in the following cases:

- alignment methods are competitive with conjugate gradient (see Section 3.4);
- GMR and alignment methods are less sensitive to perturbation, i.e., matrix A in linear system (2.1) has an initial nonsymmetric error, or quadratic function (2.2) has a small nonquadratic term (see [47, 65, 143]);
- GMR is competitive with conjugate gradient when low precision is required, such as inexact-Newton methods [74], image restoration [58] and splitting methods (see Section 4.3.3).

Apart from these cases, asymptotic properties of optimal gradient methods like SD and MG can be exploited to estimate the minimum and maximum eigenvalues (see Section 4.3.2); cyclic gradient methods like CSD and Cs-SD are quite promising in a parallel environment (see Sections 5.1 and 5.3). For the large-scale problems, we can see from the experiments that the modern gradient methods are quite promising.

There are also questions left to answer about lagged gradient methods. Raydan [126] proved the convergence of BB and Dai and Liao [48] obtained the *R*-linear convergence result. However, these advances could not fully explain its surprising behavior. Much prior work has focused on the low-dimensional linear system [17, 47, 46], but there is still much to be learned about the high-dimensional case. Some papers have revealed that fast gradient methods exhibit chaotic behavior [123, 122, 143]. The theoretical analysis of lagged methods seems to be more challenging than ever and we leave this to future work.

Gradient methods are often placed in an awkward position for the solution of linear systems: they are less intuitive than stationary iterative methods [152, 148] and less efficient than Krylov subspace methods [131, 146] in ill-conditioned cases. It is acceptable to classify several gradient variants under one-dimensional projection methods [131]. However, lagged gradient methods do not belong to this category and face a problem of taxonomy. On the other hand, these methods can be successfully applied to system (2.1), but are also commonly applied to some other problems, such as general unconstrained optimization [127] and constrained optimization [25]. In view of the rapidly developing state of the art of gradient methods, it is appealing to adapt these techniques to optimization problems. We note this as another topic for future investigation.

As discussed in Chapter 5, iterative methods with retards can be beneficial in a parallel environment. There are also unresolved questions in this case. First, we have not exploited the structure of sparse matrices, which mostly contains zeros entries, so it is important to develop efficient algorithms according to the specific situations. Then, there are few available preconditioners that can reduce communication costs, see, e.g., [85]. How to develop communication-avoiding preconditioners for ill-conditioned problems remains an open question. It is also appealing to investigate parallel algorithms for solving linear systems with multiple right-hand sides. We leave these for future work.

Appendix A

Résumé

A.1 Méthodes du gradient

A.1.1 Introduction

On considère la résolution du système linéaire

$$Ax = b,$$

où $A \in \mathbb{R}^{N\times N}$ est une matrice symétrique définie positive et $b \in \mathbb{R}^N$ le second membre. Soit $\{\lambda_1, \ldots, \lambda_N\}$ les valeurs propres de A et $\{v_1, \ldots, v_N\}$ les vecteurs propres correspondants. On suppose que $\lambda_1 \leq \cdots \leq \lambda_N$. On note $\kappa = \lambda_N/\lambda_1$ le conditionnement spectral de A. La fonction $f : \mathbb{R}^N \to \mathbb{R}$ telle que

$$f(x) = \frac{1}{2}x^\intercal Ax - b^\intercal x$$

sera minimale pour la solution $x_* = A^{-1}b$. La méthode du gradient est un schéma itératif non-stationnaire défini par

$$x_{n+1} = x_n - \alpha_n g_n,$$

avec $g_n = \nabla f(x_n) = Ax_n - b$. Il existe donc des nombres réels $\zeta_{i,n}$ tels que

$$g_n = \sum_{i=1}^{N} \zeta_{i,n} v_i,$$

pour tout n. Certaines méthodes du gradient sont résumées dans la liste suivante:

- Méthode de Cauchy ou méthode de la plus rapide descente (SD) [33]

$$\alpha_n^{\mathrm{SD}} = \frac{g_n^{\mathsf{T}} g_n}{g_n^{\mathsf{T}} A g_n}.$$

- Méthode asymptotiquement optimale (AO) [49]

$$\alpha_n^{\mathrm{AO}} = \frac{\|g_n\|}{\|A g_n\|}.$$

- Méthode de Barzilai-Borwein (BB) [17]

$$\alpha_n^{\mathrm{BB}} = \frac{g_{n-1}^{\mathsf{T}} g_{n-1}}{g_{n-1}^{\mathsf{T}} A g_{n-1}}.$$

- Méthode du gradient à retards (GMR) [74]

$$\alpha_n^{\mathrm{GMR}} = \frac{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)+1} g_{\tau(n)}}, \tag{A.1}$$

  avec

$$\tau(n) \in \{\bar{n}, \bar{n}+1, \ldots, n-1, n\}, \quad \rho(n) \in \{q_1, \ldots, q_m\}, \quad q_j \geq 0,$$

  où m est un entier positive et $\bar{n} = \max\{0,\ n-m\}$.

SD et AO offrent en général de bonnes propriétés en terme de vitesse de convergence dans les premières itérations. Malheureusement, elles travaillent de façon stagnante dès qu'on s'approche du point stationnaire. BB fut la première méthode du gradient à retards qui permet d'obtenir une convergence plus rapide que les méthodes monotones. GMR établit un cadre pour la convergence des méthodes du gradient.

Deux pas auxiliaires α_n^{A} et α_n^{Y} furent respectivement proposés dans [56] et [154] tels que

$$\alpha_n^{\mathrm{A}} = \left(\frac{1}{\alpha_{n-1}^{\mathrm{SD}}} + \frac{1}{\alpha_n^{\mathrm{SD}}} \right)^{-1},$$

et

$$\alpha_n^{\mathrm{Y}} = 2 \left(\sqrt{\left(\frac{1}{\alpha_{n-1}^{\mathrm{SD}}} - \frac{1}{\alpha_n^{\mathrm{SD}}} \right)^2 + \frac{4\,\|g_n\|^2}{\left(\alpha_{n-1}^{\mathrm{SD}}\right)^2 \|g_{n-1}\|^2}} + \frac{1}{\alpha_{n-1}^{\mathrm{SD}}} + \frac{1}{\alpha_n^{\mathrm{SD}}} \right)^{-1},$$

conduisant à certaines méthodes performantes dans la liste suivante:

- Méthode de Dai-Yuan (DY) [51]

$$\alpha_n^{\mathrm{DY}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod 4 = 0 \text{ ou } 1, \\ \alpha_n^{\mathrm{Y}}, & \text{autres cas.} \end{cases}$$

- Méthode de Cauchy avec alignement (SDA) [56]

$$\alpha_n^{\mathrm{SDA}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod (d_1+d_2) < d_1, \\ \alpha_n^{\mathrm{A}}, & n \bmod (d_1+d_2) = d_1, \quad d_1, d_2 \geq 1. \\ \alpha_{n-1}^{\mathrm{SDA}}, & \text{autres cas,} \end{cases}$$

- Méthode de Cauchy alignée par le pas constant (SDC) [57]

$$\alpha_n^{\mathrm{SDC}} = \begin{cases} \alpha_n^{\mathrm{SD}}, & n \bmod (d_1+d_2) < d_1, \\ \alpha_n^{\mathrm{Y}}, & n \bmod (d_1+d_2) = d_1, \quad d_1, d_2 \geq 1. \\ \alpha_{n-1}^{\mathrm{SDC}}, & \text{autres cas,} \end{cases}$$

A.1.2 Conditions de convergence

On suppose que $A = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$. Cette hypothèse semble assez stricte dans la pratique. Cependant, pour l'analyse théorique, on pourrait simplement ajouter une transformation orthogonale qui transforme A en une matrice diagonale.

Théorème A.1. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Alors la méthode du gradient avec le pas* (A.1) *converge vers x_* pour tout x_0.*

Définition (Propriété A)**.** Soit $g_{i,n}$ la $i^{\text{ème}}$ composante de g_n et

$$G(n,\mu) = \sum_{i=1}^{\mu} g_{i,n}^2.$$

Si $\exists m_0 \in \mathbb{N}$, $\exists c_1, c_2 > 0$, tel que $\forall \mu \in \{1, \ldots, N-1\}$, $\forall \varepsilon > 0$, $\forall j \in \{0, \ldots, \min\{n, m_0\}\}$,

1. $\lambda_1 \leq \alpha_n^{-1} \leq c_1$;
2. si $G(n-j,\mu) \leq \varepsilon$ et $g_{\mu+1,n-j}^2 \geq c_2\varepsilon$, alors $\alpha_n^{-1} \geq \frac{2}{3}\lambda_{\mu+1}$,

alors le pas α_n a la Propriété A.

Théorème A.2. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si α_n a la Propriété A, alors la suite $\{\|g_n\|\}$ générée par une méthode du gradient converge vers* 0.

A.1.3 Propriétés asymptotiques

On suppose que $0 < \lambda_1 < \cdots < \lambda_N$ et $\zeta_{1,0}$, $\zeta_{N,0} \neq 0$.

Lemme A.3. *Soit p_0 une mesure de probabilité attachée à $\{\lambda_1, \ldots, \lambda_N\}$ où $p_{i,0} = p_0(\lambda_i)$ et $0 < \lambda_1 < \cdots < \lambda_N$. On considère une application telle que*

$$p_{i,n+1} = \frac{\left(\sum_{j=1}^{N} \lambda_j p_{j,n} - \lambda_i\right)^2}{\sum_{l=1}^{N} \left(\sum_{j=1}^{N} \lambda_j p_{j,n} - \lambda_l\right)^2 p_{l,n}} p_{i,n}.$$

Alors

$$\lim_{n\to\infty} p_{i,2n} = \begin{cases} p_*, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ 1 - p_*, & i = N, \end{cases}$$

et

$$\lim_{n\to\infty} p_{i,2n+1} = \begin{cases} 1 - p_*, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ p_*, & i = N, \end{cases}$$

pour certain $p_* \in (0, 1)$.

Théorème A.4. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par AO, alors*

$$\lim_{n\to\infty} \alpha_n^{\mathrm{AO}} = \frac{2}{\lambda_1 + \lambda_N}.$$

A.2 Méthodes du gradient avec alignement

A.2.1 Analyse spectrale de gradient minimal

On considère la méthode de gradient minimal (MG) suivante:

$$\alpha_n^{\mathrm{MG}} = \frac{g_n^{\mathsf{T}} A g_n}{g_n^{\mathsf{T}} A^2 g_n}.$$

On suppose que $0 < \lambda_1 < \cdots < \lambda_N$ et $\zeta_{1,0}, \zeta_{N,0} \neq 0$. Il est possible de prouver quelques nouvelles propriétés asymptotiques qui seront utiles dans la suite de cette section.

Théorème A.5. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par MG, alors*

$$\lim_{n\to\infty} \frac{\lambda_i \zeta_{i,2n}^2}{\sum_{j=1}^N \lambda_j \zeta_{j,2n}^2} = \begin{cases} \frac{1}{1+c^2}, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ \frac{c^2}{1+c^2}, & i = N, \end{cases}$$

et

$$\lim_{n\to\infty} \frac{\lambda_i \zeta_{i,2n+1}^2}{\sum_{j=1}^N \lambda_j \zeta_{j,2n+1}^2} = \begin{cases} \frac{c^2}{1+c^2}, & i = 1, \\ 0, & i \in \{2, \ldots, N-1\}, \\ \frac{1}{1+c^2}, & i = N, \end{cases}$$

pour certain c. De plus, $\zeta_{1,2n}$, $\zeta_{N,2n}$, $\zeta_{1,2n+1}$, $\zeta_{N,2n+1}$ ont des signes fixes pour grand n.

Théorème A.6. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par MG, alors*

$$\lim_{n\to\infty} \frac{f(x_{2n+1}) - f(x_*)}{f(x_{2n}) - f(x_*)} = \frac{c^2(1+c^2\kappa^2)(\kappa-1)^2}{(c^2+\kappa^2)(1+c^2\kappa)^2},$$

et

$$\lim_{n\to\infty} \frac{f(x_{2n+2}) - f(x_*)}{f(x_{2n+1}) - f(x_*)} = \frac{c^2(c^2+\kappa^2)(\kappa-1)^2}{(1+c^2\kappa^2)(c^2+\kappa)^2},$$

pour certain c.

Théorème A.7. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par MG, alors*

$$\lim_{n\to\infty} \alpha_{2n}^{\mathrm{MG}} = \frac{1+c^2}{\lambda_1(1+c^2\kappa)},$$

et

$$\lim_{n\to\infty} \alpha_{2n+1}^{\mathrm{MG}} = \frac{1+c^2}{\lambda_1(c^2+\kappa)},$$

pour certain c.

Théorème A.8. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par MG, alors*

$$\lim_{n\to\infty} \frac{\|g_{n+1}\|^2}{\|g_n\|^2} = \frac{c^2(\kappa-1)^2}{(c^2+\kappa)(1+c^2\kappa)},$$

pour certain c.

Corollaire A.9. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par MG, alors*

$$\lim_{n\to\infty} \frac{f(x_{2n+2}) - f(x_*)}{f(x_{2n}) - f(x_*)} = \lim_{n\to\infty} \frac{\|g_{n+1}\|^4}{\|g_n\|^4}.$$

Théorème A.10. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par MG, alors*

$$\lim_{n\to\infty} \frac{g_{2n+1}^{\mathsf{T}} A g_{2n+1}}{g_{2n}^{\mathsf{T}} A g_{2n}} = \frac{c^2(\kappa-1)^2}{(1+c^2\kappa)^2},$$

et

$$\lim_{n\to\infty} \frac{g_{2n+2}^{\mathsf{T}} A g_{2n+2}}{g_{2n+1}^{\mathsf{T}} A g_{2n+1}} = \frac{c^2(\kappa-1)^2}{(c^2+\kappa)^2},$$

pour certain c.

A.2.2 Nouvelles méthodes avec alignement

Autant que nous le sachions, toutes les méthodes du gradient avec alignement existantes sont basées sur le pas de Cauchy. Nous montrons que cette condition n'est pas nécessaire. Trois

méthodes qui possèdent potentiellement la même caractéristique sans pas de Cauchy sont dérivées.

Théorème A.11. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par un pas constant $\hat{\alpha}$ tel que*

$$\hat{\alpha} \leq \frac{2}{\lambda_1 + \lambda_N},$$

alors la méthode du gradient converge vers x_* pour tout x_0. De plus, si l'égalité est appliquée,

$$\lim_{n\to\infty} \frac{\zeta_{i,n}}{\zeta_{1,n}} = \begin{cases} 0, & i = 2, 3, \ldots, N-1, \\ \frac{\zeta_{N,0}}{\zeta_{1,0}}(-1)^n, & i = N; \end{cases}$$

sinon,

$$\lim_{n\to\infty} \frac{\zeta_{i,n}}{\zeta_{1,n}} = 0, \quad i = 2, 3, \ldots, N.$$

Nous constatons que le théorème précédent a deux effets: piloter la propriété d'alignement lorsque l'ordre partiel strict est maintenu et forcer la recherche dans un espace bidimensionnel lorsque l'égalité est vérifiée. Cela signifie que s'il existe un pas permettant d'atteindre l'égalité de manière asymptotique, alors il a une tendance similaire à celle de SD, c'est-à-dire l'alternance entre deux directions orthogonales [1]. Soit $\tilde{\alpha}_n = \theta \alpha_n^{\mathrm{AO}}$ où $0 < \theta < 1$. Il s'ensuit que

$$\lim_{n\to\infty} \tilde{\alpha}_n < \frac{2}{\lambda_1 + \lambda_N}.$$

Par conséquent, nous pouvons écrire une nouvelle méthode du gradient appelée AO avec alignement (AOA) de la forme suivante:

$$\alpha_n^{\mathrm{AOA}} = \begin{cases} \alpha_n^{\mathrm{AO}}, & n \bmod (d_1 + d_2) < d_1, \\ \tilde{\alpha}_n, & n \bmod (d_1 + d_2) = d_1, \\ \alpha_{n-1}^{\mathrm{AOA}}, & \text{autres cas}, \end{cases}$$

avec $d_1, d_2 \geq 1$.

D'un autre côté, on peut noter

$$\alpha_n^{\mathrm{A2}} = \left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}}} + \frac{1}{\alpha_n^{\mathrm{MG}}} \right)^{-1},$$

et

$$\alpha_n^{\mathrm{Y2}} = 2\left(\sqrt{\left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}}} - \frac{1}{\alpha_n^{\mathrm{MG}}}\right)^2 + \frac{4g_n^{\mathsf{T}} A g_n}{\left(\alpha_{n-1}^{\mathrm{MG}}\right)^2 g_{n-1}^{\mathsf{T}} A g_{n-1}}} + \frac{1}{\alpha_{n-1}^{\mathrm{MG}}} + \frac{1}{\alpha_n^{\mathrm{MG}}}\right)^{-1}.$$

On en montre les propriétés spectrales, c'est-à-dire qu'on obtient les théorèmes suivants:

Théorème A.12. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par MG, alors*

$$\lim_{n\to\infty} \alpha_n^{\mathrm{A2}} = \frac{1}{\lambda_1 + \lambda_N}.$$

Théorème A.13. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Si la suite de solutions $\{x_n\}$ est générée par MG, alors*

$$\lim_{n\to\infty} \alpha_n^{\mathrm{Y2}} = \frac{1}{\lambda_N},$$

et

$$\lim_{n\to\infty} \left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}} \alpha_n^{\mathrm{MG}}} - \frac{g_n^{\mathsf{T}} A g_n}{\left(\alpha_{n-1}^{\mathrm{MG}}\right)^2 g_{n-1}^{\mathsf{T}} A g_{n-1}}\right) = \lambda_1 \lambda_N.$$

Nous pouvons définir MG avec alignement (MGA) et MG alignée par le pas constant (MGC) comme les équations suivantes:

$$\alpha_n^{\mathrm{MGA}} = \begin{cases} \alpha_n^{\mathrm{MG}}, & n \bmod (d_1 + d_2) < d_1, \\ \alpha_n^{\mathrm{A2}}, & n \bmod (d_1 + d_2) = d_1, \\ \alpha_{n-1}^{\mathrm{MGA}}, & \text{autres cas}, \end{cases}$$

$$\alpha_n^{\mathrm{MGC}} = \begin{cases} \alpha_n^{\mathrm{MG}}, & n \bmod (d_1 + d_2) < d_1, \\ \alpha_n^{\mathrm{Y2}}, & n \bmod (d_1 + d_2) = d_1, \\ \alpha_{n-1}^{\mathrm{MGC}}, & \text{autres cas}. \end{cases}$$

A.2.3 Analyse des convergences

Une généralisation de GMR (DGMR) peut s'écrire sous la forme suivante:

$$\alpha_n^{\text{DGMR}} = \left(\frac{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)} g_{\tau(n)}}{g_{\tau(n)}^{\mathsf{T}} A^{\rho(n)+1} g_{\tau(n)}} \right)^{\frac{1}{\upsilon}}, \tag{A.2}$$

avec

$$\tau(n) \in \{\bar{n}, \bar{n}+1, \ldots, n-1, n\}, \quad \rho(n) \in \{q_1, \ldots, q_m\}, \quad q_j \geq 0, \quad \upsilon > 0,$$

où m est un entier positive et $\bar{n} = \max\{0,\ n-m\}$. Le résultat suivant a été mentionné dans [45] sans preuve.

Théorème A.14. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Alors la méthode du gradient avec le pas* (A.2) *converge vers x_* pour tout x_0.*

On établit la convergence de AOA suivante de manière immédiate:

Théorème A.15. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Alors la suite $\{x_n\}$ générée par AOA converge vers x_* pour tout x_0.*

De manière similaire, les théorèmes suivants peuvent être formulés:

Théorème A.16. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Alors la suite $\{x_n\}$ générée par MGA converge vers x_* pour tout x_0.*

Théorème A.17. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Alors la suite $\{x_n\}$ générée par MGC converge vers x_* pour tout x_0.*

A.2.4 Résultats expérimentaux

Les algorithmes ont été codés en MATLAB R2018b. La figure A.1 représente l'influence des paramètres d_1 et d_2 sur le nombre d'itérations d'une méthode du gradient avec alignement. La figure A.2 représente le résultat des méthodes du gradient avec alignement: SDA, SDC, AOA, MGA et MGC.

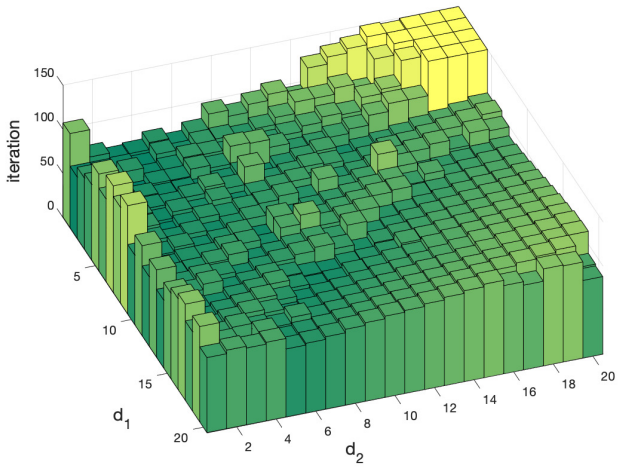

Figure A.1 L'influence des paramètres sur une méthode du gradient avec alignement (ici: MGC).

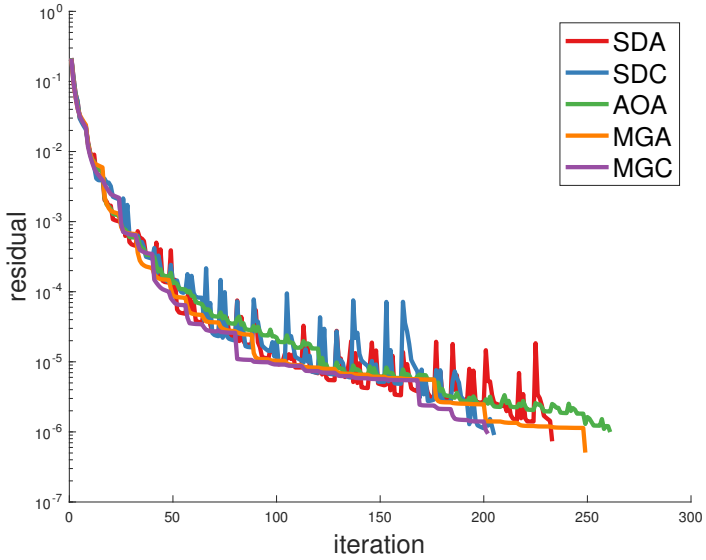

Figure A.2 Résultat des méthodes du gradient avec alignement.

A.3 Estimation de paramètre pour les méthodes de splitting

A.3.1 Splitting hermitien et anti-hermitien

Considérons le système

$$Ax = b,$$

où A est une matrice non-hermitienne. Nous désignons par H et S des parties hermitienne et anti-hermitienne de A respectivement. Il s'ensuit que

$$H = \frac{A + A^{\mathsf{H}}}{2}, \quad S = \frac{A - A^{\mathsf{H}}}{2}.$$

La méthode de splitting hermitien et anti-hermitien (HSS) [11] repose sur l'équation suivante qui consiste en l'alternance des deux systèmes linéaires:

$$\begin{cases} (\gamma I + H)x_{n+\frac{1}{2}} = (\gamma I - S)x_n + b, \\ (\gamma I + S)x_{n+1} = (\gamma I - H)x_{n+\frac{1}{2}} + b, \end{cases}$$

avec $\gamma > 0$. Il pourrait être considéré comme un processus itératif stationnaire

$$x_{n+1} = Tx_n + p.$$

Nous pouvons écrire

$$\mathscr{M}_1 = \gamma I + H, \quad \mathscr{N}_1 = \gamma I - S, \quad \mathscr{M}_2 = \gamma I + S, \quad \mathscr{N}_2 = \gamma I - H.$$

Nous obtenons donc les expressions de T et p

$$T = \mathscr{M}_2^{-1}\mathscr{N}_2\mathscr{M}_1^{-1}\mathscr{N}_1, \quad p = \mathscr{M}_2^{-1}(I + \mathscr{N}_2\mathscr{M}_1^{-1})b.$$

Soit $\sigma(\cdot)$ le spectre d'une matrice et $\rho(\cdot)$ le rayon spectral. Nous obtenons

$$\rho(T) \leq \left\|\mathscr{N}_2\mathscr{M}_1^{-1}\right\| = \max_{\lambda \in \sigma(H)} \frac{|\lambda - \gamma|}{|\lambda + \gamma|}.$$

Le paramètre optimal peut s'écrire de la forme

$$\gamma_* = \sqrt{\lambda_1(H)\lambda_N(H)},$$

conduisant à la borne supérieure

$$\rho(T) \leq \frac{\sqrt{\kappa(H)} - 1}{\sqrt{\kappa(H)} + 1}.$$

A.3.2 Analyse spectrale

On considère d'abord la méthode de Cauchy pour le système hermitien

$$\alpha_n^{\mathrm{SD}} = \frac{g_n^{\mathsf{H}} g_n}{g_n^{\mathsf{H}} H g_n}.$$

Lemme A.18. *Les propriétés spectrales de SD pour le système $Ax = b$ où A est une matrice hermitienne définie positive sont les mêmes que pour le cas réel.*

Nous écrivons $\alpha_n^{\mathrm{RA}} = \left(\alpha_n^{\mathrm{A}}\right)^{-1}$ et

$$\Gamma_n = \frac{1}{\alpha_{n-1}^{\mathrm{SD}} \alpha_n^{\mathrm{SD}}} - \frac{\|g_n\|^2}{\left(\alpha_{n-1}^{\mathrm{SD}}\right)^2 \|g_{n-1}\|^2}.$$

Alors

$$\alpha_n^{\mathrm{Y}} = \frac{2}{\alpha_n^{\mathrm{RA}} + \sqrt{\left(\alpha_n^{\mathrm{RA}}\right)^2 - 4\Gamma_n}}.$$

Nous proposerons un nouveau pas

$$\alpha_n^{\mathrm{Z}} = \frac{2}{\alpha_n^{\mathrm{RA}} - \sqrt{\left(\alpha_n^{\mathrm{RA}}\right)^2 - 4\Gamma_n}}.$$

De la même manière, on note $\alpha_n^{\mathrm{RY}} = \left(\alpha_n^{\mathrm{Y}}\right)^{-1}$ et $\alpha_n^{\mathrm{RZ}} = \left(\alpha_n^{\mathrm{Z}}\right)^{-1}$. Cela implique

$$\alpha_n^{\mathrm{RY}} + \alpha_n^{\mathrm{RZ}} = \alpha_n^{\mathrm{RA}}, \quad \alpha_n^{\mathrm{RY}} \alpha_n^{\mathrm{RZ}} = \Gamma_n.$$

Lemme A.19. *Les limites suivantes s'appliquent à la méthode de SD pour le système hermitien définie positive*

$$\lim_{n\to\infty} \Gamma_n = \lambda_1(H)\lambda_N(H).$$

$$\lim_{n\to\infty} \alpha_n^{\mathrm{Y}} = \frac{1}{\lambda_N(H)}.$$

$$\lim_{n\to\infty} \alpha_n^{\mathrm{Z}} = \frac{1}{\lambda_1(H)}.$$

Il convient de noter que α_n^{Y} et α_n^{Z} pourrait s'exprimer comme les racines d'une fonction quadratique

$$Q_n(\alpha) = \Gamma_n \alpha^2 - \alpha_n^{\mathrm{RA}} \alpha + 1,$$

avec

$$Q_n(0) = 1, \quad Q_n(\alpha_n^{\mathrm{A}}) = \Gamma_n \left(\alpha_n^{\mathrm{A}}\right)^2,$$

$$Q_n(\alpha_{n-1}^{\mathrm{SD}}) = -\frac{\|g_n\|^2}{\|g_{n-1}\|^2}, \quad Q_n(\alpha_n^{\mathrm{SD}}) = -\frac{\left(\alpha_n^{\mathrm{SD}}\right)^2 \|g_n\|^2}{\left(\alpha_{n-1}^{\mathrm{SD}}\right)^2 \|g_{n-1}\|^2},$$

d'où l'on peut constater que $\Gamma_n > 0$ et

$$\alpha_n^{\mathrm{A}} < \alpha_n^{\mathrm{Y}} < \min\{\alpha_{n-1}^{\mathrm{SD}}, \alpha_n^{\mathrm{SD}}\}.$$

A.3.3 Estimation de paramètre

Le paramètre optimal dans HSS pourrait être approximé par SD. Les théorèmes suivants précisent cette observation:

Théorème A.20. *Si la méthode de SD est utilisée pour résoudre le système hermitien avec la matrice H, alors*

$$\lim_{n\to\infty} \sqrt{\Gamma_n} = \gamma_*.$$

Théorème A.21. *Si la méthode de SD est utilisée pour résoudre le système hermitien avec la matrice $\mathscr{M}_1$, alors*

$$\lim_{n\to\infty} \sqrt{\Gamma_n - \gamma\alpha_n^{\mathrm{RA}} + \gamma} = \gamma_*.$$

De la même manière, on considère la méthode de gradient minimal

$$\alpha_n^{\mathrm{MG}} = \frac{g_n^{\mathsf{H}} H g_n}{g_n^{\mathsf{H}} H^2 g_n},$$

On peut écrire les paramètres sous la forme

$$\alpha_n^{\mathrm{A2}} = \left(\frac{1}{\alpha_{n-1}^{\mathrm{MG}}} + \frac{1}{\alpha_n^{\mathrm{MG}}}\right)^{-1}, \quad \tilde{\Gamma}_n = \frac{1}{\alpha_{n-1}^{\mathrm{MG}}\alpha_n^{\mathrm{MG}}} - \frac{g_n^{\mathsf{H}} H g_n}{\left(\alpha_{n-1}^{\mathrm{MG}}\right)^2 g_{n-1}^{\mathsf{H}} H g_{n-1}}.$$

Soit $\alpha_n^{\mathrm{RA2}} = \left(\alpha_n^{\mathrm{A2}}\right)^{-1}$. Les théorèmes suivants donnent les résultats basés sur MG:

Théorème A.22. *Si la méthode de MG est utilisée pour résoudre le système hermitien avec la matrice H, alors*

$$\lim_{n\to\infty} \sqrt{\tilde{\Gamma}_n} = \gamma_*.$$

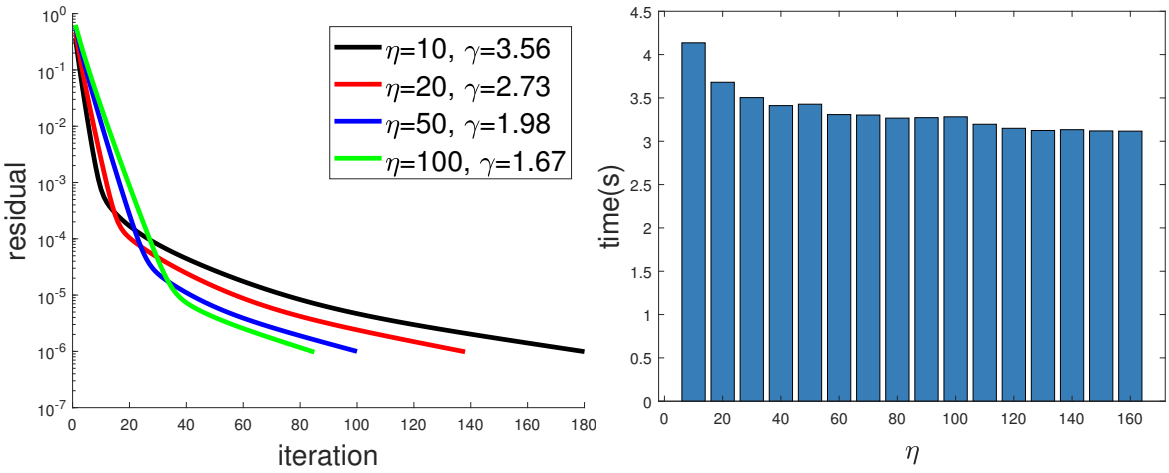

Figure A.3 L'influence de η sur le comportement de convergence et le temps de calcul de HSS.

Théorème A.23. *Si la méthode de MG est utilisée pour résoudre le système hermitien avec la matrice* $\mathscr{M}_1$*, alors*

$$\lim_{n\to\infty} \sqrt{\tilde{\Gamma}_n - \gamma\alpha_n^{\mathrm{RA2}} + \gamma} = \gamma_*.$$

A.3.4 Résultats expérimentaux

Les algorithmes ont été codés en MATLAB R2018b. Soit η le nombre d'itérations préadaptatives. La figure A.3 représente l'influence de paramètre η sur le comportement de convergence et le temps de calcul pour la méthode de HSS.

A.4 Réduction des coûts de communication

A.4.1 Différents schémas des itérations de gradient

Les algorithmes de différents schémas des itérations de gradient peuvent être résumés comme ci-dessous:

Algorithme A.1 Schéma parallèle.

calculer $A_i g_n$
calculer et synchroniser les coefficients
calculer α_n
calculer $x_{i,n+1}$, $g_{i,n+1}$
pour $j \in$ voisins dépendants **faire**
    envoyer g_i au processus j
fin pour
pour $j \in$ voisins essentiels **faire**
    recevoir g_j du processus j
fin pour
calculer le résidu
atteindre la fin des requêtes

Algorithme A.2 Schéma parallèle cyclique.

calculer $A_i g_n$
si $n \bmod d = 0$ **alors**
    calculer et synchroniser les coefficients
    calculer α_n
fin si
calculer $x_{i,n+1}$, $g_{i,n+1}$
pour $j \in$ voisins dépendants **faire**
    envoyer g_i au processus j
fin pour
pour $j \in$ voisins essentiels **faire**
    recevoir g_j du processus j
fin pour
calculer le résidu
atteindre la fin des requêtes

Algorithme A.3 Schéma asynchrone.

1: calculer $A_i g_n$
2: calculer $x_{i,n+1}$, $g_{i,n+1}$
3: **pour** $j \in$ voisins dépendants **faire**
4: envoyer g_i au processus j
5: **fin pour**
6: **pour** $j \in$ voisins essentiels **faire**
7: recevoir g_j du processus j
8: **fin pour**
9: calculer le résidu

Un exemple typique des méthodes cycliques est la méthode de Cauchy cyclique (CSD) [74]

$$\alpha_n^{\mathrm{CSD}} = \alpha_{\tau(n)}^{\mathrm{SD}}, \quad \tau(n) = \max\{j \leq n : j \bmod d = 0\},$$

avec $d \geq 1$. Soit $\mathscr{F}_i$ une application utilisée par le processeur i. On suppose que $\mathscr{P}_n \subset \{1, \ldots, v\}$ pour tout n est un sous-ensemble de processeurs. Le schéma itératif asynchrone peut s'écrire sous la forme suivante:

$$x_{i,n+1} = \begin{cases} \mathscr{F}_i\left(x_{1,\hat{\tau}_{i,1}(n)}, \ldots, x_{v,\hat{\tau}_{i,v}(n)}\right), & i \in \mathscr{P}_n, \\ x_{i,n}, & \text{autres cas,} \end{cases}$$

où $\hat{\tau}_{i,j}(n)$ est un entier positif satisfaisant $\hat{\tau}_{i,j}(n) \leq n$ pour chaque élément j dans chaque processeur i. Un exemple trivial est la méthode de Richardson asynchrone [24].

A.4.2 Méthodes du gradient s-dimensionnelles

On considère l'ensemble suivant:

$$L_n^{(s)} = \left\{ x_n - \sum_{j=1}^{s} \alpha_n^{(j)} A^{j-1} g_n : \alpha_n^{(j)} \in \mathbb{R} \right\}.$$

La méthode de Cauchy s-dimensionnelle (s-SD) est sous la forme

$$x_{n+1} = x_n - \alpha_n^{(1)} g_n - \cdots - \alpha_n^{(s)} A^{s-1} g_n,$$

où $\alpha_n^{(1)}, \ldots, \alpha_n^{(s)}$ sont sélectionnés pour minimiser la fonction quadratique. La récurrence du vecteur gradient peut s'écrire sous la forme

$$g_{n+1} = g_n - \alpha_n^{(1)} A g_n - \cdots - \alpha_n^{(s)} A^s g_n.$$

On sait que g_{n+1} devrait être orthogonal à l'espace $L_n^{(s)}$. En conséquence, on trouve le système d'équations suivant

$$\begin{array}{ccccccc}
\alpha_n^{(1)} g_n^\intercal A g_n & + & \ldots & + & \alpha_n^{(s)} g_n^\intercal A^s g_n & = & g_n^\intercal g_n, \\
\alpha_n^{(1)} (A g_n)^\intercal A g_n & + & \ldots & + & \alpha_n^{(s)} (A g_n)^\intercal A^s g_n & = & (A g_n)^\intercal g_n, \\
\vdots & & & & \vdots & & \vdots \\
\alpha_n^{(1)} (A^{s-1} g_n)^\intercal A g_n & + & \ldots & + & \alpha_n^{(s)} (A^{s-1} g_n)^\intercal A^s g_n & = & (A^{s-1} g_n)^\intercal g_n.
\end{array}$$

Soit $\omega_n^{(j)} = g_n^\intercal A^j g_n$. En tenant compte de $(A^k g_n)^\intercal A^l g_n = g_n^\intercal A^{k+l} g_n$, il vient

$$\begin{pmatrix}
\omega_n^{(1)} & \omega_n^{(2)} & \cdots & \omega_n^{(s)} \\
\omega_n^{(2)} & \omega_n^{(3)} & \cdots & \omega_n^{(s+1)} \\
\vdots & \vdots & & \vdots \\
\omega_n^{(s)} & \omega_n^{(s+1)} & \cdots & \omega_n^{(2s-1)}
\end{pmatrix}
\begin{pmatrix}
\alpha_n^{(1)} \\ \alpha_n^{(2)} \\ \vdots \\ \alpha_n^{(s)}
\end{pmatrix} =
\begin{pmatrix}
\omega_n^{(0)} \\ \omega_n^{(1)} \\ \vdots \\ \omega_n^{(s-1)}
\end{pmatrix}. \tag{A.3}$$

La version parallèle de cette méthode est présentée dans l'algorithme A.4. On propose deux nouvelles propriétés dans les théorèmes suivants:

Théorème A.24. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Soit $\Gamma_n = \{\alpha_n^{(1)}, \ldots, \alpha_n^{(s)}\}$. Si la suite de solutions $\{x_n\}$ est générée par s-SD, alors*

$$\max_{\alpha \in \Gamma_n} |\alpha| \geq \frac{1}{\lambda_N + \cdots + \lambda_N^s}$$

Théorème A.25. *On considère le système $Ax = b$ où A est une matrice symétrique définie positive. Suppose que A n'est pas une matrice scalaire. Si la suite de solutions $\{x_n\}$ est générée par 2-SD, alors on a $\alpha_n^{(1)} > 0$ et $\alpha_n^{(2)} < 0$.*

Nous avons les observations suivantes:

- SD and s-SD ont tous la propriété d'invariance en deux étapes [68].

Algorithme A.4 Méthode de s-SD parallèle.

$g_{i,0} = -b_i$
$\omega_0^{(0)} = g_{i,0}^{\intercal} g_{i,0}$
`Allreduce`($\omega_0^{(0)}$, `SUM`)
calculer le résidu initial
pour $n = 0, 1, \ldots$ **faire**
    set $u_{i,n} = g_{i,n}$
    pour $j = 1, \ldots, s$ **faire**
        `Allgather`(u_n)
        $p_{i,n}^{(j)} = A_i u_n$
        $u_{i,n} = p_{i,n}^{(j)}$
    fin pour
    calculer $\omega_n^{(1)}, \ldots, \omega_n^{(2s-1)}$
    `Allreduce`($[\omega_n^{(1)} \ldots \omega_n^{(2s-1)}]$, `SUM`)
    résoudre le système de Hankel
    $x_{i,n+1} = x_{i,n} - \alpha_n^{(1)} g_{i,n} - \alpha_n^{(2)} p_{i,n}^{(1)} - \cdots - \alpha_n^{(s)} p_{i,n}^{(s-1)}$
    $g_{i,n+1} = g_{i,n} - \alpha_n^{(1)} p_{i,n}^{(1)} - \alpha_n^{(2)} p_{i,n}^{(2)} - \cdots - \alpha_n^{(s)} p_{i,n}^{(s)}$
    $\omega_{n+1}^{(0)} = g_{i,n+1}^{\intercal} g_{i,n+1}$
    `Allreduce`($\omega_{n+1}^{(0)}$, `SUM`)
    calculer le résidu
fin pour

- Les méthodes du gradient sont moins sensibles à l'erreur d'arrondi que la méthode du gradient conjugué.
- Le calcul et la communication peuvent tous être réduits en imposant des itérations à retards de façon cyclique.

Il est possible d'ajouter des retards dans les itérations de s-SD de la forme suivante:

$$\hat{\alpha}_n = \frac{\omega_{n-r}^{(j)}}{\omega_{n-r}^{(j+1)}} = \frac{g_{n-r}^{\intercal} A^j g_{n-r}}{g_{n-r}^{\intercal} A^{j+1} g_{n-r}},$$

avec $1 \leq r < d$ et $0 \leq j < 2s-1$, conduisant à l'algorithme A.5.

Algorithme A.5 Méthode de Cs-SD parallèle.

1: identique aux lignes 1 à 4 dans l'algorithme A.4
2: **pour** $n = 0, 1, \ldots$ **faire**
3: **si** $n \bmod d = 0$ **alors**
4: identique aux lignes 6 à 16 dans l'algorithme A.4
5: $r = 1$
6: **sinon**
7: `Allgather`(g_n)
8: $p_{i,n} = A_i g_n$
9: calculer $\hat{\alpha}_n$ by $\omega_{n-r}^{(1)}, \ldots, \omega_{n-r}^{(2s-1)}$
10: $x_{i,n+1} = x_{i,n} - \hat{\alpha}_n g_{i,n}$
11: $g_{i,n+1} = g_{i,n} - \hat{\alpha}_n p_{i,n}$
12: $r = r + 1$
13: **fin si**
14: identique aux lignes 17 à 19 dans l'algorithme A.4
15: **fin pour**

On sait que

$$\frac{\omega_n^{(2s-2)}}{\omega_n^{(2s-1)}} < \cdots < \frac{\omega_n^{(0)}}{\omega_n^{(1)}}.$$

Donc on peut choisir $j = r - 1$ et obtenir la méthode de Cs-SD amortie. De manière similaire, on peut formaliser la méthode de SDC s-dimensionnelle illustrée dans l'algorithme A.6.

Algorithme A.6 Méthode de s-CSD parallèle.

1: identique aux lignes 1 à 4 dans l'algorithme A.4
2: **pour** $n = 0, 1, \ldots$ **faire**
3: **si** $n \bmod d = 0$ **alors**
4: identique aux lignes 6 à 16 dans l'algorithme A.4
5: $\tilde{\alpha} = \omega_n^{(0)} / \omega_n^{(1)}$
6: $\theta = \omega_n^{(0)}$
7: **sinon**
8: `Allgather`(g_n)
9: $p_{i,n} = A_i g_n$
10: **si** $n \bmod d = 1$ **alors**
11: $\delta = p_{i,n}^{\mathsf{T}} g_{i,n}$
12: `Allreduce`$(\delta,$ `SUM`$)$
13: $\hat{\alpha} = \omega_n^{(0)} / \delta$
14: $\alpha_n = 2/(\sqrt{(1/\tilde{\alpha} - 1/\hat{\alpha})^2 + 4\omega_n^{(0)}/(\tilde{\alpha}^2\theta)} + 1/\tilde{\alpha} + 1/\hat{\alpha})$
15: $r = 0$
16: **fin si**
17: $x_{i,n+1} = x_{i,n} - \alpha_{n-r} g_{i,n}$
18: $g_{i,n+1} = g_{i,n} - \alpha_{n-r} p_{i,n}$
19: $r = r + 1$
20: **fin si**
21: identique aux lignes 17 à 19 dans l'algorithme A.4
22: **fin pour**

Nous faisons maintenant les hypothèses simplificatrices suivantes:

- La communication est le principal goulet d'étranglement par rapport au calcul, dans lequel la latence joue un rôle dominant par rapport à la bande passante.
- Nous comptons les opérations de réduction et d'assemblage comme générant le même coût de communication.
- Nous n'utilisons pas la synchronisation bloquante pour le calcul du résidu.
- L'itération $1/s$ dans s-SD est considérée comme équivalente à l'itération 1 dans SD ou CSD.

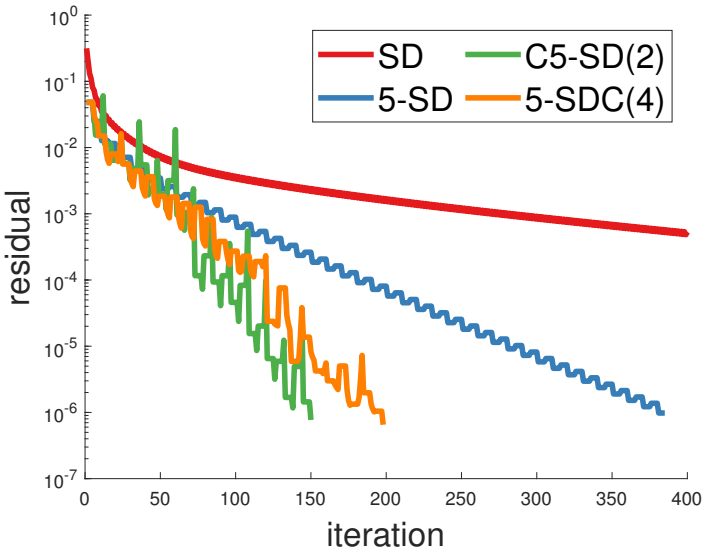

Figure A.4 Résultat des méthodes du gradient s-dimensionnelles à retards.

La Table A.1 présente la comparaison des différentes méthodes du gradient en utilisant ces hypothèses.

Table A.1 Nombre moyen d'opérations de réduction et d'assemblage (opération-re et opération-as).

	SD	s-SD	CSD	SDC	Cs-SD	s-SDC
opération-re	1	$1/s$	$1/d$	$(d_1+1)/(d_1+d_2)$	$1/(s+d-1)$	$2/(s+d-1)$
opération-as	1	1	1	1	1	1

A.4.3 Résultats expérimentaux

Les algorithmes ont été codés en MATLAB R2018b. La figure A.4 représente le résultat des méthodes du gradient s-dimensionnelles à retards: Cs-SD et s-SDC, comparées avec SD et s-SD.

References

[1] H. Akaike. On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method. *Ann. Inst. Stat. Math.*, 11(1): 1–16, 1959.

[2] J. A. Álvarez-Dios, J. C. Cabaleiro, and G. Casal. A generalization of *s*-step variants of gradient methods. *J. Comput. Appl. Math.*, 236(12):2938–2953, 2012.

[3] H. Anzt, S. Tomov, J. J. Dongarra, and V. Heuveline. A block-asynchronous relaxation method for graphics processing units. *J. Parallel Distrib. Comput.*, 73(12):1613–1626, 2013.

[4] W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Q. Appl. Math.*, 9(1):17–29, 1951.

[5] U. M. Ascher, K. van den Doel, H. Huang, and B. F. Svaiter. Gradient descent and fast artificial time tntegration. *Math. Model. Numer. Anal.*, 43(4):689–708, 2009.

[6] O. Axelsson and A. Kucherov. Real valued iterative methods for solving complex symmetric linear systems. *Numer. Linear Algebra Appl.*, 7(4):197–218, 2000.

[7] J. M. Bahi, S. Domas, and K. Mazouzi. Jace: a Java environment for distributed asynchronous iterative computations. In *12th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, pages 350–357. IEEE, 2004.

[8] J. M. Bahi, S. Contassot-Vivier, R. Couturier, and F. Vernier. A decentralized convergence detection algorithm for asynchronous parallel iterative algorithms. *IEEE Trans. Parallel Distrib. Syst.*, 16(1):4–13, 2005.

[9] J. M. Bahi, S. Contassot-Vivier, and R. Couturier. *Parallel Iterative Algorithms: From Sequential to Grid Computing*. CRC Press, London, UK, 2007.

[10] Z.-Z. Bai. Several splittings for non-Hermitian linear systems. *Sci. China Ser. A*, 51 (8):1339–1348, 2008.

[11] Z.-Z. Bai, G. H. Golub, and M. K. Ng. Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems. *SIAM J. Matrix Anal. Appl.*, 24(3):603–626, 2003.

[12] Z.-Z. Bai, G. H. Golub, and M. K. Ng. On successive-overrelaxation acceleration of the Hermitian and skew-Hermitian splitting iterations. *Numer. Linear Algebra Appl.*, 14(4):319–335, 2007.

[13] Z.-Z. Bai, M. Benzi, and F. Chen. Modified HSS iteration methods for a class of complex symmetric linear systems. *Computing*, 87(3):93–111, 2010.

[14] Z.-Z. Bai, M. Benzi, and F. Chen. On preconditioned MHSS iteration methods for complex symmetric linear systems. *Numer. Algorithms*, 56(2):297–317, 2011.

[15] Z.-Z. Bai, M. Benzi, F. Chen, and Z.-Q. Wang. Preconditioned MHSS iteration methods for a class of block two-by-two linear systems with applications to distributed control problems. *IMA J. Numer. Anal.*, 33(1):343–369, 2013.

[16] R. Barrett, M. W. Berry, T. F. Chan, J. Demmel, J. Donato, J. J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. A. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, 1994.

[17] J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA J. Numer. Anal.*, 8(1):141–148, 1988.

[18] G. M. Baudet. Asynchronous iterative methods for multiprocessors. *J. ACM*, 25(2): 226–244, 1978.

[19] M. Benzi. Preconditioning techniques for large linear systems: A survey. *J. Comput. Phys.*, 182(2):418–477, 2002.

[20] M. Benzi. A generalization of the Hermitian and skew-Hermitian splitting iteration. *SIAM J. Matrix Anal. Appl.*, 31(2):360–374, 2009.

[21] M. Benzi and D. Bertaccini. Block preconditioning of real-valued iterative algorithms for complex linear systems. *IMA J. Numer. Anal.*, 28(3):598–618, 2008.

[22] D. Bertaccini, G. H. Golub, S. S. Capizzano, and C. T. Possio. Preconditioned HSS methods for the solution of non-Hermitian positive definite linear systems and applications to the discrete convection-diffusion equation. *Numer. Math.*, 99(3): 441–484, 2005.

[23] D. P. Bertsekas. Distributed asynchronous computation of fixed points. *Math. Program.*, 27(1):107–120, 1983.

[24] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Upper Saddle River, 1989.

[25] E. G. Birgin, J. M. Martínez, and M. Raydan. Spectral projected gradient methods: Review and perspectives. *J. Stat. Softw.*, 60(3):539–559, 2014.

[26] M. S. Birman. Some estimates for the method of steepest descent. *Uspekhi Mat. Nauk*, 5(3)(37):152–155, 1950. (in Russian).

[27] A. Bunse-Gerstner and R. Stöver. On a conjugate gradient-type method for solving complex symmetric linear systems. *Linear Algebra Appl.*, 287(1):105–123, 1999.

[28] E. C. Carson. *Communication-Avoiding Krylov Subspace Methods in Theory and Practice*. PhD thesis, UC Berkeley, 2015.

[29] E. C. Carson. The adaptive *s*-step conjugate gradient method. *SIAM J. Matrix Anal. Appl.*, 39(3):1318–1338, 2018.

[30] E. C. Carson and J. W. Demmel. Accuracy of the *s*-step Lanczos method for the symmetric eigenproblem in finite precision. *SIAM J. Matrix Anal. Appl.*, 36(2): 793–819, 2015.

[31] E. C. Carson, N. Knight, and J. W. Demmel. Avoiding communication in nonsymmetric Lanczos-based Krylov subspace methods. *SIAM J. Sci. Comput.*, 35(5):S42–S61, 2013.

[32] E. C. Carson, N. Knight, and J. W. Demmel. A residual replacement strategy for improving the maximum attainable accuracy of *s*-step Krylov subspace methods. *SIAM J. Matrix Anal. Appl.*, 35(1):22–43, 2014.

[33] A. L. Cauchy. Méthode générale pour la résolution des systèmes d'équations simultanées. *Comp. Rend. Sci. Paris*, 25(1):536–538, 1847. (in French).

[34] M. Chau. *Algorithmes Parallèles Asynchrones pour la Simulation Numérique*. PhD thesis, Institut National Polytechnique de Toulouse, 2005.

[35] D. Chazan and W. Miranker. Chaotic relaxation. *Linear Algebra Appl.*, 2(2):199–222, 1969.

[36] E. Chow, H. Anzt, and J. J. Dongarra. Asynchronous iterative algorithm for computing incomplete factorizations on GPUs. In J. M. Kunkel and T. Ludwig, editors, *High Performance Computing: 30th International Conference, ISC High Performance 2015*, pages 1–16, Cham, 2015. Springer.

[37] A. T. Chronopoulos. *s*-step iterative methods for (non)symmetric (in)definite linear systems. *SIAM J. Numer. Anal.*, 28(6):1776–1789, 1991.

[38] A. T. Chronopoulos and C. W. Gear. *s*-step iterative methods for symmetric linear systems. *J. Comput. Appl. Math.*, 25(2):153–168, 1989.

[39] A. T. Chronopoulos and D. R. Kincaid. On the Odir iterative method for non-symmetric indefinite linear systems. *Numer. Linear Algebra Appl.*, 8(2):71–82, 2001.

[40] A. T. Chronopoulos and A. B. Kucherov. Block *s*-step Krylov iterative methods. *Numer. Linear Algebra Appl.*, 17(1):3–15, 2010.

[41] A. T. Chronopoulos and C. D. Swanson. Parallel iterative *s*-step methods for unsymmetric linear systems. *Parallel Comput.*, 22(5):623–641, 1996.

[42] S. Cools and W. Vanroose. The communication-hiding pipelined BiCGstab method for the parallel solution of large unsymmetric linear systems. *Parallel Comput.*, 65: 1–20, 2017.

[43] R. Couturier and S. Domas. CRAC: a grid environment to solve scientific applications with asynchronous iterative algorithms. In *2007 IEEE International Parallel and Distributed Processing Symposium*, pages 1–8. IEEE, 2007.

[44] F. E. Curtis and W. Guo. *R*-linear convergence of limited memory steepest descent. *IMA J. Numer. Anal.*, 38(2):720–742, 2018.

[45] Y.-H. Dai. Alternate step gradient method. *Optimization*, 52(4-5):395–415, 2003.

[46] Y.-H. Dai. A new analysis on the Barzilai-Borwein gradient method. *J. Oper. Res. Soc. n.a.*, 1(2):187–198, 2013.

[47] Y.-H. Dai and R. Fletcher. On the asymptotic behaviour of some new gradient methods. *Math. Program.*, 103(3):541–559, 2005.

[48] Y.-H. Dai and L.-Z. Liao. *R*-linear convergence of the Barzilai and Borwein gradient method. *IMA J. Numer. Anal.*, 22(1):1–10, 2002.

[49] Y.-H. Dai and X.-Q. Yang. A new gradient method with an optimal stepsize property. *Comput. Optim. Appl.*, 33(1):73–88, 2006.

[50] Y.-H. Dai and Y.-X. Yuan. Alternate minimization gradient method. *IMA J. Numer. Anal.*, 23(3):377–393, 2003.

[51] Y.-H. Dai and Y.-X. Yuan. Analysis of monotone gradient methods. *J. Ind. Manag. Optim.*, 1(2):181–192, 2005.

[52] Y.-H. Dai, W. W. Hager, K. Schittkowski, and H. Zhang. The cyclic Barzilai-Borwein method for unconstrained optimization. *IMA J. Numer. Anal.*, 26(3):604–627, 2006.

[53] T. A. Davis and Y. Hu. The University of Florida sparse matrix collection. *ACM Trans. Math. Softw.*, 38(1):1:1–1:25, 2011.

[54] D. Day and M. A. Heroux. Solving complex-valued linear systems via equivalent real formulations. *SIAM J. Sci. Comput.*, 23(2):480–498, 2001.

[55] R. De Asmundis. *New Gradient Methods: Spectral Properties and Steplength Selection*. PhD thesis, Sapienza University of Rome, 2014.

[56] R. De Asmundis, D. di Serafino, F. Riccio, and G. Toraldo. On spectral properties of steepest descent methods. *IMA J. Numer. Anal.*, 33(4):1416–1435, 2013.

[57] R. De Asmundis, D. di Serafino, W. W. Hager, G. Toraldo, and H. Zhang. An efficient gradient method using the Yuan steplength. *Comput. Optim. Appl.*, 59(3):541–563, 2014.

[58] R. De Asmundis, D. di Serafino, and G. Landi. On the regularizing behavior of the SDA and SDC gradient methods in the solution of linear ill-posed problems. *J. Comput. Appl. Math.*, 302:81–93, 2016.

[59] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct Methods for Sparse Matrices*. Oxford University Press, 2nd edition, 2017.

[60] D. El Baz, P. Spiteri, J.-C. Miellou, and D. Gazen. Asynchronous iterative algorithms with flexible communication for nonlinear network flow problems. *J. Parallel Distrib. Comput.*, 38(1):1–15, 1996.

[61] D. El Baz, A. Frommer, and P. Spiteri. Asynchronous iterations with flexible communication: Contracting operators. *J. Comput. Appl. Math.*, 176(1):91–103, 2005.

[62] M. N. El Tarazi. Some convergence results for asynchronous algorithms. *Numer. Math.*, 39(3):325–340, 1982. (in French).

[63] M. N. El Tarazi. Algorithmes mixtes asynchrones. etude de convergence monotone. *Numer. Math.*, 44(3):363–369, 1984. (in French).

[64] R. Fletcher. Conjugate gradient methods for indefinite systems. In G. A. Watson, editor, *Numerical Analysis*, pages 73–89, Berlin Heidelberg, 1976. Springer.

[65] R. Fletcher. On the Barzilai-Borwein method. In L. Qi, K. Teo, and X. Yang, editors, *Optimization and Control with Applications*, pages 235–256. Springer, Boston, MA, 2005.

[66] R. Fletcher. A limited memory steepest descent method. *Math. Program.*, 135(1): 413–436, 2012.

[67] G. E. Forsythe. Solving linear algebraic equations can be interesting. *Bull. Amer. Math. Soc.*, 59:299–329, 1953.

[68] G. E. Forsythe. On the asymptotic directions of the s-dimensional optimum gradient method. *Numer. Math.*, 11(1):57–76, 1968.

[69] S. P. Frankel. Convergence rates of iterative treatments of partial differential equations. *MTAC*, 4(30):65–75, 1950.

[70] G. Frassoldati, L. Zanni, and G. Zanghirati. New adaptive stepsize selections in gradient methods. *J. Ind. Manag. Optim.*, 4(2):299–312, 2008.

[71] R. W. Freund. Conjugate gradient-type methods for linear systems with complex symmetric coefficient matrices. *SIAM J. Sci. Stat. Comput.*, 13(1):425–448, 1992.

[72] R. W. Freund. A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems. *SIAM J. Sci. Comput.*, 14(2):470–482, 1993.

[73] R. W. Freund and N. M. Nachtigal. QMR: A quasi-minimal residual method for non-Hermitian linear systems. *Numer. Math.*, 60(1):315–339, 1991.

[74] A. Friedlander, J. M. Martínez, B. Molina, and M. Raydan. Gradient method with retards and generalizations. *SIAM J. Numer. Anal.*, 36(1):275–289, 1999.

[75] A. Frommer and D. B. Szyld. Asynchronous two-stage iterative methods. *Numer. Math.*, 69(2):141–153, 1994.

[76] A. Frommer and D. B. Szyld. Asynchronous iterations with flexible communication for linear systems. *Calculateurs Parallèles*, 10:91–103, 1998.

[77] A. Frommer and D. B. Szyld. On asynchronous iterations. *J. Comput. Appl. Math.*, 123(1-2):201–216, 2000.

[78] G. Gbikpi-Benissan. *Asynchronous Domain Decomposition Methods for Massively Parallel Computing*. PhD thesis, CentraleSupélec, 2017.

[79] G. Gbikpi-Benissan, Q. Zou, and F. Magoulès. *Asynchronous Iterative Methods: Programming Models and Parallel Implementation*. Institute of Computer Science, 2018.

[80] P. Ghysels, T. J. Ashby, K. Meerbergen, and W. Vanroose. Hiding global communication latency in the GMRES algorithm on massively parallel machines. *SIAM Journal on Scientific Computing*, 35(1):C48–C71, 2013.

[81] G. H. Golub and D. P. O'Leary. Some history of the conjugate gradient and Lanczos algorithms: 1948–1976. *SIAM Rev.*, 31(1):50–102, 1989.

[82] G. H. Golub and R. S. Varga. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods. *Numer. Math.*, 3:147–168, 1961.

[83] C. C. Gonzaga and R. M. Schneider. On the steepest descent algorithm for quadratic functions. *Comput. Optim. Appl.*, 63(2):523–542, 2016.

[84] A. Greenbaum. *Iterative Methods for Solving Linear Systems*. Society for Industrial and Applied Mathematics, 1997.

[85] L. Grigori and S. Moufawad. Communication avoiding ILU0 preconditioner. *SIAM J. Sci. Comput.*, 37(2):C217–C246, 2015.

[86] L. Grigori, S. Moufawad, and F. Nataf. Enlarged Krylov subspace conjugate gradient methods for reducing communication. *SIAM J. Matrix Anal. Appl.*, 37(2):744–773, 2016.

[87] L. Grippo, F. Lampariello, and S. Lucidi. A nonmonotone line search technique for Newton's method. *SIAM J. Numer. Anal.*, 23(4):707–716, 1986.

[88] G. Heinig and K. Rost. Fast algorithms for Toeplitz and Hankel matrices. *Linear Algebra Appl.*, 435(1):1–59, 2011.

[89] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Natl. Bur. Stand.*, 49(6):409–436, 1952.

[90] D. Hezari, V. Edalatpour, and D. K. Salkuyeh. Preconditioned GSOR iterative method for a class of complex symmetric system of linear equations. *Numer. Linear Algebra Appl.*, 22(4):761–776, 2015.

[91] M. Hoemmen. *Communication-Avoiding Krylov Subspace Methods*. PhD thesis, UC Berkeley, 2010.

[92] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 2nd edition, 2013.

[93] D. Imberti and J. Erhel. Varying the *s* in your *s*-step GMRES. *Electron. Trans. Numer. Anal.*, 47:206–230, 2017.

[94] C. G. J. Jacobi. Ueber eine neue auflösungsart der bei der methode der kleinsten quadrate vorkommenden lineären gleichungen. *Astronomische Nachrichten*, 22(20): 297–306, 1845. (in German).

[95] I. M. Khabaza. An iterative least-square method suitable for solving large sparse matrices. *Comput. J.*, 6(2):202–206, 1963.

[96] S. K. Kim and A. T. Chronopoulos. A class of Lanczos-like algorithms implemented on parallel computers. *Parallel Comput.*, 17(6):763–778, 1991.

[97] S. K. Kim and A. T. Chronopoulos. An efficient nonsymmetric Lanczos method on parallel vector computers. *J. Comput. Appl. Math.*, 42(3):357–374, 1992.

[98] V. S. Kozjakin and M. A. Krasnosel'skii. Some remarks on the method of minimal residues. *Numer. Funct. Anal. Optim.*, 4(3):211–239, 1982.

[99] M. A. Krasnosel'skii and S. G. Krein. An iteration process with minimal residuals. *Numer. Funct. Anal. Optim.*, 31(73)(2):315–334, 1952. (in Russian).

[100] J.-L. Lamotte, B. Molina, and M. Raydan. Smooth and adaptive gradient method with retards. *Math. Comput. Model.*, 36(9):1161–1168, 2002.

[101] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Natl. Bur. Stand.*, 45(4):255–282, 1950.

[102] C. Lanczos. Solution of systems of linear equations by minimized iterations. *J. Res. Natl. Bur. Stand.*, 49(1):33–53, 1952.

[103] P. Lascaux and R. Théodor. *Analyse Numérique Matricielle Appliquée à l'Art de l'Ingénieur. Tome 2: Méthodes Itératives*. Dunod, Paris, 2004. (in French).

[104] F. Magoulès and A.-K. Cheik Ahamed. Alinea: An advanced linear algebra library for massively parallel computations on graphics processing units. *Int. J. High Perform. Comput. Appl.*, 29(3):284–310, 2015.

[105] F. Magoulès and G. Gbikpi-Benissan. JACK: An asynchronous communication kernel library for iterative algorithms. *J. Supercomput.*, 73(8):3468–3487, 2017.

[106] F. Magoulès and G. Gbikpi-Benissan. Distributed convergence detection based on global residual error under asynchronous iterations. *IEEE Trans. Parallel Distrib. Syst.*, 29(4):819–829, 2018.

[107] F. Magoulès and G. Gbikpi-Benissan. JACK2: An MPI-based communication library with non-blocking synchronization for asynchronous iterations. *Adv. Eng. Softw.*, 119: 116–133, 2018.

[108] F. Magoulès and G. Gbikpi-Benissan. Asynchronous Parareal time discretization for partial differential equations. *SIAM J. Sci. Comput.*, 40(6):C704–C725, 2018.

[109] F. Magoulès and C. Venet. Asynchronous iterative sub-structuring methods. *Math. Comput. Simul.*, 145:34–49, 2018.

[110] F. Magoulès, D. B. Szyld, and C. Venet. Asynchronous optimized Schwarz methods with and without overlap. *Numer. Math.*, 137(1):199–227, 2017.

[111] F. Magoulès, G. Gbikpi-Benissan, and Q. Zou. Asynchronous iterations of Parareal algorithm for option pricing models. *Mathematics*, 6(4):1–18, 2018.

[112] J. A. Meijerink and H. A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Math. Comput.*, 31 (137):148–162, 1977.

[113] J.-C. Miellou. Algorithmes de relaxation chaotique à retards. *Math. Model. Numer. Anal.*, 9(R1):55–82, 1975. (in French).

[114] J.-C. Miellou and P. Spiteri. Un critère de convergence pour des méthodes générales de point fixe. *ESAIM: M2AN*, 19(4):645–669, 1985. (in French).

[115] J.-C. Miellou, D. El Baz, and P. Spiteri. A new class of asynchronous iterative algorithms with order intervals. *Math. Comput.*, 67(221):237–255, 1998.

[116] B. Molina and M. Raydan. Preconditioned Barzilai-Borwein method for the numerical solution of partial differential equations. *Numer. Algorithms*, 13(1):45–60, 1996.

[117] J. Nocedal, A. Sartenaer, and C. Zhu. On the behavior of the gradient norm in the steepest descent method. *Comput. Optim. Appl.*, 22(1):5–35, 2002.

[118] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975.

[119] V. Partimbene. *Calcul Haute Performance pour la Simulation d'Interactions Fluide-Structure*. PhD thesis, Institut National Polytechnique de Toulouse, 2018.

[120] D. W. Peaceman and H. H. Rachford. The numerical solution of parabolic and elliptic differential equations. *J. Soc. Indust. Appl. Math.*, 3(1):28–41, 1955.

[121] M. Pourbagher and D. K. Salkuyeh. On the solution of a class of complex symmetric linear systems. *Appl. Math. Lett.*, 76:14–20, 2018.

[122] L. Pronzato and A. A. Zhigljavsky. Gradient algorithms for quadratic optimization with fast convergence rates. *Comput. Optim. Appl.*, 50(3):597–617, 2011.

[123] L. Pronzato, H. P. Wynn, and A. A. Zhigljavsky. *An Introduction to Dynamical Search*, pages 115–150. Springer, Boston, MA, 2002.

[124] L. Pronzato, H. P. Wynn, and A. A. Zhigljavsky. Asymptotic behaviour of a family of gradient algorithms in $\mathbb{R}^d$ and Hilbert spaces. *Math. Program.*, 107(3):409–438, 2006.

[125] M. Raydan. *Convergence Properties of the Barzilai and Borwein Gradient Method*. PhD thesis, Rice University, 1991.

[126] M. Raydan. On the Barzilai and Borwein choice of steplength for the gradient method. *IMA J. Numer. Anal.*, 13(3):321–326, 1993.

[127] M. Raydan. The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM J. Optim.*, 7(1):26–33, 1997.

[128] M. Raydan and B. F. Svaiter. Relaxed steepest descent and Cauchy-Barzilai-Borwein method. *Comput. Optim. Appl.*, 21(2):155–167, 2002.

[129] J. K. Reid. The use of conjugate gradients for systems of linear equations possessing "Property A". *SIAM J. Numer. Anal.*, 9(2):325–332, 1972.

[130] L. F. Richardson. The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Philos. Trans. Royal Soc. A*, 210(459-470):307–357, 1910.

[131] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2nd edition, 2003.

[132] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, 1986.

[133] Y. Saad and H. A. van der Vorst. Iterative solution of linear systems in the 20th century. *J. Comput. Appl. Math.*, 123(1):1–33, 2000.

[134] D. K. Salkuyeh, D. Hezari, and V. Edalatpour. Generalized successive overrelaxation iterative method for a class of complex symmetric linear system of equations. *Int. J. Comput. Math.*, 92(4):802–815, 2015.

[135] S. A. Savari and D. P. Bertsekas. Finite termination of asynchronous iterative algorithms. *Parallel Comput.*, 22(1):39–56, 1996.

[136] P. L. Seidel. Uber ein verfahren die gleichungen, auf welche die methode der kleinsten quadrate führt, sowie lineare gleichungen überhaupt, durch successive annäherung aufzulösen. *Abh. Math. Phys. Kl. Bayer. Akad. Wiss.*, 11(3):81–108, 1874. (in German).

[137] J. W. Sheldon. On the numerical solution of elliptic difference equations. *MTAC*, 9 (51):101–112, 1955.

[138] V. Simoncini and D. B. Szyld. Recent computational developments in Krylov subspace methods for linear systems. *Numer. Linear Algebra Appl.*, 14(1):1–59, 2007.

[139] T. Sogabe and S.-L. Zhang. A COCR method for solving complex symmetric linear systems. *J. Comput. Appl. Math.*, 199(2):297–303, 2007.

[140] R. V. Southwell. *Relaxation Methods in Engineering Science: A Treatise on Approximate Computation*. Oxford University Press, 1940.

[141] R. Thakur, R. Rabenseifner, and W. Gropp. Optimization of collective communication operations in MPICH. *Int. J. High Perform. Comput. Appl.*, 19(1):49–66, 2005.

[142] E. E. Tyrtyshnikov. How bad are Hankel matrices? *Numer. Math.*, 67(2):261–269, 1994.

[143] K. van den Doel and U. M. Ascher. The chaotic nature of faster gradient descent methods. *J. Sci. Comput.*, 51(3):560–581, 2012.

[144] A. van der Sluis and H. A. van der Vorst. The rate of convergence of conjugate gradients. *Numer. Math.*, 48(5):543–560, 1986.

[145] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci, Stat. Comput.*, 13(2): 631–644, 1992.

[146] H. A. van der Vorst. *Iterative Krylov Methods for Large Linear Systems*. Cambridge University Press, 2003.

[147] H. A. van der Vorst and J. B. M. Melissen. A Petrov-Galerkin type method for solving $Ax = b$, where A is symmetric complex. *IEEE Trans. Magn.*, 26(2):706–708, 1990.

[148] R. S. Varga. *Matrix Iterative Analysis*, volume 27. Springer, Berlin Heidelberg, 2000.

[149] A. J. Wathen. Preconditioning. *Acta Numer.*, 24:329–376, 2015.

[150] S.-L. Wu. Several variants of the Hermitian and skew-Hermitian splitting method for a class of complex symmetric linear systems. *Numer. Linear Algebra Appl.*, 22(2): 338–356, 2015.

[151] D. M. Young. Iterative methods for solving partial difference equations of elliptic type. *Trans. Amer. Math. Soc.*, 76(1):92–111, 1954.

[152] D. M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, Orlando, 1971.

[153] D. M. Young. A historical overview of iterative methods. *Comput. Phys. Commun.*, 53(1):1–17, 1989.

[154] Y.-X. Yuan. A new stepsize for the steepest descent method. *J. Comput. Math.*, 24(2): 149–156, 2006.

[155] Y.-X. Yuan. Step-sizes for the gradient method. In K.-S. Lau, Z.-P. Xin, and S.-T. Yau, editors, *Third International Congress of Chinese Mathematicians*, pages 785–796. AMS/IP, 2008.

[156] Y.-X. Yuan. Gradient methods for large scale convex quadratic functions. In Y.-F. Wang, C.-C. Yang, and A. G. Yagola, editors, *Optimization and Regularization for Computational Inverse Problems and Applications*, pages 141–155. Springer, 2010.

[157] H. Zhang. *Méthodes Itératives à Retard pour Architecture Massivement Parallèles*. PhD thesis, CentraleSupélec, 2016.

[158] B. Zhou, L. Gao, and Y.-H. Dai. Gradient methods with adaptive step-sizes. *Comput. Optim. Appl.*, 35(1):69–86, 2006.

[159] Q. Zou and F. Magoulès. A new cyclic gradient method adapted to large-scale linear systems. In *Proceedings of the 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science*, pages 196–199, Wuxi, China, 2018. IEEE.

[160] Q. Zou and F. Magoulès. Convergence detection of asynchronous iterations based on modified recursive doubling. In *Proceedings of the 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science*, pages 288–291, Wuxi, China, 2018. IEEE.

[161] Q. Zou and F. Magoulès. Parallel iterative methods with retards for linear systems. In *Proceedings of the 6th International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*. Civil-Comp Press, 2019.

[162] Q. Zou, G. Gbikpi-Benissan, and F. Magoulès. Asynchronous Parareal algorithm applied to european option pricing. In *Proceedings of the 16th International Symposium on Distributed Computing and Applications for Business Engineering and Science*, pages 37–40, AnYang, China, 2017. IEEE.

[163] Q. Zou, G. Gbikpi-Benissan, and F. Magoulès. Asynchronous communications library for the parallel-in-time solution of Black-Scholes equation. In *Proceedings of the 16th International Symposium on Distributed Computing and Applications for Business Engineering and Science*, pages 45–48, AnYang, China, 2017. IEEE.

université PARIS-SACLAY
ÉCOLE DOCTORALE
INTERFACES
Approches Interdisciplinaires : Fondements, Applications et Innovation

Titre : Méthodes itératives à retard pour la résolution des systèmes linéaires à grande échelle

Mots clés : Méthodes du gradient avec alignement, Méthodes du gradient à retard, Propriétés spectrales, Splitting hermitien et anti-hermitien, Calcul parallèle

Résumé : Toute perturbation dans les systèmes linéaires peut gravement dégrader la performance des méthodes itératives lorsque les directions conjuguées sont constituées. Ce problème peut être partiellement résolu par les méthodes du gradient à retard, qui ne garantissent pas la descente de la fonction quadratique, mais peuvent améliorer la convergence par rapport aux méthodes traditionnelles. Les travaux ultérieurs se sont concentrés sur les méthodes du gradient alternées avec deux ou plusieurs types de pas afin d'interrompre le zigzag. Des papiers récents ont suggéré que la révélation d'information de second ordre avec des pas à retard pourrait réduire de manière asymptotique les espaces de recherche dans des dimensions de plus en plus petites. Ceci a conduit aux méthodes du gradient avec alignement dans lesquelles l'étape essentielle et l'étape auxiliaire sont effectuées en alternance. Des expériences numériques ont démontré leur efficacité. Cette thèse considère d'abord des méthodes du gradient efficaces pour résoudre les systèmes linéaires symétriques définis positifs. Nous commençons par étudier une méthode alternée avec la propriété de terminaison finie à deux dimensions. Ensuite, nous déduisons davantage de propriétés spectrales pour les méthodes du gradient traditionnelles. Ces propriétés nous permettent d'élargir la famille de méthodes du gradient avec alignement et d'établir la convergence de nouvelles méthodes. Nous traitons également les itérations de gradient comme un processus peu coûteux intégré aux méthodes de splitting. En particulier, nous abordons le problème de l'estimation de paramètre et suggérons d'utiliser les méthodes du gradient rapide comme solveurs internes à faible précision. Dans le cas parallèle, nous nous concentrons sur les formulations avec retard pour lesquelles il est possible de réduire les coûts de communication. Nous présentons également de nouvelles propriétés et méthodes pour les itérations de gradient s-dimensionnelles. En résumé, cette thèse s'intéresse aux trois sujets interreliés dans lesquelles les itérations de gradient peuvent être utilisées en tant que solveurs efficaces, qu'outils intégrés pour les méthodes de splitting et que solveurs parallèles pour réduire la communication. Des exemples numériques sont présentés à la fin de chaque sujet pour appuyer nos résultats théoriques.

Title : Iterative methods with retards for the solution of large-scale linear systems

Keywords : Gradient methods with alignment, Lagged gradient methods, Spectral properties, Hermitian and skew-Hermitian splitting, Parallel computing

Abstract : Any perturbation in linear systems may severely degrade the performance of iterative methods when conjugate directions are constructed. This issue can be partially remedied by lagged gradient methods, which does not guarantee descent in the quadratic function but can improve the convergence compared with traditional gradient methods. Later work focused on alternate gradient methods with two or more steplengths in order to break the zigzag pattern. Recent papers suggested that revealing of second-order information along with lagged steps could reduce asymptotically the search spaces in smaller and smaller dimensions. This led to gradient methods with alignment in which essential and auxiliary steps are conducted alternately. Numerical experiments have demonstrated their effectiveness. This dissertation first considers efficient gradient methods for solving symmetric positive definite linear systems. We begin by studying an alternate method with two-dimensional finite termination property. Then we derive more spectral properties for traditional steplengths. These properties allow us to expand the family of gradient methods with alignment and establish the convergence of new methods. We also treat gradient iterations as an inexpensive process embedded in splitting methods. In particular we address the parameter estimation problem and suggest to use fast gradient methods as low-precision inner solvers. For the parallel case we focus on the lagged formulations for which it is possible to reduce communication costs. We also present some new properties and methods for s-dimensional gradient iterations. To sum up, this dissertation is concerned with three inter-related topics in which gradient iterations can be employed as efficient solvers, as embedded tools for splitting methods and as parallel solvers for reducing communication. Numerical examples are presented at the end of each topic to support our theoretical findings.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France