Acceleration techniques of the Parareal algorithm for solving some differential equations

Van-Thanh Nguyen

Sorbonne Université

INRIA

Techniques d'accélération de l'algorithme Pararéel pour résoudre certaines équations différentielles

Thèse présentée par

Van-Thanh NGUYEN

École doctorale Sciences Mathématiques de Paris Centre
Unité de recherche Laboratoire Jacques-Louis Lions

Thèse dirigée par Laura GRIGORI

Composition du jury

Rapporteurs: Matthias BOLTEN, Professeur à Bergische Universität Wuppertal
Pascal OMNES, Directeur de recherche à CEA

Examinateurs: Yvon MADAY, Professeur à Sorbonne Université
Damien TROMEUR-DERVOUT, Professeur à Université Lyon-I
Sever HIRSTOAGA, Chargé de recherche à Inria

Directeur de thèse: Laura GRIGORI, Directrice de recherche à Inria

Sorbonne Université
INRIA

Acceleration techniques of the Parareal algorithm for solving some differential equations

PhD Thesis of

Van-Thanh NGUYEN

Doctoral school Sciences Mathématiques de Paris Centre
University Department Laboratoire Jacques-Louis Lions

Under the supervision of Laura GRIGORI
Committee members

Referees: Matthias BOLTEN, Professor at Bergische Universität Wuppertal
Pascal OMNES, Research director at CEA
Examiners: Yvon MADAY, Professor at Sorbonne Université
Damien TROMEUR-DERVOUT, Professor at Université Lyon-I
Sever HIRSTOAGA, Researcher at Inria
Supervisor: Laura GRIGORI, Research director at Inria

Inria Paris
2 rue Simone Iff
75012 paris
France
and
Laboratoire Jacques-Louis Lions
4 place Jussieu
75005 Paris
France

Keywords: Acceleration techniques, Parareal algorithm, differential equations

Mots Clés: Techniques d'accélération, algorithme Pararéel, équations différentielles

Acknowledgements

I would like to express my deep gratitude to my family, my teachers, my colleagues and my friends in Vietnam and also in France. I would like to say thank you to all of you who made my PhD period unforgettable.

First of all, from the bottom of my heart I would like to say thank you to Laura. Thank you for giving me a chance to work with you in this thesis. Thank you for your extremely useful advice, thank you for your interesting discussions and for your patience in explaining me new difficult problems. Especially, thank you for always giving big amount of time to think and to work independently in approaching and solving new problems. I also would like to say thank you for all the chances you gave me during my PhD period such at attending conferences where I presented our work and received lots of comments and met lots of experts to discuss in order to develop more and more. Thank you also for giving me a chance to work in a very professional and highly knowledgeable ALPINES team, where I received lots of comments and advice when I had some problem to discuss. Thank you for giving me a chance to work at INRIA, a very professional research environment where I received lots of support, where I met my colleagues who are always willing to help as well as give good advice during my PhD period.

I would like to say thank you to Prof.Dr. Matthias Bolten and Dr. Pascal Omnes for your very clear, detailed and careful reports as well as very helpful comments for my manuscript. I also would like to say thank you to Prof.Dr. Yvon Maday, Prof.Dr. Damien Tromeur-Dervout and Dr. Sever Hirstoaga for accepting to be part of the jury for my PhD defense. Thank you so much for your time and your valuable discussions.

I would like to say thank you to Dr. François Poitier, Dr. Martin Vohralík and Dr. Jacques Sainte-Marie for being my thesis follow-up committee. Thank you for your comments, your useful advice and your patience in helping me to finalize my PhD. I really appreciate your help.

I would like to thank you Prof.Dr. Yvon Maday for his kindness and highly knowledgeable in leading the parareal project CINE-PARA. I also would like to say thank you to Dr. Pascal Omnes, Dr. Caroline Japhet, Dr. Olga Mula for your discussion and advice at Cine-Para days. I acknowledge the French National Research Agency (ANR) Contract ANR-15-CE23-0019 (project CINE-PARA) for funding this thesis.

I would like to say thank you to Prof.Dr. Duong Minh Duc for giving me a chance to study in the Vietnam-France Master program. Thank you for your interesting lectures as well as useful advice during my study at the University of Science in HCM city. I also would like to say thank you to all my teachers: Dr. Ha, Dr. Tam, Dr. Hai, Dr. Phuong, Dr. Cam, Dr. Trung, Dr. Binh,... thank you very much for your interesting lectures and your valuable experiences.

I would like to say thank you to Dr. Martin Gander, thank you very much for your very interesting discussions at PinT conferences as well as your comments in developing my thesis.

I would like to deeply thank Dr. Hieu, my little teacher who helped me a lot during my first internship abroad, who always gave me extremely useful advice as well as instruction in academy and also in the life in France. Without you I couldn't have such interesting and memorable PhD period in France, thank you so much.

I would like to say thank you to all ALPINES members, and former members: Oliver, Jan, Sésbatien, Igor, Zakariae, Niu, Hussam,... thank you for your discussions, for your help, your experience, you made my PhD period full of nice memories. Especially I would like to deeply thank Matthias and Sever for their significantly contributions in finalizing two chapters in my thesis. Thank you so much Matthias for your patience in more than ten times discussions on RRQR. Thank you so much Sever for your knowledgeable advice as well as interesting discussions. Thank you so much again for your availability whenever I need to discuss, I really appreciate it.

I would like to say thank you to Laurence, Christine, Julien for your helps in administrative process as well as your encouragement in finalizing my PhD.

I would like to say thank you to all colleagues on the third floor, Fabien, Léa, Liudi, Haibo, Ngoc, Apolline, Matthieu, Julien,... thank you for being with me at the canteen, thank you for your sharing, thank you for your experience and your useful advice. I also would like to say thank you to Matthias, Suraj, Siwar, Edouard, Frédéric, Niels ... for your advice, your kindness and your humor beside the working time. Thank you Fabien for your jokes, for your sharing. Thank you Liudi for our discussion in a lot of fields. Thank you Siwar for being my officemate in the end of my thesis, thank you Edouard for your funny jokes, thank you Suraj for always abusing us when you are free, and thank you Matthias for your kindness as well as your useful help and advice during my PhD period.

I would like to say thank you to my Vietnamese friends: Binh, Son, Viet Anh, Dat, Diem, Thoi, Tam, The Anh, Duy, Thien, Quang, Tram, Thi, Hoan, Ngoc, Hieu, Huy, Bang, Hau, Huan, Tin, Linh... thank you for being with me in my first time living abroad. Thank you for sharing and helping me. You are an indispensable part of my PhD period.

I would like to say thank you to my family, to my parents who are always by my side (from Vietnam), who are always supporting, watching and encouraging me day by day. I also would like to say thank you to my sister, who always listened to me and encouraged me.

I would like to say thank you to my girlfriend Nhung. Thank you for coming to my life, thank you for being with me here and there, sharing memories, you make me happy and give me optimistic energies to work and to think about the future.

Finally I would like to say thank you to all people appeared in my life, whether long term or just a moment, whether good or bad memories. Thank you for making my life abundant.

Abstract

Parareal algorithm is well known in the literature for its ability in exploiting parallelism in time for solving time dependent problems. Having the same idea of domain decomposition methods, but instead of decomposing in space, parareal decomposes the time direction. In this thesis, we first present an interpretation of parareal as a two-level domain decomposition preconditioner so-called the two-level additive Schwarz in time preconditioner and, based on that, a variant that accelerates convergence by using a GMRES-type procedure. Connections to the MGRIT generalization of parareal are shown. The interpretation also allows us to derive convergence results for multiple variations of the method where we vary the number and order of coarse and fine level iterations. GMRES shows its potential in improving parareal's convergence rate, especially in solving reaction-advection-diffusion equation when advection and reaction coefficients are large. We also present the idea of using a different model so-called the reduced model which is based on the two-scale asymptotic expansion for the coarse propagator in parareal framework. This coupling strategy is studied for efficiently solving oscillatory singularly perturbed ODEs which are characteristics of a six-dimensional Vlasov equation. The coupling strategy shows its accuracy and efficiency in various numerical experiments with a uniform convergence rate. In the last part of the thesis, we study the acceleration of GMRES using a deflation technique of the smallest singular values of the problem. We present a new deflation method using sparse QR, especially Strong Rank Revealing QR (Strong RRQR) factorization with tournament pivoting strategy and nested dissection partition. The coupling strategy shows that the smallest singular values close to zero are replaced by larger ones. Furthermore, the combination between the deflation and the Block Jacobi preconditioner results in much faster convergence of GMRES.

Keywords: parareal, two-level additive Schwarz in time preconditioner, MGRIT, GMRES, reaction-advection-diffusion equation, two-scale asymptotic expansion, Vlasov equation, deflation technique, strong RRQR, tournament pivoting, nested dissection partition, Block Jacobi preconditioner.

Contents

List of Figures

List of Tables

List of Algorithms

Introduction (version française)

Sommaire du présent chapitre

1 Contexte

Le calcul parallèle joue un rôle très important en informatique et dans les mathématiques appliquées. Surtout avec un nombre croissant de processeurs et d'ordinateurs massivement parallèles, des millions voire des milliards de calculs peuvent être exécutés en même temps. Cela nous permet d'exploiter le parallélisme dans la résolution rapide et efficace des problèmes de Cauchy. En effet, les méthodes de décomposition de domaine sont bien connues dans la littérature, voir [16] pour plus de détails. Ces méthodes reposent sur la décomposition d'un problème posé dans un domaine en sous-problèmes posés dans des sous-domaines. Chaque sous-problème est alors résolu dans chaque sous-domaine et communique avec les autres sous-problèmes par la condition de transmission sur les interfaces. Pour un problème elliptique dans l'espace, il est prouvé que les méthodes de décomposition de domaine convergent indépendamment avec la taille du maillage si le chevauchement entre les sous-domaines est suffisamment grand, voir [16]. Cependant, lorsque le nombre de sous-domaines devient trop grand, le taux de convergence se détériore à cause du manque d'information globale couplant les sous-domaines. Afin d'obtenir un algorithme de décomposition de domaine évolutif qui dépend faiblement du nombre de sous-domaines, un espace grossier peut être utilisé pour coupler les informations globales de tous les sous-domaines. Cela conduit à l'idée de préconditionneurs de type décomposition de domaine à deux niveaux. Pour le premier niveau, une méthode de Schwarz additive ou multiplicative peut être appliquée, puis une correction d'espace grossier est effectuée pour coupler les informations de tous les sous-domaines. Sur des machines parallèles, ces méthodes passent bien à l'échelle sur des milliers de processeurs, mais leurs performances se dégradent lorsque le nombre de processeurs augmente encore. Il est donc important de développer de nouvelles méthodes qui exploitent également le parallélisme en temps.

Par conséquent, nous considérons dans ce travail l'algorithme pararéel et nous nous intéressons à accélérer sa convergence. L'approche pararéelle provient des méthodes de décomposition de domaine mais au lieu de traiter du parallélisme dans l'espace, elle traite du parallélisme dans le temps. S'appuyant sur la décomposition de tout le domaine temporel en sous-domaines temporels, l'algorithme pararéel utilise deux solveurs, un

grossier qui est facile et pas coûteux en termes de calcul et un solveur fin qui est plus précis mais plus coûteux en termes de coût de calcul. À partir d'une solution initiale obtenue en utilisant séquentiellement le solveur grossier, l'algorithme pararéel la corrige itérativement par la différence entre la solution fine obtenue en parallèle à l'aide du solveur fin et la solution grossière obtenue à l'itération précédente. Il a été montré dans [36] que l'algorithme peut être écrit comme un schéma de correction résiduelle au niveau grossier. Dans ce travail nous considérons le pararéel comme un schéma de correction résiduelle avec un préconditionneur au niveau fin dans le but d'améliorer sa convergence. Notre objectif est de construire une matrice de préconditionnement telle qu'elle agisse comme l'approche pararéelle au niveau fin. Cette matrice de préconditionnement, appelée SC, est formée par le produit de deux termes, le terme de Schwarz additif et le terme de correction temporelle grossière. Outre les méthodes de décomposition de domaine et le pararéel, les méthodes multigrilles sont également bien connues dans la littérature pour résoudre des EDP en parallèle, en particulier la méthode de réduction en temps multigrille (MGRIT), voir [20, 25, 36]. Dans cette thèse, nous avons d'abord prouvé que le préconditionneur SC en temps à deux niveaux de type Schwarz additif est équivalent à MGRIT avec F-relaxation. Ensuite, il a été montré dans [20, 25, 30] que l'algorithme pararéel est équivalent à MGRIT avec F-relaxation. Ainsi, les trois méthodes, pararéelle, MGRIT avec F-relaxation et le préconditionneur SC en temps à deux niveaux de type Schwarz additif sont équivalentes. Deuxièmement, nous montrons que des étapes supplémentaires de propagation fine ou grossière dans le préconditionneur conduisent à des procédures équivalentes à MGRIT avec FCF-relaxation et à MGRIT avec $F(CF)^2$-relaxation ou superposition pararéelle [36]. Nous proposons une variante, appelée SCS^2, qui converge plus rapidement et exploite efficacement le calcul parallèle. En outre, l'approche par le schéma de correction résiduelle au niveau fin avec ce préconditionneur temporel nous permet d'explorer l'utilisation des méthodes de sous-espace de Krylov, à savoir GMRES (Generalized Minimal Residual) [91] pour accélérer l'algorithme pararéel. Troisièmement, il est connu [78, 89, 99] que les petites valeurs propres qui sont proches de zéro affectent mal le comportement de convergence de GMRES, et ceci est également vrai pour les plus petites valeurs singulières [96]. Une idée très intéressante pour surmonter cette difficulté est de supprimer l'impact de ces petites valeurs singulières dans le spectre en utilisant un sous-espace de déflation. Suivant cette idée, dans ce travail nous étudions les approximations des plus petites valeurs singulières et de l'espace nul en utilisant pour la technique de déflation qui sont basées sur le QR sparse, en particulier le rang fort révélant la factorisation QR (strong RRQR) [19, 48] avec stratégie de pivotement de tournoi [19, 46]. Ici, notre objectif est d'utiliser le pivotement de tournoi pour obtenir un calcul efficace et hautement parallèle de l'espace à diminuer. Le pivotement de tournoi a été introduit pour approximer les plus grandes valeurs singulières et les vecteurs singuliers associés, et nous avons modifié l'algorithme de manière à pouvoir l'utiliser pour notre objectif. Enfin, dans le cadre de la résolution d'équations différentielles ordinaires fortement oscillatoires, nous appliquons l'idée d'utiliser un modèle (réduit) différent pour le solveur grossier de l'algorithme pararéel. L'avantage d'utiliser de tels modèles réduits est leur faible coût de calcul.

2 Résumé et contributions

Cette thèse a été financée par l'Agence Nationale de la Recherche (ANR) Contrat ANR-15-CE23-0019 (projet CINE-PARA), projet dans lequel nous menons des études sur l'algorithme pararéel afin d'accélérer ses performances. La thèse contient quatre chapitres.

Au chapitre 1, nous rappelons d'abord le contexte de l'algorithme pararéel, des préconditionneurs de décomposition de domaine à deux niveaux et de MGRIT afin de souligner leur équivalence. On rappelle également la méthode GMRES qui conduit à une accélération pararéelle. Ensuite, nous détaillons la factorisation RRQR forte, la stratégie de pivotement du tournoi et la technique de déflation afin de dériver un préconditionneur de déflation des plus petites valeurs singulières du problème pour accélérer le taux de convergence de GMRES. De plus, par une méthode de développements asymptotiques à deux échelles, nous obtenons des modèles réduits que nous utilisons pour la résolution grossière dans le cadre pararéel. Ces modèles sont des approximations d'équations différentielles ordinaires fortement oscillatoires qui sont les caractéristiques d'une équation de Vlasov à six dimensions.

Au chapitre 2, nous montrons l'équivalence entre pararéel, MGRIT avec F-relaxation et SC Schwarz additif à deux niveaux dans le préconditionneur temporel. Plus précisément, nous rappelons d'abord l'exécution pararéelle d'un point de vue algébrique qui conduit à l'expression de pararéel comme une itération stationnaire préconditionnée au niveau fin. Cette approche nous permet de décrire une interprétation du pararéel en tant que préconditionneur de Schwarz additif à deux niveaux dans le domaine temporel appelé préconditionneur de Schwarz additif à deux niveaux SC en temps. Sur la base de ce préconditionneur additif Schwarz en temps à deux niveaux, nous introduisons quelques variantes comme SCS, S(CS)^2 préconditionneur additif Schwarz en temps à deux niveaux et montrons leur équivalence à MGRIT avec FCF-relaxation, et à MGRIT avec F(CF)^2-relaxation ou superposition pararéelle. En outre, nous proposons une variante appelée SCS^2 additif Schwarz à deux niveaux dans le préconditionneur temporel qui converge plus rapidement et exploite efficacement le calcul parallèle. Dans la partie suivante, nous présentons l'analyse de convergence et l'estimation de convergence du SC Schwarz additif à deux niveaux dans le préconditionneur temporel et ses variantes. Nous discutons également des coûts de mise en œuvre et du compromis de ces variantes. Dans la suite de ce chapitre, nous menons des expériences numériques pour montrer l'équivalence entre l'algorithme pararéel et SC Schwarz additif à deux niveaux dans le préconditionneur temporel, la comparaison entre SC additif à deux niveaux Schwarz dans le préconditionneur temporel et ses variantes, le pararéel avec accélération GMRES pour le problème de Dahlquist, l'équation de la chaleur et l'équation d'advection-réaction-diffusion. En particulier, nous étudions l'impact de l'accélération GMRES du pararéel pour l'équation d'advection-réaction-diffusion dans deux cas: un dominé par la diffusion et l'autre dominé par l'advection. On verra que GMRES améliore la convergence pararéelle surtout dans le cas où les coefficients d'advection et de réaction sont grands devant le terme de diffusion. Enfin, nous montrons les comportements de convergence du pararéel et du pararéel avec accélération GMRES avec une méthode différente pour le propagateur fin.

Au chapitre 3, nous considérons la résolution d'équations de Vlasov raides en six dimensions, pour modéliser la dynamique de charges particules dans un champ magnétique fort. L'idée est d'utiliser des modèles réduits qui sont dérivés de développements asymptotiques à deux échelles et se sont avérés approximer l'équation initiale de Vlasov en des temps finaux courts. Cependant, nous avons trouvé par des expériences numériques que leur approximation devient imprécise en temps long. Dans ce cadre, nous appliquons l'algorithme pararéel en utilisant des modèles réduits pour la résolution grossière. Nous montrons numériquement la convergence rapide de l'algorithme pararéel pour plusieurs cas tests. Plus précisément, nous rappelons d'abord le développement asymptotique à deux échelles, puis les modèles réduits d'ordre zéro et d'ordre un, qui résultent de [28]. Ensuite, nous présentons le cas d'un champ magnétique constant, qui comprend un champ électrique uniforme qui varie dans le temps et un champ électrique stationnaire non uniforme, et le cas d'un champ magnétique variable. Ensuite, nos simulations montrent l'efficacité de la stratégie avec l'algorithme pararéel, en obtenant la convergence de l'algorithme en quelques itérations, alors que le nombre des fenêtres de temps est assez élevé. Les cas-test sont pertinents pour la physique des plasmas, comme dynamique de particules dans un piégeage de Penning, la séparation isotopique par résonance cyclotronique ionique et un exemple de dynamique d'une particule dans un champ magnétique variable.

Au chapitre 4, nous présentons la déflation des plus petites valeurs singulières basée sur le QR parcimonieux, en particulier la factorisation RRQR forte avec la stratégie de pivotement de tournoi. Plus précisément, nous rappelons d'abord la factorisation QR révélatrice du rang fort (strong RRQR) et la stratégie de pivotement du tournoi. Ensuite, nous présentons et analysons une étape de la stratégie de couplage avec partition de dissection imbriquée sur $\boldsymbol{A}^T\boldsymbol{A}$. Dans la partie suivante, nous dérivons une nouvelle technique de déflation des plus petites valeurs singulières basée sur une factorisation QR parcimonieuse avec une stratégie de pivotement de tournoi et une partition de dissection imbriquée. Nous combinons ensuite le préconditionneur de déflation et le préconditionneur de bloc Jacobi pour obtenir un taux de convergence plus rapide de GMRES. La déflation de vecteurs singuliers basée sur une factorisation RRQR forte est également présentée. Dans la suite de ce chapitre, nous effectuons des tests numériques pour montrer l'efficacité de la technique de déflation. En particulier, nous donnons d'abord une comparaison entre le GMRES déflaté par RRQR fort et le GMRES déflaté par SVD et discutons des coûts de mise en œuvre et du compromis. Ensuite, nous présentons les résultats numériques de la déflation de l'approximation des plus petites valeurs singulières par QR parcimonieux avec stratégie de pivotement de tournoi et partition de dissection imbriquée sur $\boldsymbol{A}^T\boldsymbol{A}$. Enfin, nous présentons la comparaison entre la déflation, le bloc Jacobi et les préconditionneurs mixtes s'appliquant au GMRES.

Dans la dernière partie de la thèse, nous donnons la conclusion et les perspectives.

Cette thèse a donné lieu aux publications suivantes.

Article de revue publié

L. Grigori, S. A. Hirstoaga, V. T. Nguyen and J. Salomon. *Reduced model-based parareal simulations of oscillatory singularly perturbed ordinary differential equations.* In: *Journal of Computational Physics*, Volume 436, 1 July 2021, 110282, ISSN 0021-9991,
https://doi.org/10.1016/j.jcp.2021.110282.

Article de revue soumis

L. Grigori, and V. T. Nguyen. *Interpretation of Parareal as a Two-level Additive Schwarz In Time Preconditioner and Its Acceleration with GMRES.* Inria Paris, Laboratoire Jacques-Louis Lions, Sorbonne Université, Paris, 2021.

Article de revue en préparation

M. Beaupère, L. Grigori, and V. T. Nguyen. *Deflation of smallest singular values based on strong RRQR and nested dissection partition with tournament pivoting strategy.* Inria Paris, Laboratoire Jacques-Louis Lions, Sorbonne Université, Paris, 2021.

Introduction (English version)

Contents

1 Context

Parallel computing plays a very important role in computer science. Especially with an increasing number of processors and massively parallel computers, millions even billions computations can be executed at the same time. This allows us to exploit parallelism in solving initial value problems quickly and effectively. Indeed, domain decomposition methods are well-known in the literature for that purpose, see [16] for further details. Those methods rely on decomposing the problem in the domain into subproblems in subdomains. Each subproblem is then solved in each subdomain and communicates with each other by the transmission condition on the boundaries. For an elliptic problem in space, domain decomposition methods are proven to converge independently with the mesh size if the overlapping between subdomains is large enough, see [16]. However, when the number of subdomains becomes too large, the convergence rate deteriorates because of the lack of global information coupling the subdomains. In order to obtain a scalable domain decomposition algorithm which depends weakly on the number of subdomains, a coarse space can be used to couple global information of all subdomains. This leads to the idea of two-level domain decomposition preconditioners. For the first level, additive or multiplicative Schwarz can be applied and then a coarse space correction is performed to couple information of all subdomains. On parallel machines those methods scale well on thousands of processors, however their performance degrades when the number of processors is increased further. It is important hence to develop new methods that exploit parallelism in the time direction as well.

Therefore, we consider in this work the parareal algorithm and we are interested in accelerating its convergence. The parareal approach comes from domain decomposition methods but instead of dealing with parallelism in space, it deals with parallelism in time. Relying on decomposing the whole time domain into time subdomains, parareal uses two solvers, a coarse one which is very cheap and easy to compute, and a fine one which is more accurate but more expensive in terms of computational cost. From an initial solution obtained by using sequentially the coarse solver, parareal iteratively corrects it by the difference between the fine solution obtained in parallel using the fine solver and the

coarse solution obtained from the previous iteration. It was shown in [36] that parareal can be written as a residual correction scheme at the coarse level. In this work we consider parareal as a residual correction scheme with a preconditioner at the fine level with the aim of improving its convergence. Our goal is to build a preconditioner matrix such that it acts like parareal at the fine level. This preconditioner matrix, called SC, is formed by the product of two terms, the additive Schwarz term and the coarse time correction term. Beside domain decomposition methods and parareal, multigrid methods are also well known in the literature for solving PDEs with parallelism, especially the multigrid reduction in time (MGRIT) method, see [20, 25, 36]. In this thesis, we first proved that the SC two-level additive Schwarz in time preconditioner is equivalent to MGRIT with F-relaxation. Then, it was shown in [20, 25, 30] that parareal is equivalent to MGRIT with F-relaxation. Thus, the three methods, parareal, MGRIT with F-relaxation and SC two-level additive Schwarz in time preconditioner are equivalent. Second, we show that additional fine or coarse propagation steps in the precondtioner lead to procedures equivalent to MGRIT with FCF-relaxation and to MGRIT with F(CF)^2-relaxation or overlapping parareal [36]. We propose a variant, called SCS^2, which converges faster and efficiently exploits parallel computing. Furthermore, approaching by the residual correction scheme at the fine level with SC two-level additive Schwarz in time preconditioner allows us to explore the usage of Krylov subspace methods, namely GMRES (Generalized Minimal Residual) [91] to accelerate the parareal algorithm. Third, it is known from [78, 89, 99] that the small eigenvalues which are close to zero badly affect the convergence behavior of GMRES, and this is also true for the smallest singular values [96]. A very interesting idea to overcome this difficulty is to remove the impact of those small singular values in the spectrum by using a deflation subspace. Following this idea, in this work we study the approximations of the smallest singular values and the null space using for the deflation technique which are based on the sparse QR , especially strong rank revealing QR factorization (strong RRQR) [19, 48] with tournament pivoting strategy [19, 46]. Here our goal is to use tournament pivoting to get an efficient and highly parallel computation of the space to be deflated. Tournament pivoting was introduced to approximate the largest singular values and associated singular vectors, and we have modified the algorithm such that we can use it for our goal. Finally, in the framework of solving highly oscillatory ordinary differential equations, we apply the idea of using a different (reduced) model for the coarse solver of parareal algorithm. The advantage of using such reduced models is their low computational cost.

2 Summary and contributions

This thesis was funded by the French National Research Agency (ANR) Contract ANR-15-CE23-0019 (project CINE-PARA) in which we conduct studies around parareal algorithm topic in order to accelerate its performance. This thesis contains four chapters.

In Chapter 1, we first recall the background of the parareal algorithm, two-level domain decomposition preconditioners and MGRIT in order to emphasize the equivalence between them. We also recall GMRES method which leads to parareal acceleration. Then, we detail the strong RRQR factorization, the tournament pivoting strategy and the deflation technique in order to derive a deflation preconditioner of the smallest singular values

of the problem to accelerate GMRES's convergence rate. A part from that, by means of two-scale asymptotic expansions, we obtain reduced models that we use for coarse solving in the parareal framework. These models are approximations of highly oscillatory ordinary differential equations which are characteristics of a six-dimensional Vlasov equation.

In Chapter 2, we show the equivalence between parareal, MGRIT with F-relaxation and SC two-level additive Schwarz in time preconditioner. More specifically, we first recall parareal execution from an algebraic point of view which leads to the expression of parareal as a preconditioned stationary iteration at the fine level. This approach allows us to describe an interpretation of parareal as a two-level additive Schwarz preconditioner in the time domain so-called SC two-level additive Schwarz in time preconditioner. Based on this two-level additive Schwarz in time preconditioner, we introduce some variants as SCS, $S(CS)^2$ two-level additive Schwarz in time preconditioner and show their equivalence to MGRIT with FCF-relaxation, and to MGRIT with $F(CF)^2$-relaxation or overlapping parareal. Additionally, we propose a variant referred to as SCS^2 two-level additive Schwarz in time preconditioner which converges faster and efficiently exploits parallel computing. In the following part, we present the convergence analysis and convergence estimate of SC two-level additive Schwarz in time preconditioner and its variants. We also discuss about the implementation costs and the trade-off of those variants. In the rest of this chapter, we conduct numerical experiments to show the equivalence between parareal and SC two-level additive Schwarz in time preconditioner, the comparison between SC two-level additive Schwarz in time preconditioner and its variants, the parareal with GMRES acceleration for Dahlquist problem, heat equation and advection-reaction-diffusion equation. In particular, we study the impact of GMRES acceleration of parareal for the advection-reaction-diffusion equation in two cases, diffusion dominated and advection dominated. It will be seen that GMRES improves parareal convergence especially in the case when the advection and reaction coefficients are large compared to the diffusion term. In the end, we show the convergence behaviors of parareal and parareal with GMRES acceleration with a different method for the fine propagator.

In Chapter 3, we consider solving stiff Vlasov equations in six dimensions, for modeling the dynamics of charged particles in a strong magnetic field. The idea is to use reduced models which are derived from two-scale asymptotic expansions and are proved to approximate the initial Vlasov equation in short final times. However, we found by numerical experiments that their approximation becomes inaccurate in long times. In this framework, we apply the parareal algorithm with the strategy of using reduced models for the coarse solving. We show numerically the rapid convergence of the parareal algorithm for several test cases. More specifically, we first recall the two-scale asymptotic expansion and then the zero and first order reduced models that result from [28]. Next, we present the case of a constant magnetic field, which includes a uniform time varying electric field and a non uniform stationary electric field, and the case of a variable magnetic field. Then, our simulations show the efficiency of the strategy with parareal, by obtaining convergence of the algorithm in a few iterations, while the number of time slices is quite high. The test cases are relevant for plasma physics, as particles in a Penning trap, isotope separation by ion cyclotron resonance and an example of dynamics in a variable magnetic field.

In Chapter 4, we present the deflation of the smallest singulars values based on the sparse QR, especially strong RRQR factorization with tournament pivoting strategy. More specifically, we first recall the strong rank revealing QR (strong RRQR) factorization and the tournament pivoting strategy. Then, we present and analyze one step of the coupling strategy with nested dissection partition on $\boldsymbol{A}^T\boldsymbol{A}$. In the following part, we derive a new deflation technique of smallest singular values based on sparse QR factorization with tournament pivoting strategy and nested dissection partition. We then combine the deflation preconditioner and block Jacobi preconditioner to obtain faster convergence rate of GMRES. The deflation of singular vectors based on strong RRQR factorization is also presented. In the rest of this chapter, we perform numerical tests to show the efficiency of the deflation technique. In particular, we first give a comparison between strong RRQR deflated GMRES and SVD deflated GMRES and discuss about the implementation costs and the trade-off. Then we presents numerical results of the deflation of smallest singular values approximation by sparse QR with tournament pivoting strategy and nested dissection partition on $\boldsymbol{A}^T\boldsymbol{A}$. Finally, we present the comparison between the deflation, block Jacobi and mixed preconditioners applying to GMRES.

In the last part of the thesis, we give the conclusion and perspectives.

This thesis has led to the following publications.

Journal paper published

L. Grigori, S. A. Hirstoaga, V. T. Nguyen and J. Salomon. *Reduced model-based parareal simulations of oscillatory singularly perturbed ordinary differential equations.* In: *Journal of Computational Physics*, Volume 436, 1 July 2021, 110282, ISSN 0021-9991,
https://doi.org/10.1016/j.jcp.2021.110282.

Journal paper submitted

L. Grigori, and V. T. Nguyen. *Interpretation of Parareal as a Two-level Additive Schwarz In Time Preconditioner and Its Acceleration with GMRES.* Inria Paris, Laboratoire Jacques-Louis Lions, Sorbonne Université, Paris, 2021.

Journal paper in preparation

M. Beaupère, L. Grigori, and V. T. Nguyen. *Deflation of smallest singular values based on strong RRQR and nested dissection partition with tournament pivoting strategy.* Inria Paris, Laboratoire Jacques-Louis Lions, Sorbonne Université, Paris, 2021.

1 State-of-the-art

Contents

1.1 Parareal algorithm

Introduced by J.L. Lions et at. [68] in 2001, parareal (parallel in real time) algorithm has become very attractive over the past 20 years for the solution of initial value problems. Because of its strong effectiveness in exploiting parallelism in time for real time problems, a wide range of its applications has been studied in various fields around the world. For example in the field of multiscale modeling, parareal can be very useful for molecular dynamics simulations [5], unsteady hydrodynamic simulations [23], the kinetic neutron diffusion equation [8, 9], the Navier-Stokes equations [27, 98], the Korteveg-deVries-Burgers' equations [61] or the Hamiltonian systems [17, 34]. In industry, parareal can also be applied for the morphological transformation in cubic alloys [55], heat transfer and heat flow [39], plasma turbulence [85, 93], automotive industry [70], fusion plasma edge [92], power systems dynamic simulations [22, 50, 51, 94]. Even in medical applications, parareal can also be used for skin transport [64]. In financial mathematics, parareal is also very efficient for the Black-Scholes equations using option pricing models for an American put [7, 72, 81]. Despite of its potential in exploiting parallelism in time for parabolic problems, parareal does not seem very efficient for hyperbolic equations [6, 26]. Difficulties come from dealing with complex eigenvalue problems that cause instability in convergence, mainly because of the regularity of the solution which strongly depends on the initial conditions [17]. An analysis has also been done in [87] for a better understanding of what causes instability problems for hyperbolic equations, and the author determines that the phase errors in the coarse propagator are the reasons so that with specifically tailored coarse level methods, the stability problems could be solved. Many interpretations as well

as variants of parareal have been studied. M. Gander and S. Vandewalle gave a derivation of the parareal algorithm as a multiple shooting method [37]. M. Minion and S. A. Williams investigated using spectral deferred corrections in the framework of parareal [75, 76]. Coupling parareal with non-overlapping domain decomposition method [49], parareal with Schwarz Waveform Relaxation methods [32, 35] are also very promising directions.

The parareal algorithm is a two-level method in which its idea comes from domain decomposition methods, but instead of decomposing in space, it decomposes the time direction. Therefore, the algorithm displays its advantage by covering various fields of applications where it exploits very efficiently parallel computing over a large number of processors to solve problems in real time constraint context. Since its conception, the algorithm has been intensively analyzed [6, 33, 37, 97]. The parareal algorithm's version used nowadays is much simpler and easier to implement than the original one but the main idea is still the same, see [37]. Let us briefly recall this approach. Consider the simple time dependent problem

$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u) \ \text{ in } (0,T),\ u(0) = \mathrm{u}_0. \tag{1.1}$$

The time interval $[0,T]$ is decomposed into N uniform time slices $[T_n, T_{n+1}]$, for $n \in \{0,\dots,N-1\}$. Let $\mathcal{F}(T_{n+1}, T_n, U_n)$ denote the fine solver, which gives a very accurate approximation of the solution at time T_{n+1} with the initial solution U_n at time T_n and let $\mathcal{G}(T_{n+1}, T_n, U_n)$ denote the coarse solver, which gives a coarse approximation of the solution at time T_{n+1} also with the initial solution U_n at time T_n. The coarse solver is to be chosen such that its cost is much lower than the one of the fine solver. A popular strategy consists in using the approximation method considered in the fine solver but with a larger time step [37]. Alternatively, one can use an approximation method with lower accuracy, or even use a different model from the original problem as long as it can give a reasonable coarse and fast approximation of the solution of the original problem [71].
The parareal algorithm aims at computing a sequence $(U_n^k)_{k,n}$ of approximations of $u(T_n)$ for $n \in \{0,\dots,N\}$ for every k in the following way. At the first step, the initial approximation U_n^0 at coarse time points $0 = T_0 < T_1 < \cdots < T_N = T$ can be computed sequentially using the coarse solver that reads

$$U_{n+1}^0 = \mathcal{G}(T_{n+1}, T_n, U_n^0),\ U_0^0 = u_0,$$

and then for $k = 0, 1, \dots$ with $U_0^{k+1} = u_0$, the parareal algorithm computes a more accurate approximation

$$U_{n+1}^{k+1} = \mathcal{G}(T_{n+1}, T_n, U_n^{k+1}) + \mathcal{F}(T_{n+1}, T_n, U_n^k) - \mathcal{G}(T_{n+1}, T_n, U_n^k).$$

In this iteration, the terms $\mathcal{F}(T_{n+1}, T_n, U_n^k)$ have the largest computational cost. Therefore, all these fine computations could be performed in parallel over each interval $[T_n, T_{n+1}]$, the main goal of parareal being to speed up the computation time. However, in order to achieve a real speed-up, the algorithm should converge in a number of iterations significantly smaller than the number of time intervals. Furthermore, one can also use a coarsened spatial mesh for the coarse solver, see [86], the parareal iteration becomes

$$U_{n+1}^{k+1} = \boldsymbol{I}\mathcal{G}(T_{n+1}, T_n, \boldsymbol{R}U_n^{k+1}) + \mathcal{F}(T_{n+1}, T_n, U_n^k) - \boldsymbol{I}\mathcal{G}(T_{n+1}, T_n, \boldsymbol{R}U_n^k),$$

where $\boldsymbol{R}$ and $\boldsymbol{I}$ denote interpolation and restriction operators between the two spatial meshes. For the sake of simplicity, linear interpolations are used for $\boldsymbol{I}$ while $\boldsymbol{R}$ is always a simple injection.

In the following discussion we give some numerical results of parareal algorithm from the literature. The figures were obtained by running the Matlab code from [31]. We first consider the Dahlquist problem which is a simple scalar linear ODE,

$$\frac{du}{dt} = au, \qquad u(0) = \mathrm{u}_0, \qquad t \in [0, T]. \tag{1.2}$$

Figure 1.1 shows the convergence behavior of parareal algorithm applied to Dahlquist problem and the theoretical linear and super linear error bounds, which can be found in [37]. We consider next the Lorenz equation coming from a very simplified model for

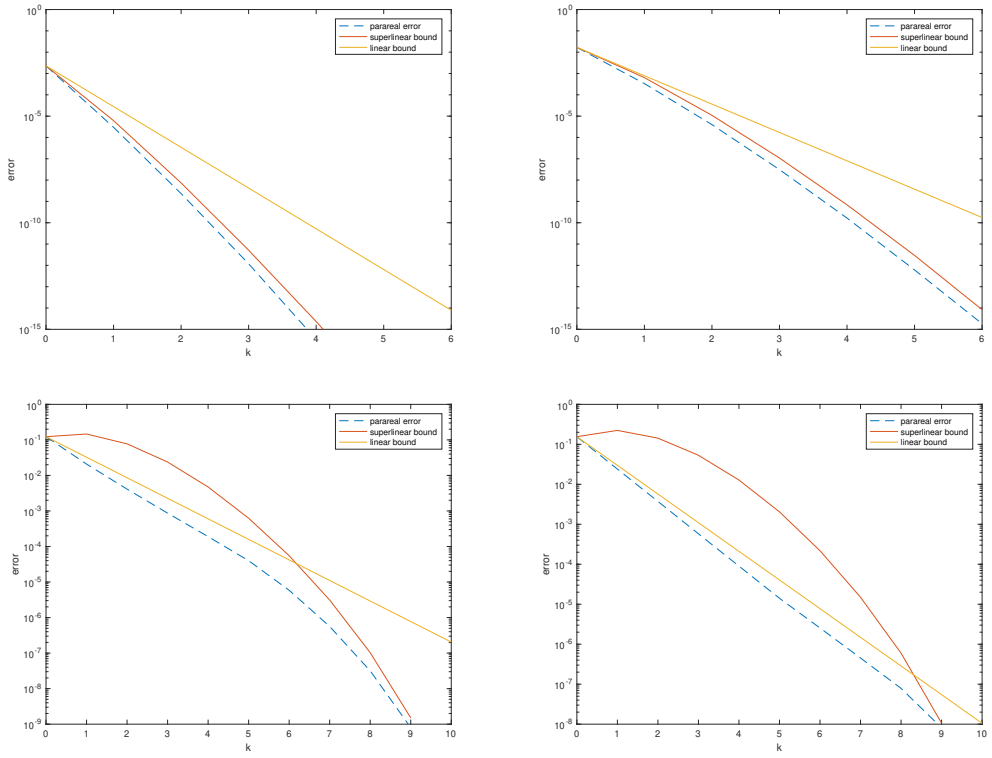

Figure 1.1: Error in maximum norm between approximate solution and fine sequential solution for Dahlquist problem $\frac{du}{dt} = au$ using backward Euler discretization with $a = -1$, $T = 0.25, 1, 10, 50$, $u_0 = 1$, $N = 10$.

weather prediction, including a system of 3 ODEs,

$$\begin{aligned} \dot{x} &= -\sigma x + \sigma y &&, x(0) = x_0, \\ \dot{y} &= -xz + rx - y &&, y(0) = y_0, \\ \dot{z} &= xy - bz &&, z(0) = z_0, \end{aligned} \tag{1.3}$$

with the parameters $\sigma, r, b \in \mathbb{R}$. In Figure 1.2 we show the approximate solution (on red dotted lines), the fine solution (solid blue line) and the error convergence after 20 iterations of parareal algorithm applied to the Lorenz equation (1.2). We give next some results on

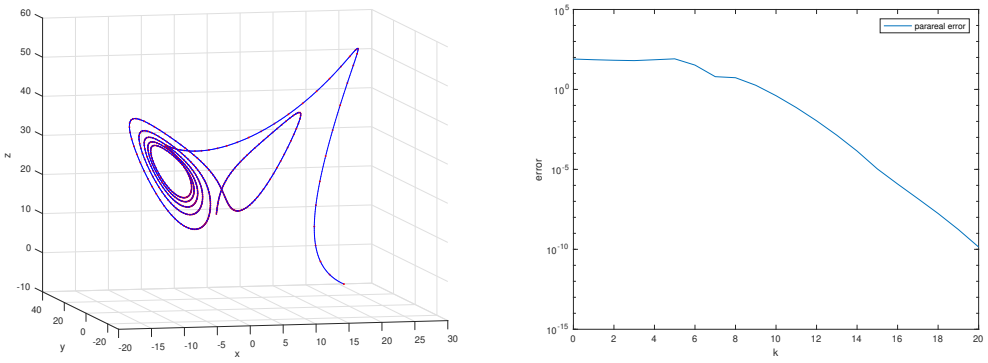

Figure 1.2: Solution (left) and error (right) in maximum norm between approximate solution and fine sequential solution for Lorenz equation (1.3) using forward Euler discretization with $\sigma = 10, r = 28, b = 8/3, T = 5, u_0 = [20; 5; -5], N = 500$.

PDEs, in particular the 1D heat equation which simulates the temperature in a nail as discussed in [31],

$$\begin{aligned} \partial_t u(x,t) &= \partial_{xx} u(x,t) && \text{in } (0,\pi) \times (0,T], \\ u(x,0) &= u_0(x) && \text{in } (0,\pi), \\ u(0,t) &= 0 && \text{in } (0,T], \\ u(\pi,t) &= 0 && \text{in } (0,T]. \end{aligned} \tag{1.4}$$

Figure 1.3 shows the solution and the error convergence after 16 iterations of parareal algorithm applied to the 1D heat equation (1.4). Finally, we give some results for the transport equation,

$$\begin{aligned} \partial_t u(x,t) + a\partial_x u(x,t) &= 0 && \text{in } \mathbb{R} \times (0,T], \\ u(x,0) &= u_0(x) && \text{in } \mathbb{R}, \end{aligned} \tag{1.5}$$

with the velocity $a \in \mathbb{R}$. We show in Figure 1.4 the solution and the error after 16 iterations of parareal algorithm applied to the transport equation (1.5). The convergence analysis and further discussion of parareal algorithm for PDEs can also be found in [37].

The main idea of parareal comes from domain decomposition methods in time, so it is natural to think about the equivalence between them. Indeed, by constructing a preconditioner matrix from the stationary iteration for the fine level and combine with domain decomposition strategy, we can show the equivalence between parareal and the two-level domain decomposition preconditioner in time. Additionally in the convergence we prove that the two-level domain decomposition preconditioner in time has the same error propagation with MGRIT with F-relaxation at the coarse time points, see [25, 30].

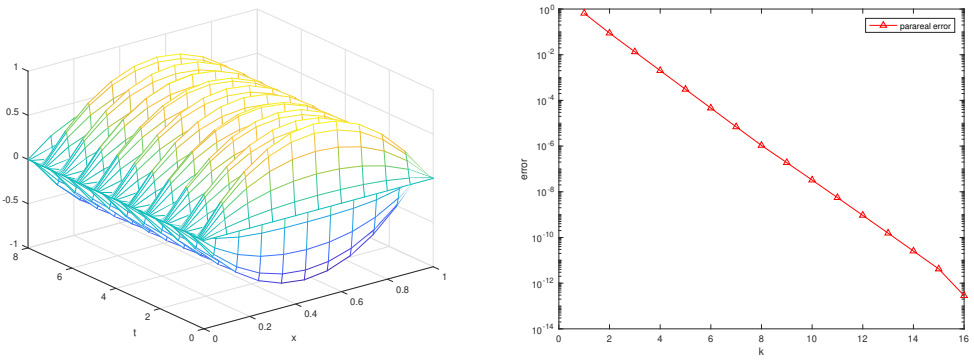

Figure 1.3: Parareal solution (left) and error (right) in maximum norm between approximate solution and fine sequential solution for 1D heat equation (1.4) using backward Euler discretization with $T = 8, u_0 = 0, N = 16, \delta t = 0.05, \Delta t = 0.5$.

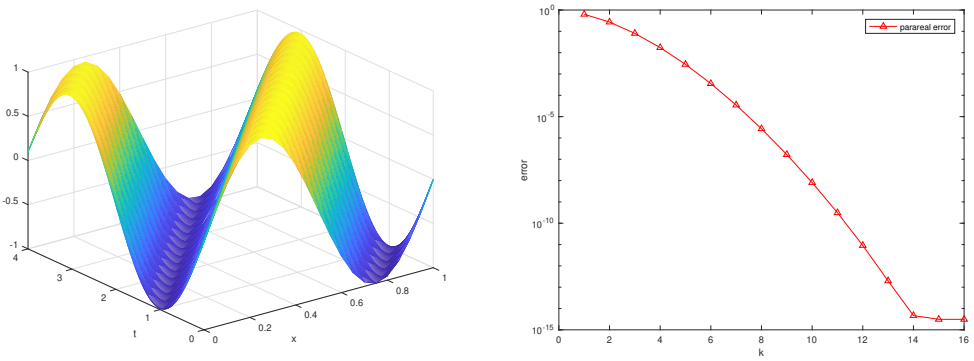

Figure 1.4: Parareal solution (left) and error (right) in maximum norm between approximate solution and fine sequential solution for transport equation (1.5) using backward Euler discretization with $a = 0.25, T = 4, u_0 = \sin(2\pi x), N = 16, \delta t = 0.0125, \Delta t = 0.25$.

This leads to the conclusion that the three algorithms, parareal, MGRIT and the two-level domain decomposition preconditioner in time are equivalent. This also confirms again the equivalence between parareal and MGRIT as already proven in [30]. Furthermore, using the idea of Krylov subspace methods to accelerate the parareal algorithm also gives promising results. By combining parareal with Krylov subspace methods, we can exploit the parallelism both in time and space. There are many techniques to improve the parallelism in space, exploiting the information of the Krylov subspace as well as building a coarse space generated from last iterations. Ideas of recycling a subspace that minimizes the loss of orthogonality with the Krylov subspace from the previous system can be found in [1, 78]. Recycled subspace is utilized by minimizing the residual over this subspace and then maintaining orthogonality with the image of this space in the Arnoldi recurrence. However in this work, we aim at building a deflation preconditioner which removes the

bad impact of the smallest singular values on the convergence of Krylov subspace methods, namely GMRES. In the next section we give a brief introduction about two-level domain decomposition preconditioners which lead to the idea of the two-level additive Schwarz in time preconditioner or parareal.

1.2 Two-level domain decomposition preconditioners

In parallel computing, domain decomposition [16, 21, 83] is one of the most interesting methods that can exploit the massive parallelism in spatial domain by decomposing the original problem into subproblems on subdomains. Each subdomain problem is iteratively solved in parallel then the information of solution is exchanged between subdomains by some transmission condition. For their potential parallel efficiency, domain decomposition methods have been well studied in various application fields, see [60, 84, 105]. They can be either applied directly to solve the PDEs, or used as preconditioners for Krylov subspace methods. In this work, we put our interest in the latter use and we first briefly recall its formulation. Consider a large sparse symmetric positive definite linear system which is obtained from the discretization of an elliptic problem by Finite Difference method or Finite Element method

$$Au = f. \tag{1.6}$$

The condition number of A in (1.6) can be large. To avoid that, a transformation can be applied to make (1.6) easier to solve with a much smaller condition number, e.g., instead of solving (1.6), we now solve

$$M^{-1}Au = M^{-1}f, \tag{1.7}$$

in which M is called the preconditioner of the matrix A such that $M^{-1}A$ has a smaller condition number than A.

The one-level domain decomposition such as additive and multiplicative Schwarz are well-known in the literature for domain decomposition in space, especially for elliptic problems, see [16]. Those Schwarz methods are used as preconditioners for Krylov subspace methods in which the original problem is decomposed into subproblems whose solution only gives an approximate correction to the global error. Consider solving on each subdomain approximately the linear system involving A_i, the accuracy is ensured in the preconditioner because we are still solving the original problem on the entire domain. However, the iterative linear system solver must have some mechanism for the global communication of informations at each iteration. Suppose that we have two overlapping subdomains $\{\Omega_1, \Omega_2\}$ with the corresponding restriction matrices R_1, R_2 and prolongation matrices R_1^T, R_2^T such that

$$A_1 = R_1 A R_1^T, \quad A_2 = R_2 A R_2^T$$

are principal submatrices of A.

Hence, to solve the linear system $Au = f$, we start with an initial guess u^0 and a sequence of iterates $u^0, u^1, \ldots$ are generated following

$$\begin{aligned} u^{k+\frac{1}{2}} &= u^k + R_1^T A_1^{-1} R_1 (f - Au^k), \\ u^{k+1} &= u^{k+\frac{1}{2}} + R_2^T A_2^{-1} R_2 (f - Au^{k+\frac{1}{2}}), \end{aligned}$$

we can combine the two steps and obtain

$$u^{k+1} = u^k + (R_1^T A_1^{-1} R_1 + R_2^T A_2^{-1} R_2 - R_2^T A_2^{-1} R_2 A R_1^T A_1^{-1} R_1)(f - Au^k).$$

As stated in [16], the iteration above corresponds to a generalization of the block Gauss-Seidel iteration with overlapping blocks and the two subdomain solvers A_1^{-1} and A_2^{-1} are required for each iteration. By defining

$$P_i \equiv R_i^T A_i^{-1} R_i A, \quad i = 1, 2,$$

this procedure is referred to as a multiplicative Schwarz iteration since its convergence is governed by the iteration matrix $(I - P_2)(I - P_1)$.

The additive Schwarz preconditioner can be analogously defined using the block Jacobi as

$$\begin{aligned} u^{k+\frac{1}{2}} &= u^k + R_1^T A_1^{-1} R_1 (f - Au^k), \\ u^{k+1} &= u^{k+\frac{1}{2}} + R_2^T A_2^{-1} R_2 (f - Au^k), \end{aligned}$$

and by eliminating $u^{k+\frac{1}{2}}$ we obtain

$$u^{k+1} = u^k + (R_1^T A_1^{-1} R_1 + R_2^T A_2^{-1} R_2)(f - Au^k).$$

Following this strategy, we can exploit the parallelism since the two subdomain solvers can be executed in parallel. This is also equivalent to a Richardson iteration with an additive Schwarz preconditioner

$$M^{-1} = R_1^T A_1^{-1} R_1 + R_2^T A_2^{-1} R_2.$$

With sufficient overlap, both multiplicative and additive Schwarz iterations can converge with a rate independent of the mesh size. However, their convergence rate deteriorates when the number of subdomains becomes large because of a lack of global information coupling the subdomains. In order to obtain a scalable domain decomposition algorithm which depends weakly on the number of subdomains, a coarse space can be used to couple global information of all subdomains. Thus, the two-level additive Schwarz preconditioner is defined as

$$M_{AS}^{-1} = \sum_{j=1}^{M} R_j^T A_j^{-1} R_j + R_0^T B^{-1} R_0,$$

which is equivalent to the iterates

$$u^{k+1} = u^k + (R_0^T B^{-1} R_0 + \sum_{j=1}^{M} R_j^T A_j^{-1} R_j)(f - Au^k),$$

and the two-level multiplicative Schwarz preconditioner is analogously defined as

$$M_{MS}^{-1} = \left[I - (I - R_0^T B^{-1} R_0 A) \left(I - \sum_{j=1}^{M} R_j^T A_j^{-1} R_j A \right) \right] A^{-1},$$

which is equivalent to the iterates

$$\begin{aligned} u^{k+\frac{1}{2}} &= u^k + \sum_{j=1}^{M} R_j^T A_j^{-1} R_j (f - A u^k), \\ u^{k+1} &= u^{k+\frac{1}{2}} + R_0^T B^{-1} R_0 (f - A u^{k+\frac{1}{2}}), \end{aligned}$$

where A_j for $j = 1, \ldots, M$ denotes the i^{th} subdomain matrix of A, M is the number of subdomains, B is the coarse problem matrix with the corresponding restriction matrix R_0 and prolongation matrix R_0^T.

This leads to the idea of two-level domain decomposition preconditioners in time in which each subdomain problem can be solved in parallel and then a coarse time correction can be performed sequentially to couple information between all temporal subdomains. This strategy results in a very similar framework with parareal and indeed, one can figure out that it is equivalent to parareal by Lemma 2.2.1 and Lemma 2.3.1 in Chapter 2. It will be seen that parareal is equivalent to using the preconditioned stationary iteration which computes a new approximate solution U_F^{k+1} from U_F^k,

$$U_F^{k+1} = U_F^k + M_{SC}^{-1}(f - A U_F^k),$$

where M_{SC}^{-1} is a two-level additive Schwarz in time preconditioner defined as,

$$M_{SC}^{-1} = (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \left(\sum_{i=1}^{\hat{N}} R_i^T A_i^{-1} R_i \right). \tag{1.8}$$

One can know from [25] that parareal is equivalent to MGRIT with F-relaxation. In this work, we show that the two-level domain decomposition preconditioner in time also has the same error propagation as MGRIT after one iteration with F-relaxation at coarse time points. For that reason, the next section is dedicated to the multigrid reduction in time or MGRIT method.

1.3 Multigrid reduction in time (MGRIT) method

The multigrid method is well studied in the literature [12, 52, 101, 103], especially its typical application to solving numerically elliptic problems [88]. The most interesting characteristic of multigrid methods is that they can give very accurate numerical solutions to very complicated non-symmetric and nonlinear systems of equations like the Navier-Stokes equations [95] in many domains and many boundary conditions. Similarly to domain decomposition methods, multigrid methods can be either used as direct solvers for PDEs, or used as preconditioners for Krylov subspace methods. Given that multigrid methods rely on combining the use of a coarse and a fine grid, those methods first reduce the high frequency errors by some relaxation methods, then the residual error of the fine grid problem is computed and transferred to the coarse grid by a restriction operator. The coarse grid problem is then solved to produce a correction between the coarse grid problem solution and the restriction of the fine grid problem solution. This correction is

transferred back to the fine grid by a prolongation operator. The final step is to correct the fine solution by adding the prolongated correction from the coarse grid problem. We represent those steps by considering the discretization of a linear PDE on a spatial domain by using the fine mesh to obtain the linear system

$$A_F u_F = f_F, \tag{1.9}$$

where A_F stands for the discretization of the problem , u_F and f_F stand for the approximate solution and the right hand side on the fine mesh. We consider additionally the error equation on the fine mesh

$$A_F e_F = r_F, \tag{1.10}$$

where $e_F = u_{ex,F} - u_F$ stands for the error between the exact solution and the approximate solution and $r_F = f_F - A_F u_F$ stands for the residual of (1.9). If e_F is known, then the solution u_F to (1.9) can be approximated by

$$u_F = u_F + e_F. \tag{1.11}$$

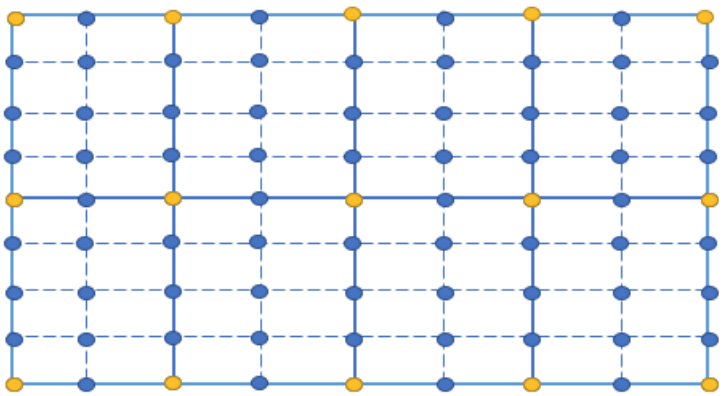

Figure 1.5: Two-level spatial mesh with the coarse level at yellow nodes and the fine level at all nodes.

The main idea of multigrid methods is to approximate the error e_F of the fine grid problem by the error from the coarse grid problem. If the error on the coarse grid is known, it can be linearly interpolated back to the fine grid and used as a correction. This coarse grid problem is cheaper, easier to solve and provides an acceptable approximate error e_C for e_F. Hence to obtain the error e_C we need to solve the error equation

$$A_C e_C = r_C, \tag{1.12}$$

where A_C stands for the coarse problem matrix, $e_C = u_{ex,C} - u_C$ stands for the error between the exact solution and the approximate solution and r_C stands for the residual of the coarse grid problem. The residual r_C of (1.12) is not known, however it can be approximated by

the residual r_F of (1.10) which is known. Denote by R^T the prolongation matrix of linear interpolation from the coarse grid to the fine grid and by R the restriction matrix from the fine grid to the coarse grid. A simple coarse grid correction can be derived by

$$u_F = u_F + R^T A_c^{-1} R(f_F - A_F u_F),$$

it means, we compute the residual on the fine grid, restrict it to the coarse grid then solve the coarse grid problem and prolongate the coarse grid correction to the fine grid. Multigrid methods then perform some post-smoothing step and repeat until the convergence is obtained following their strategies. Those are the main steps of multigrid methods for solving PDEs on spatial domains, in this work we only consider the use of multigrid methods for time dependent problems. Thus we briefly recall the multigrid reduction in time (MGRIT) method following [20]. Consider a time dependent problem in which we can compute approximately the numerical solution iteratively by following steps

$$\begin{aligned} u_0 &= g_0, \\ u_j &= \phi u_{j-1} + g_j, \quad j = 1, 2, \ldots, N_t, \end{aligned}$$

in which $u_j \in \mathbb{R}^{N_x}$, where N_x stands for the spatial dimension, ϕ stands for the one-step time discretization matrix. We can represent those steps in a linear system of equations

$$Au := \begin{bmatrix} \mathcal{I} & & & \\ -\phi & \mathcal{I} & & \\ & \ddots & \ddots & \\ & & -\phi & \mathcal{I} \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N_t} \end{bmatrix} = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{N_t} \end{bmatrix} =: g. \tag{1.13}$$

The system (1.13) is discretized on a temporal mesh with $t_j = j\delta t, \quad j = 0, 1, \ldots, N_t$ and the time step $\delta t = T/N_t$. Instead of sequentially solving this system, MGRIT exploits the parallelism by combining the sequential solve of the original time-stepping problem with a coarse grid approximation. Thus, a coarse grid is defined at $T_j = j\Delta t, \quad j = 0, 1, \ldots, N_T$ and the coarse time step $\Delta t = T/N_T = m\delta t$ where m is the coarsening factor and N_T is the number of coarse time points.

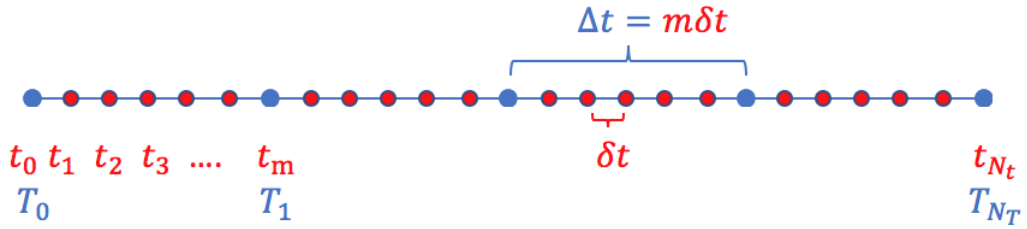

Figure 1.6: Two-level temporal mesh with the coarse level (C-points) at blue nodes and the fine level (F-points) at all nodes.

By eliminating all F-points values, a coarse grid problem can be obtained at all C-points

$$\begin{aligned} u_0 &= g_0, \\ u_{km} &= \phi^m u_{(k-1)m} + \sum_{i=0}^{m-1} \phi^i g_{km-i}, \end{aligned}$$

this coarse grid problem produces exactly the same solution as the system (1.13) at coarse time points. However it is not much easier to solve than the original problem. Thus, MGRIT reduces the cost of ϕ^m by an approximate operator ϕ_Δ and combines the use of both ϕ and ϕ_Δ in the following Algorithm 1, which can be found in [20].

Algorithm 1: MGRIT$(\phi, \phi_{\Delta t}, u, g)$

1 **repeat**
2 Relax the approximate solution using ϕ.
3 Compute the residual on the coarse grid with ϕ.
4 Solve the coarse-grid correction problem using $\phi_{\Delta t}$.
5 Correct the approximate solution at the C-points.
6 Update the solution at the F-points with ϕ.
7 **until** norm of residual is small enough.

Algorithm 1 presents the two-level MGRIT algorithm. One can obtain a multilevel algorithm by recursively applying the algorithm in step 4. There are several approaches that can be used to relax the approximate solution using ϕ in step 2. Basically they come from the F-relaxation and the C-relaxation in which F-relaxation propagates to obtain the approximate solution at fine time points based on the coarse time points, and C-relaxation propagates to obtain the approximate solution at coarse time points based on the previous fine time points. Additionally, one can combine both F-relaxation and C-relaxation to obtain the FCF-relaxation in which the solution is relaxing by first using the F-relaxation, then the C-relaxation and followed by one more F-relaxation. The FCF-relaxation produces more accurate approximate solution than just only using the F-relaxation in step 2. The error propagation as well as the convergence behavior of MGRIT are well studied in [20].

It is known from [37] that parareal can be interpreted as a two-level multigrid method. In this work, we present a new interpretation of parareal algorithm as a two-level Schwarz preconditioner which produces the same error propagation as MGRIT after one iteration with F-relaxation at coarse time points. Furthermore, we find that adding more coarse or fine propagation steps in the two-level preconditioner in time gives faster convergence. In particular the preconditioner produces the same error propagation as MGRIT with FCF-relaxation or F(CF)^2-relaxation. However, there is a trade-off to be considered in terms of computational costs and parallel implementation, which will be discussed in more detail in Chapter 2. Based on this interpretation, a variant that accelerates convergences by using a GMRES-type procedure is also presented.

1.4 Generalized minimal residual (GMRES) method

As stated in the previous section, parareal can be interpreted as a two-level Schwarz preconditioner. We consider the preconditioned system obtained from the discretization of a linear time dependent problem

$$M^{-1}\boldsymbol{A}u = M^{-1}f, \tag{1.14}$$

in which M stands for the two-level Schwarz preconditioner whose form is given in detail in Chapter 2, $\boldsymbol{A}$, u are the same as in the system (1.13) and f stands for the right-hand side. It is

obvious that the matrix A is not symmetric, thus if we want to accelerate the preconditioned system (1.14) by some Krylov subspace method, GMRES becomes a bright candidate since it is a Krylov method designed for non-symmetric linear systems. Introduced by Yousef Saad and Martin H. Schultz in 1986 [90], GMRES shows its effectiveness in solving linear sparse systems especially when combined with some preconditioners [4, 73, 77, 80]. Furthermore, parallelism conducted on massively parallel computer allows to effectively reduce the computation time as studied in ILU-preconditioned GMRES [102]. We briefly recall the GMRES algorithm together with the Krylov subspaces and the Arnoldi procedure by considering a linear sparse system obtained from the discretization of some PDEs using finite difference or finite elements methods

$$\boldsymbol{A}x = b, \tag{1.15}$$

where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is non-singular and diagonalizable, $b \in \mathbb{R}^n$ is the right-hand side and $x \in \mathbb{R}^n$ is the unknown vector. The Krylov subspace of dimension $k > 0$ associated to $\boldsymbol{A}$ and b is defined as

$$K_k(A, b) = span\{b, \boldsymbol{A}b, \ldots, \boldsymbol{A}^{k-1}b\}.$$

An orthogonalization procedure, namely Arnoldi is used to construct the basis vectors for the Krylov subspace and we obtain at each iteration k the relation

$$\boldsymbol{A}V_k = V_{k+1}H_k,$$

where $V_k = \{v_1, \ldots, v_k\}, V_{k+1} = [V_k, v_{k+1}]$, and $H_k = (h_{i,j})_{1\leq i\leq k+1, 1\leq j\leq k}$. GMRES approximates the exact solution of (1.15) by the vector $x_k \in K_k$ such that the Euclidean norm of the residual $r_k = \boldsymbol{A}x_k - b$ is minimized. Starting from an initial guess $x_0 \neq 0$, the vector $x_k \in K_k$ can be approximated by $x_k = x_0 + V_k y_k$, where $y_k \in \mathbb{R}^n$ is obtained from solving the linear least squares problem $\min_{y_k} ||r_k||_2 = \min_{y_k} ||H_k y_k - \beta e_1||_2$ with $\beta = b - \boldsymbol{A}x_0$, $e_1 = (1, 0, 0, \ldots, 0)^T$ and V_k is a matrix whose columns form an orthonormal basis $\{v_1, \ldots, v_k\}$ of K_k. Specifically, x_k is approximated as in Algorithm 2 which displays GMRES, where V_k and H_k come from the Arnoldi procedure in Algorithm 3.

GMRES algorithm repeats until the residual is less than some given tolerance. It will be seen that GMRES improves slightly the convergence of parareal and it allows to solve problems for which parareal has difficulty to converge , as in the case when the advection and reaction coefficients are large compared to the diffusion term for the advection-reaction-diffusion problem . However in general it does not improve drastically the convergence of parareal for our test problems, and this was also observed in previous works as [79] which studied the acceleration of waveform relaxation methods.

It has been studied in [78, 89, 99] that the convergence rate of Krylov methods is significantly affected by the spectrum of the coefficient matrix $\boldsymbol{A}$. The authors showed that the small eigenvalues which are close to zero make the convergence rate become slow. In the work of Simoncini [96] for unsymmetric matrices, small singular values also affect the convergence of Krylov methods, namely GMRES. As it can be seen in Example 1 from [96], the coefficient matrix A is obtained using centered finite differences method discretization for the equation,

$$(-e^{-xy}u_x)_x + (-e^{-xy}u_y)_y + 10(u_x + u_y) - 60u = f, \tag{1.16}$$

Algorithm 2: GMRES

Input: Coefficient matrix $\boldsymbol{A}$, right-hand side b, initial guess x_0, Krylov subspace dimension k, $(k+1) \times k$ Hessenberg matrix H_k of coefficients $(h_{i,j})_{1 \leq i \leq k+1, 1 \leq j \leq k}$ initialized to 0.

1 Compute $r_0 = b - \boldsymbol{A}x_0$, $\beta := ||r_0||_2$ and $v_1 := \beta^{-1} r_0$
2 $j := 1$
3 **while** $j < k$ **do**
4 Compute $w := \boldsymbol{A}v_j$
5 **for** $i = 1 : j$ **do**
6 $h_{i,j} := w^T v_i$
7 $w := w - h_{i,j} v_i$
8 $h_{j+1,j} := ||w||_2$
9 **if** $h_{j+1,j} \neq 0$ **then**
10 $v_{j+1} := h_{j+1,j}^{-1} w$
11 **else**
12 Set $m = j$ and go to line 14
13 $j := j + 1$
14 Compute y_k that minimizes $||\beta e_1 - H_k y_k||_2$ with $e_1 = (1, 0, 0, \ldots, 0)^T$
15 Update the initial guess $x_k = x_0 + V_k y_k$ where $V_k = \{v_1, \ldots, v_k\}$

Algorithm 3: Arnoldi$(\boldsymbol{A}, v_1, m)$

Input: Normal vector $v_1 \in \mathbb{R}^n$, coefficient matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, number of iterations m.
Output: Orthonormal basis vectors V_m, Hessenberg matrix $H_m \in \mathbb{R}^{(m+1) \times m}$.

1 **for** $j = 1 : m$ **do**
2 $w = \boldsymbol{A}v_j$
3 **for** $i = 1 : j$ **do**
4 $h_{i,j} = v_i^H w$
5 $w = w - \sum_{i=1}^{j} h_{i,j} v_i$
6 $h_{j+1,j} = ||w||_2$
7 **if** $h_{j+1,j} = 0$ **then**
8 set $m = j$, $v_{m+1} = 0$
9 **else**
10 $v_{j+1} = \frac{w}{h_{j+1,j}}$
11 $V_m = \{v_1, \ldots, v_m\}$, $V_{m+1} = [V_m, v_{m+1}]$, $H_m = (h_{i,j})_{i,j}$

with homogeneous Dirichlet boundary conditions on the unit square. The size of the coefficient matrix is $n = 100$, the 2 smallest singular values are $\sigma_{99} = 1.1982 \times 10^{-1}$ and $\sigma_{100} = 8.4646 \times 10^{-3}$. The convergence history of GMRES together with MinPert (restarted minimum perturbation method) and FOM (restarted full orthogonalization method) are shown with different choices of right-hand side and $m = 10$ (dimension of the Krylov

subspace). We can observe that GMRES almost stagnates while the two other methods converge. In Example 2 from [96], the coefficient matrix A_1 is obtained from A as, $A_1 := A - u_{99}11u_{99}^T$ and the two smallest singular values are $\sigma_{99} = 9.815\times10^{-3}$, $\sigma_{100} = 8.4646\times10^{-3}$. The right-hand side $b = \tilde{b}/||\tilde{b}||$ with $\tilde{b} = u_{99} - u_{100}$. It can be seen from this example that GMRES stagnates with $m = 15$ while FOM(15) and Minpert(15) converge. To avoid stagnation as well as to obtain fast convergence with robustness, a deflation subspace can be used to remove the impact of those small singular values in the spectrum. In this work we aim at building a deflation preconditioner which replaces the smallest singular values of $\boldsymbol{A}$ which are close to zero by much larger ones. For that purpose, we first introduce again the strong rank revealing QR (RRQR) from [19, 48] to show how the smallest singular values of $\boldsymbol{A}$ can be approximated and the tournament pivoting strategy which was proved in [19, 46] to be an efficient and highly parallel algorithm to build the space to be deflated in GMRES. The combination results in building a deflation precondtioner using Q_2 and R_{22} from (1.17) which removes the smallest singular values of $\boldsymbol{A}$ that are approximated by the singular values of R_{22} in the QR factorization of the permuted original matrix obtained from tournament pivoting strategy.

1.5 Strong Rank Revealing QR (strong RRQR) factorization

Strong rank revealing QR factorization is well-known in defining a basis for the approximate right null space of $\boldsymbol{A}$, especially in rank-deficient least-squares problems [44, 45]. We can also find its applications in subspace selection and linear dependency analysis as in [43, 62, 104] since its property allows to identify the linearly independent columns of $\boldsymbol{A}$. Furthermore, its interesting application can also be found in subspace tracking [11]. In this work we focus on approximating the smallest singular values of $\boldsymbol{A}$. For that purpose, we recall the definition of the strong rank revealing QR (RRQR) factorization following [48, Theorem 3.2] and [19, Theorem 2.4].

Theorem 1.5.1. *([48, Theorem 3.2] and [19, Theorem 2.4]) Let $\boldsymbol{A}$ be an $m \times n$ matrix and $1 \leq k \leq \min(m, n)$. Let $f > 1$ and Π be a permutation matrix such that the decomposition*

$$A\Pi = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}, \tag{1.17}$$

verifies for all $(i, j) \in [1, k] \times [1, n - k]$,

$$\gamma_j^2(R_{11}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq kf^2. \tag{1.18}$$

Then for any $1 \leq j \leq n - k$ and $1 \leq i \leq k$,

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})} \leq \sqrt{1 + kf^2(n-k)},\ 1 \leq \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} \leq \sqrt{1 + kf^2(n-k)}, \tag{1.19}$$

where $\Pi \in \mathbb{R}^{n\times n}, R_{11} \in \mathbb{R}^{k\times k}, R_{12} \in \mathbb{R}^{k\times(n-k)}, R_{22} \in \mathbb{R}^{(m-k)\times(n-k)}, Q_1 \in \mathbb{R}^{m\times k}$ and $Q_2 \in \mathbb{R}^{m\times(n-k)}, \gamma_j(R_{22})$ is the 2-norm of the jth column of R_{22}, $\sigma_i(A), 1 \leq i \leq \min(m, n)$ denotes

the ith singular value of $\boldsymbol{A}$ and $\sigma_{\min}(R_{11})$ is the smallest singular value of R_{11}. We note in (1.19) that the lower bounds always hold for any permutation Π thanks to the interlacing property of singular values.

We can see from (1.19) that the QR factorization in Theorem 1.5.1 reveals the rank of $\boldsymbol{A}$ by considering the singular values of R_{11} as approximations of the k largest singular values of $\boldsymbol{A}$, and the singular values of R_{22} as approximations of the $\min(m,n)-k$ smallest singular values of $\boldsymbol{A}$. This is called strong rank revealing factorization as the upper bound of (1.19) is expressed likely by a low degree polynomial in n. The factorization (1.17) was presented in [42], such QR factorization can be derived using QR factorization with column pivoting (QRCP). We present here Algorithm 4 and Algorithm 5 following [48] to compute a permutation matrix Π and a QR factorization (1.17) that satisfies (1.18) and (1.19).

Algorithm 4: QR with column pivoting

Input: Coefficient matrix $\boldsymbol{A}$, tolerance $\delta > 0$.

$k := 0,\ R := \boldsymbol{A},\ \Pi := I$

while $\max_{1\le j\le n-k}(\gamma_j(R_{22})) \ge \delta$ **do**

  $j_{\max} := argmax_{1\le j\le n-k}(\gamma_j(R_{22}))$

  $k := k+1$

  Compute $R := \mathcal{R}$ where $\mathcal{R}$ is the triangular matrix obtained from $QR(R\Pi_{k,k+j_{\max}-1})$ and $\Pi := \Pi\,\Pi_{k,k+j_{\max}-1}$

Algorithm 5: Strong RRQR factorization

Input: Coefficient matrix $\boldsymbol{A}$, tolerance $\delta > 0,\ k > 0,\ f \ge 1$.

$R := \mathcal{R}$, where $\mathcal{R}$ is the triangular matrix obtained from $QR(\boldsymbol{A}),\ \Pi := I$

while *there exist i and j such that* $\det(\bar{R}_{11})/\det(R_{11}) > f$, *where* $R = \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}$ *and* $\bar{\mathcal{R}} = \begin{bmatrix} \bar{R}_{11} & \bar{R}_{12} \\ & \bar{R}_{22} \end{bmatrix}$, *where* $\bar{\mathcal{R}}$ *is the triangular matrix obtained from* $QR(R\Pi_{i,j+k})$ **do**

  Find such an i and j

  Compute $R := \bar{\mathcal{R}}$ and $\Pi := \Pi\,\Pi_{i,j+k}$

As mentioned previously, in this work we study the approximations of the smallest singular values of $\boldsymbol{A}$, which are the singular values of R_{22}. For that reason, 1.5.1 is applied differently, which will be given in more detail in Chapter 4 to adapt that purpose. We note that the results and the proof are similar to the original version in 1.5.1. In the next section we give a brief introduction about tournament pivoting strategy, which is very effective in communication avoiding algorithms [19, 46].

1.6 Tournament pivoting strategy

Tournament pivoting idea was introduced in [19, 46] as a communication avoiding technique for LU and strong RRQR factorizations. The goal of the technique is to select

k columns from the input matrix $\boldsymbol{A}$. We present again the tournament pivoting strategy following [10]. Suppose that the matrix $\boldsymbol{A}$ can be divided into 4 blocks of columns, $\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \end{bmatrix}$. In the first step, strong RRQR is performed to select k columns from each column block $\boldsymbol{A}_{1i},\ 1 \leq i \leq 4$, the set of indices of selected columns corresponding to each column block $\boldsymbol{A}_{1i}$ is denoted by $I_{i0},\ 1 \leq i \leq 4$. In the next step, we put the sets of selected columns from the previous step two by two together following a reduction binary tree then perform strong RRQR to select k columns from each pair. For example in this case for the second step we perform strong RRQR to select k columns from $\boldsymbol{A}(:, I_{10} \cup I_{20})$ and k columns from $\boldsymbol{A}(:, I_{30} \cup I_{40})$, where $\boldsymbol{A}(:, I)$ denotes the submatrix whose columns come from the indices I of $\boldsymbol{A}$. Those sets of selected columns from the second step are denoted by $I_{i1},\ 1 \leq i \leq 2$. We then put the $2k$ selected columns from the second step together as $\boldsymbol{A}(:, I_{11} \cup I_{21})$ and perform strong RRQR to select the final set of k columns which is denoted by I_{12}. However in this work we do not use a reduction binary tree, the selected columns from the first step are concatenated and then strong RRQR is performed to select the final set of k columns. All steps of the tournament pivoting technique to select k columns from the columns of matrix $\boldsymbol{A}$ are presented in the following Algorithm 6.

Algorithm 6: Tournament pivoting for 1D column partitioned matrices

Input: $\boldsymbol{A}_1, \cdots, \boldsymbol{A}_p$ submatrices of rank k approximation

1 Perform strong RRQR to select k columns from each $\boldsymbol{A}_i$, indices of selected columns are denoted by $I_i, 1 \leq i \leq p$

2 Selected columns are concatenated in $\tilde{\boldsymbol{A}} = [\boldsymbol{A}_1(:, I_1) \cdots \boldsymbol{A}_p(:, I_p)]$

3 Perform again strong RRQR of $\tilde{\boldsymbol{A}}$ to select k columns

Output: indices of k rank revealing columns of $\boldsymbol{A}$

It was shown in [46] that

$$\gamma_j^2(R_{11}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq F_{TP}^2, \tag{1.20}$$

and the singular values of $\boldsymbol{A}$ can be approximated by the singular values of R_{11}, R_{22} in the sense that for any $1 \leq j \leq n-k$ and $1 \leq i \leq k$,

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})} \leq \sqrt{1 + F_{TP}^2(n-k)},\ 1 \leq \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} \leq \sqrt{1 + F_{TP}^2(n-k)}, \tag{1.21}$$

$$||R_{11}^{-1}R_{12}||_{\max} \leq F_{TP}, \tag{1.22}$$

where F_{TP} depends on k, f, n, the shape of reduction tree for tournament pivoting, and the number of iterations of CARRQR. For a binary reduction tree of depth $\log_2(n/k)$,

$$F_{TP} \leq \frac{1}{\sqrt{2k}}(n/k)^{\log_2(\sqrt{2}fk)}. \tag{1.23}$$

Although tournament pivoting was introduced to approximate the largest singular values and associated singular vectors for example see [10, 19, 46], in this work the algorithm is modified to approximate the smallest singular values and associated singular vectors of $\boldsymbol{A}$, which will be described in more detail in Chapter 4.

In the following section we give a short introduction of the deflation based on eigenvectors associated to the smallest eigenvalues leading to the idea of the deflation of singular vectors associated to the smallest singular values.

1.7 Deflation technique

We introduce the deflation of eigenvalues and eigenvectors strategies following [1]. Denote by S an $\boldsymbol{A}$-invariant subspace of dimension $k > 0$ associated to the smallest eigenvalues of the matrix $\boldsymbol{A}$. Denote by $Z \in \mathbb{R}^{n\times k}$ a matrix whose columns form an orthonormal basis of the subspace S and T the projection of $\boldsymbol{A}$ on the subspace S, i.e., $T = Z^H \boldsymbol{A} Z$. The following relation holds

$$\boldsymbol{A}Z = ZT.$$

Let $\{\lambda_1, \ldots, \lambda_n\}$ be the eigenvalues of $\boldsymbol{A}$ such that $|\lambda_1| \leq \cdots \leq |\lambda_n|$. Consider the preconditioned system

$$(I + Z(T^{-1} - I)Z^H)\boldsymbol{A}x = (I + Z(T^{-1} - I)Z^H)b. \tag{1.24}$$

By adding a low-rank correction, the following [24, Theorem 1] describes how to deflate the eigenvalues of the matrix $\boldsymbol{A}$.

Theorem 1.7.1. *([24, Theorem 1]) The eigenvalues of the matrix $\boldsymbol{A} + Z(T^{-1} - I)Z^H\boldsymbol{A}$ are $\{1, \ldots, 1, \lambda_{k+1}, \ldots, \lambda_n\}$. The equation* (1.24) *can be rewritten as*

$$((\boldsymbol{A} - ZZ^H\boldsymbol{A}) + ZT^{-1}Z^H\boldsymbol{A})x = ((b - ZZ^Hb) + ZT^{-1}Z^Hb). \tag{1.25}$$

Hence, to solve (1.15), we approximate the solution in two subspaces, one is inside S, the other is outside S. The first subspace approximates the part of the solution generated by the basis vectors V by solving the deflation subspace problem with the matrix T. The second subspace approximates the part that is orthogonal to S using the Krylov method. In the next discussion, we introduces the deflation of singular vectors based on [1, Theorem 2].

Theorem 1.7.2. *[1, Theorem 2] Let x_* be the exact solution of $\boldsymbol{A}x = b$. Let $\boldsymbol{A} = U\Sigma V^H$ be the singular value decomposition of $\boldsymbol{A}$ such that the singular values are ordered increasingly. Let k_τ be the number of singular values smaller than a given threshold $\tau > 0$. Consider the splitting of the SVD decomposition*

$$\boldsymbol{A} = U_1\Sigma_1V_1^H + U_2\Sigma_2V_2^H,$$

where $\Sigma_2 \in \mathbb{R}^{k_\tau\times k_\tau}$ is a diagonal matrix whose diagonal elements are the singular values smaller than τ. Consider $\tilde{x}$ an approximate solution of the following linear system of equations

$$(I - U_2U_2^H)\boldsymbol{A}x = (I - U_2U_2^H)b, \tag{1.26}$$

such that $||\hat{x} - \tilde{x}||_2 \leq \epsilon$, where $\hat{x}$ is an exact solution of 1.26 and $\epsilon > 0$. Then, the following holds

$$||x_* - (I - V_2V_2^H)\tilde{x} - V_2\Sigma_2^{-1}U_2^Hb||_2 \leq \epsilon.$$

The approximate solution of $\boldsymbol{A}x = b$ then can be recovered as

$$x = (I - V_2V_2^H)\tilde{x} + V_2x_2, \tag{1.27}$$

where $x_2 = \Sigma_2^{-1} U_2^H b$ and $\tilde{x}$ is the approximation solution of (1.26) given by the Krylov method.

Based on the deflation of singular vectors introduced above, we derive a new deflation strategy using strong Rank Revealing QR (RRQR) factorization [48]. The matrix $\boldsymbol{A}$ is first partitioned by using nested dissection on $\boldsymbol{A}^T\boldsymbol{A}$ and we apply sparse QR, especially strong RRQR with tournament pivoting to obtain a matrix with the last k columns corresponding to the k smallest singular values. From the QR factorization of that permuted matrix, Q_2 and $\tilde{V}_2$ are used for the deflation of singular vectors, as it can be seen in the following Theorem 1.7.3, which will be discussed in more detail in Chapter 4.

Theorem 1.7.3. *Let x_* be the exact solution of $\boldsymbol{A}x = b$. Consider the QR factorization of $\boldsymbol{A}$ with the last k columns corresponding to the k smallest singular values,*

$$\boldsymbol{A}\Pi = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}. \tag{1.28}$$

where Π is such that for all $(i, j) \in [1, n-k] \times [1, k]$,

$$\gamma_j^2(R_{11}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq (n-k)f^2, \tag{1.29}$$

If $||R_{22}||_2$ is small, then

$$\tilde{V}_2 = \Pi \begin{bmatrix} -R_{11}^{-1}R_{12} \\ I_k \end{bmatrix}, \tag{1.30}$$

is an approximate right null space of $\boldsymbol{A}$ where $\tilde{V}_2 \in \mathbb{R}^{n\times k}, \Pi \in \mathbb{R}^{n\times n}, R_{11} \in \mathbb{R}^{(n-k)\times(n-k)}, R_{12} \in \mathbb{R}^{(n-k)\times k}, R_{22} \in \mathbb{R}^{k\times k}, Q_1 \in \mathbb{R}^{n\times(n-k)}, Q_2 \in \mathbb{R}^{n\times k}$ and I_n, I_k denote the identity matrices of order n, k respectively.

Consider $\tilde{x}$ an approximate solution of the following linear system of equations

$$(I_n - Q_2Q_2^H)\boldsymbol{A}x = (I_n - Q_2Q_2^H)b, \tag{1.31}$$

such that $||\hat{x} - \tilde{x}||_2 \leq \epsilon$, where $\hat{x}$ is an exact solution of (4.43) and $\epsilon > 0$. Then, the following holds

$$||x_* - (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^H)\tilde{x} - \tilde{V}_2 R_{22}^{-1} Q_2^H b||_2 \leq \epsilon. \tag{1.32}$$

Given that the deflation based on strong RRQR is not as good as the deflation based on SVD in the sense that it only deflates the approximations of the smallest singular values of $\boldsymbol{A}$, in compensation we gain in terms of computation cost by exploiting parallel computing through tournament pivoting strategy.

In order to have better performance in deflated GMRES as well as the convenience in combination with other preconditioners later, Q_2 and R_{22} in the QR factorization can be used to define a deflation preconditioner which allows to replace the smallest singular values which are close to zero by much larger ones following [48, Theorem 3.2] , so that GMRES is not affected by those smallest singular values and converges faster. In particular, we build a deflation preconditioner from Q_2, R_{22} as,

$$M^{-1} = (I_n - Q_2Q_2^T) + Q_2R_{22}^{-1}Q_2^T. \tag{1.33}$$

Theorem 1.7.4. *[48, Theorem 3.2] Let* $R \equiv \begin{pmatrix} A_k & B_k \\ & C_k \end{pmatrix} = \mathcal{R}_k(M\Pi)$ *satisfy* $\rho(R,k) \leq f$. *Then*

$$\sigma_i(A_k) \geq \frac{\sigma_i(M)}{\sqrt{1+f^2k(n-k)}}, \qquad 1 \leq i \leq k,$$

and

$$\sigma_j(C_k) \leq \sigma_{j+1}(M)\sqrt{1+f^2k(n-k)}, \qquad 1 \leq j \leq n-k.$$

In the context of this thesis, $A_k \equiv R_{11}, B_k \equiv R_{12}$ and $C_k \equiv R_{22}, M$ stands for $\boldsymbol{A}, \Pi$ is the permutation matrix from strong RRQR with tournament pivoting strategy and $\rho(R,k) = \max_{1\leq i\leq n-k, 1\leq j\leq k} \sqrt{(R_{11}^{-1}R_{12})_{ij}^2 + (\gamma_j(R_{22})/\omega_i(R_{11}))^2}$, with $1/\omega_i(R_{11})$ denotes the 2-norm of the ith row of R_{11}^{-1}.

Furthermore, we also combine the deflation preconditioner with the block Jacobi preconditioner to obtain better convergence rate of GMRES method. Apart from the Krylov subspace acceleration of parareal, we give next the background on two-scale asymptotic expansion, which provides a very good candidate for the coarse propagator in the context of parareal algorithm.

1.8 Two-scale asymptotic expansion

As mentioned in section 1.1, there are various choices for the coarse propagator in the framework of parareal algorithm. In this work, we consider the idea of using a different model so-called the reduced model for the coarse propagator. This is not a new idea, there are similar approaches in the literature, e.g., Maday used a reduced model of stiff molecular kinetic reactions for the coarse solver in [71]. Haut and Wingate used a classical averaged model for the coarse solver to solve PDEs with linear high oscillating term in [53]. Ariel, Kim and Tsai used a multi-scale method solving the slow evolution for the coarse solver in [3]. In our work, we obtained the reduced model by using a two-scale asymptotic expansion and this reduced model is proved to provide an accurate approximation of the original equation when the small parameter ε vanishes [28]. The first advantage of using such reduced models is that, they are not stiff ODEs and can be computed with low computational cost. However, they contain information about the high oscillations and capture well the phase of the solution for the coarse solver. These are very important characteristics in solving stiff system of equations and thus they become very good choices for the coarse propagator in parareal algorithm. In the following discussion, we present the principles and the main result of two-scale asymptotic expansion leading to the reduced models. For that purpose, we consider the general singularly perturbed dynamical system

$$\frac{\mathrm{d}\mathcal{X}_\varepsilon}{\mathrm{d}t} = \mathbf{a}(t,\mathcal{X}_\varepsilon) + \frac{1}{\varepsilon}\mathbf{b}(t,\mathcal{X}_\varepsilon), \qquad \mathcal{X}_\varepsilon(s) = \mathcal{X}, \tag{1.34}$$

where $\mathcal{X}_\varepsilon : \mathbb{R} \to \mathbb{R}^d$ and $\mathbf{a}$ and $\mathbf{b}$ are given fields satisfying suitable assumptions and s is the initial time. We recall from [28] the asymptotic two-scale expansion method to

approximate the solution $\mathcal{X}_\varepsilon(t)$ when $\varepsilon \to 0$. Under regularity assumptions on $\mathbf{a}$ and $\mathbf{b}$ and assuming the solution $\mathbf{Z}(t;\theta,\mathbf{z})$ to equation

$$\frac{\mathrm{d}\mathbf{Z}}{\mathrm{d}\theta} = \mathbf{b}(t,\mathbf{Z}), \qquad \mathbf{Z}(t;0,\mathbf{z}) = \mathbf{z} \tag{1.35}$$

to be periodic in θ, for every $t \in \mathbb{R}$ and every $\mathbf{z} \in \mathbb{R}^d$, it is proved in [28] that $\mathcal{X}_\varepsilon$ can be approximated by the following two-scale expansion in time

$$\mathcal{X}_\varepsilon(t) = \mathcal{X}^0\Big(t, \frac{t-s}{\varepsilon}\Big) + \varepsilon\,\mathcal{X}^1\Big(t, \frac{t-s}{\varepsilon}\Big) + \varepsilon^2\,\mathcal{X}^2\Big(t, \frac{t-s}{\varepsilon}\Big) + \ldots, \tag{1.36}$$

when $\varepsilon \to 0$ and where the functions $\mathcal{X}^i(t,\theta)$ are periodic in θ for every $i \in \mathbb{N}$. On the other hand, strong convergence theorems are proved, giving that for the zero-th order two-scale model solution we have

$$\mathcal{X}_\varepsilon(t) \sim \mathcal{X}^0\Big(t, \frac{t-s}{\varepsilon}\Big), \quad \text{when } \varepsilon \to 0,$$

and for the first order two-scale model solution,

$$\mathcal{X}_\varepsilon(t) \sim \mathcal{X}^0\Big(t, \frac{t-s}{\varepsilon}\Big) + \varepsilon\,\mathcal{X}^1\Big(t, \frac{t-s}{\varepsilon}\Big), \quad \text{when } \varepsilon \to 0.$$

The following [28, Theorem 1.1] shows the convergence result for the two-scale limit model or the zero-th order approximation in the case of a six dimensional space ($d = 6$).

Theorem 1.8.1. *[28, Theorem 1.1] We assume that*[1] $\mathbf{a} \in \big(C_b^1(\mathbb{R}\times\mathbb{R}^6)\big)^6$ *and* $\mathbf{b} \in \big(C_b^2(\mathbb{R}\times\mathbb{R}^6)\big)^6$. *Assume also that the solution of* (1.35) *is* 2π*-periodic in* θ*, for every* $t \in \mathbb{R}$ *and every* $\mathbf{z} \in \mathbb{R}^6$. *Then, for every initial condition* $\mathcal{X} \in \mathbb{R}^6$*, every* $\varepsilon > 0$*, and every* $\Delta S > 0$*, the solution* $\mathcal{X}_\varepsilon$ *of* (1.34) *exists on* $[s, s+\Delta S]$*, is unique and satisfies*

$$\lim_{\varepsilon\to 0} \sup_{t\in[s,s+\Delta S]} \Big|\mathcal{X}_\varepsilon(t) - \mathcal{X}^0\Big(t, \frac{t-s}{\varepsilon}\Big)\Big| = 0, \tag{1.37}$$

where $|\cdot|$ *stands for the Euclidean norm on* $\mathbb{R}^6$ *and* $\mathcal{X}^0$ *satisfies*

$$\mathcal{X}^0(t,\theta) = \mathbf{Z}\big(t;\theta,\mathcal{Y}^0(t)\big) \tag{1.38}$$

and where $\mathcal{Y}^0$ *is the solution to*

$$\frac{\mathrm{d}\mathcal{Y}^0}{\mathrm{d}t} = \alpha(t,\mathcal{Y}^0), \qquad \mathcal{Y}^0(s) = \mathcal{X}, \tag{1.39}$$

with α *defined by*

$$\alpha(t,\mathcal{Y}) = \frac{1}{2\pi}\int_0^{2\pi} \{\nabla\mathbf{Z}(t;\theta,\mathcal{Y})\}^{-1}\Big\{\mathbf{a}(t,\mathbf{Z}(t;\theta,\mathcal{Y})) - \frac{\partial\mathbf{Z}}{\partial t}(t;\theta,\mathcal{Y})\Big\}\,d\theta.$$

Hence, it can be seen that (1.39) is just a very simple ODE, we can easily compute $\mathcal{Y}^0$ from (1.39), then we obtain $\mathcal{X}^0$ by (1.38) as a zero-th order two-scale model solution.

The efficiency of coupling parareal with reduced model based on the two-scale asymptotic expansion for the coarse propagator will be seen in Chapter 3, in which the convergence is obtained uniformly for various test cases of charged particle simulations.

[1] C_b^m stands for the space of continuous functions with bounded derivatives to the order m.

2 Interpretation of Parareal as a Two-level Additive Schwarz In Time Preconditioner and Its Acceleration with GMRES

Contents

Abstract

We describe an interpretation of parareal as a two-level additive Schwarz preconditioner in the time domain. We show that this two-level preconditioner in time is equivalent to parareal and to multigrid reduction in time (MGRIT) with F-relaxation. We also discuss the case when additional fine or coarse propagation steps are applied in the preconditioner. This leads to procedures equivalent to MGRIT with FCF-relaxation and to MGRIT with $F(CF)^2$-relaxation or overlapping parareal. Numerical results show that these variants have faster convergence in some cases. In addition, we also apply a Krylov subspace method, namely GMRES (Generalized Minimal Residual), to accelerate the parareal algorithm. Better convergence is obtained, especially for the advection-reaction-diffusion equation in the case when advection and reaction coefficients are large.

Keywords: Parareal, Two-level additive Schwarz in time preconditioner, MGRIT with F-relaxation, FCF-relaxation, $F(CF)^2$-relaxation, GMRES.

2.1 Introduction

In this chapter we focus on parareal, an algorithm introduced by J.L. Lions et al. [68] in 2001, which allows to exploit parallelism in time for initial value problems. Over the last two decades, this algorithm has been studied for a range of applications, going from molecular dynamics simulations [5], unsteady hydrodynamic simulations [23], kinetic neutron diffusion equation [8, 9], the Korteveg-deVries-Burgers' equations [61], Hamiltonian systems [17, 34], to financial mathematics as the Black-Scholes equations [7, 72, 81]. Its stability and convergence are studied in a series of papers, e.g. [6, 33, 37, 97].

Given a time dependent problem, parareal allows parallel in time integration by relying on a combination between a fine propagator, which gives a very accurate approximate of the solution, and a coarse propagator, which is less expensive and gives a coarse approximate of the solution. For this, the time domain is decomposed into a number of uniform time subdomains. From an initial solution obtained by sequentially using the coarse propagator, parareal iteratively corrects it by the difference between the fine solution obtained in parallel using the fine propagator and the coarse solution obtained from the previous iteration.

Several different interpretations of parareal exist in the literature. A derivation of the parareal algorithm as a multiple shooting method is given in [37]. An investigation of the usage of spectral deferred corrections in the framework of parareal is given in [75, 76]. Coupling parareal in time with Schwarz waveform relaxation methods [32, 35] to exploit parallelism in both time and space are promising directions of research as well. Parareal can also be interpreted as a multigrid method in time, referred to as MGRIT with F-relaxation [25, 30]. Following this interpretation, several different variants have been

investigated, as MGRIT with FCF-relaxation, MGRIT with F(CF)^2-relaxation, e.g. [25, 58], where F refers to the F-relaxation and C refers to the C-relaxation.

Given that parareal relies on a decomposition of the time domain into subdomains, in this work we study the connection between parareal and domain decomposition methods. Traditionally domain decomposition methods are used for solving a linear system of equations $\tilde{A}\tilde{u} = \tilde{f}$, $\tilde{A} \in \mathbb{R}^{n\times n}$, arising from the discretization of a PDE by using for example the finite element method, and they rely on a decomposition of the space domain into subdomains. We consider here the case in which this linear system is solved by using an iterative method as a Krylov subspace method, preconditioned by $\tilde{M}^{-1}$,

$$\tilde{M}^{-1}\tilde{A}\tilde{u} = \tilde{M}^{-1}\tilde{f},$$

where $\tilde{M}^{-1}$ is a domain decomposition method. One-level domain decomposition preconditioners such as additive and multiplicative Schwarz preconditioners are well-known in the literature for domain decomposition in space, see e.g. [16]. However, their convergence rate deteriorates when the number of subdomains becomes large because of a lack of global information coupling the subdomains. In order to obtain a scalable domain decomposition algorithm which depends weakly on the number of subdomains, a coarse space can be used to couple global information of all subdomains. This leads to the idea of two-level domain decomposition preconditioners. Given a spatial decomposition of the degrees of freedom of $\tilde{A}$ into $\tilde{N}$ subdomains, the restriction of $\tilde{A}$ to a spatial subdomain i, for $i = 1, \ldots, \tilde{N}$, is referred to as $\tilde{A}_i$ and is obtained by defining a restriction matrix $\tilde{R}_i$ together with a prolongation matrix $\tilde{R}_i^T$, such that $\tilde{A}_i = \tilde{R}_i\tilde{A}\tilde{R}_i^T$. By defining the coarse matrix $\tilde{A}_0$ and corresponding restriction and prolongation matrices $\tilde{R}_0, \tilde{R}_0^T$, the two-level additive Schwarz preconditioner is defined as,

$$\tilde{M}_{AS2}^{-1} = \tilde{R}_0^T\tilde{A}_0^{-1}\tilde{R}_0 + \sum_{i=1}^{\tilde{N}} \tilde{R}_i^T\tilde{A}_i^{-1}\tilde{R}_i,$$

and the two-level multiplicative Schwarz preconditioner is analogously defined as

$$\tilde{M}_{MS2}^{-1} = \left[\mathcal{I} - (\mathcal{I} - \tilde{R}_0^T\tilde{A}_0^{-1}\tilde{R}_0\tilde{A}) \prod_{i=1}^{\tilde{N}} (\mathcal{I} - \tilde{R}_i^T\tilde{A}_i^{-1}\tilde{R}_i\tilde{A})\right]\tilde{A}^{-1}.$$

To show the equivalence between parareal and two-level domain decomposition methods, we consider the linear time dependent problem,

$$\frac{du}{dt} = f(u),\ u(0) = \mathrm{u}_0,\ \ u(t) \in \mathbb{R}^d, t \ \text{ in } (0, T), \tag{2.1}$$

and an algebraic framework in which the solution to (2.1) can be obtained by solving with a residual correction scheme the linear system of equations,

$$AU_F = f, \tag{2.2}$$

where the time domain $(0, T)$ was decomposed into N time subdomains, $A \in \mathbb{R}^{(N+1)d\times(N+1)d}$ is bidiagonal and denotes the time-stepping coefficient matrix with the form,

$$A := \begin{bmatrix} \mathcal{I} & & & \\ -\phi & \mathcal{I} & & \\ & \ddots & \ddots & \\ & & -\phi & \mathcal{I} \end{bmatrix}.$$

In this equation $\mathcal{I} \in \mathbb{R}^{d\times d}$ is the identity matrix, $\phi \in \mathbb{R}^{d\times d}$ denotes an arbitrary stable discretization method in space and time, $U_F := [u_0, \ldots, u_N]^T$ denotes the solution at fine time steps and $f := [\mathrm{u}_0, 0 \ldots, 0]^T$ is the right-hand side. The matrix A includes all the time steps for the whole time domain. If N and d become large, (2.2) results in a very large and sparse system. This is the case where domain decomposition type methods show their advantages. We consider the problem on a uniform grid, the time steps and space steps do not change from one to the next so the discretization matrix for each time step, namely ϕ, does not change . We show that parareal is equivalent to using the preconditioned stationary iteration which computes a new approximate solution U_F^{k+1} from U_F^k,

$$U_F^{k+1} = U_F^k + M_{SC}^{-1}(f - AU_F^k),$$

where M_{SC}^{-1} is a two-level additive Schwarz in time preconditioner defined as,

$$M_{SC}^{-1} = (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0)\left(\sum_{i=1}^{\hat{N}} R_i^T A_i^{-1} R_i\right). \tag{2.3}$$

We give in section 2.3 the exact definitions of the subdomain matrices A_i, for $i = 1, \ldots, \hat{N}$, the coarse time correction matrix A_0, as well as the restriction and prolongation matrices R_i, R_i^T, for $i = 0, \ldots, \hat{N}$, where $\hat{N}$ is the number of subdomain matrices of A. The matrix $\mathbb{I} \in \mathbb{R}^{(N+1)d\times(N+1)d}$ is the identity matrix. The first term denotes an additive Schwarz preconditioner in time, which is computed in parallel by using the fine propagators, followed by a coarse correction in time, based on a coarse propagator, which is computed sequentially and transfers the information globally between the different time subdomains.

Furthermore, we show that this two-level additive Schwarz in time preconditioner has the same error propagation as MGRIT with F-relaxation at coarse time points, discussed in [20, 25, 30]. As expected, this shows that the three algorithms parareal, MGRIT with F-relaxation, and two-level additive Schwarz in time preconditioner from (2.3) are equivalent. We also discuss that applying additional fine or coarse propagation steps in the two-level additive Schwarz in time preconditioner is equivalent to MGRIT with FCF-relaxation and MGRIT with F(CF)^2-relaxation or overlapping parareal, discussed in [36]. Faster convergence can be achieved in some cases, but the trade-off is also important to consider. To improve the convergence, a variant of two-level domain decomposition method, referred to as SCS^2 two-level additive Schwarz in time preconditioner, provides a good alternative, since it relies on increasing the number of additive Schwarz in time steps, while keeping only one coarse correction step, which is performed in sequential. Note that the notations S and C used here in the context of two-level additive Schwarz in time preconditioner

correspond to the use of fine and coarse propagators. They are different from F-relaxation and C-relaxation used in MGRIT. Specifically, S and C propagation steps in the two-level additive Schwarz in time preconditioner start from the same coarse time points and propagate to obtain the approximate solution at the end of each time subdomain. While F-relaxation propagates to obtain the approximate solution at fine time points based on the coarse time points, and C-relaxation propagates to obtain the approximate solution at coarse time points based on the previous fine time points, for more detail see [20]. We also explore the usage of Krylov subspace methods for solving the system (2.2). This gives promising numerical results, especially for solving the advection-reaction-diffusion equation with large advection and reaction terms.

The paper is organised as follows. Section 2.2 recalls parareal algorithm and its formulation as a residual correction scheme. Section 2.3 introduces an interpretation of parareal as a two-level additive Schwarz in time preconditioner. Section 2.4 discusses several variants of this two-level additive Schwarz in time preconditioner and gives their convergence analysis. Further, theoretical convergence bounds are given in section 2.5. Several numerical experiments are presented in section 2.6, where we consider the Dahlquist problem, the heat equation, and the advection-reaction-diffusion equation. Conclusions and perspectives are given in section 2.7.

2.2 Parareal algorithm

In this section we describe the parareal algorithm by following its presentation from e.g. [37]. For the simplicity of the exposition, we consider the scalar linear time dependent problem,

$$\frac{du}{dt} = f(u) \ , \ u(0) = \mathrm{u}_0, \ \ u(t) \in \mathbb{R}^d, t \ \text{ in } (0,T), \tag{2.4}$$

The time interval $[0,T]$ is decomposed into N_C uniform time subdomains $[T_n, T_{n+1}]$ with $n = 0, \ldots, N_C - 1$. Parareal uses two solvers, a fine solver $\mathcal{F}(T_{n+1}, T_n, U_n)$, which gives a very good approximate, and a coarse solver $\mathcal{G}(T_{n+1}, T_n, U_n)$, which gives a coarse approximate of the solution at time T_{n+1} starting from the initial solution U_n at time T_n. The initial approximate U_n^0 at coarse time points is obtained typically by using sequentially the coarse solver,

$$U_{n+1}^0 = \mathcal{G}(T_{n+1}, T_n, U_n^0), \ U_0^0 = \mathrm{u}_0.$$

From this initial solution in time, parareal iteratively computes a new approximate of the solution of equation (2.4) until some convergence criterion is met. At each iteration $k+1$, $k \geq 0$, a new approximate is computed as,

$$U_{n+1}^{k+1} = \mathcal{G}(T_{n+1}, T_n, U_n^{k+1}) + \mathcal{F}(T_{n+1}, T_n, U_n^k) - \mathcal{G}(T_{n+1}, T_n, U_n^k). \tag{2.5}$$

The coarse and the fine solvers can be chosen in various ways. Very often a higher order approximation is used for the fine solver and a lower order approximation is used for the coarse solver. The coarse solver can also solve a different problem, which is simpler to solve than the original one, as long as it gives an acceptable approximate of the solution. However, the coarse solver plays an important role in the convergence of the parareal

algorithm. It should be chosen in such a way that it is cheap but accurate enough compared to the fine one, otherwise parareal algorithm can converge slowly. One simple approach is to choose the same discretizations in both time and space for both coarse and fine solvers, but with larger time step Δt for the coarse solver and smaller δt for the fine solver. Furthermore, one can also use a coarsened spatial mesh for the coarse solver, see [86].

2.2.1 Parareal execution from an algebraic point of view

Consider the time dependent problem from equation (2.4) for which the time interval $[0, T]$ is divided into N uniform time slices $[t_n, t_{n+1}]$ with length δt, for $n = 0, \ldots, N-1$. On the other hand, $[0, T]$ is also partitioned into N_C uniform coarse time intervals $[T_l, T_{l+1}]$ with length ΔT, for $l = 0, \ldots, N_C - 1$. We denote by ϕ a stable discretization method in time such as forward Euler, backward Euler, Runge-Kutta or higher order methods, and by $\phi_{\Delta T}$ the coarse solver for which the same methods are used but with larger time step, or lower order methods or spatial coarsening, in particular $\phi_{\Delta T}$ approximates the fine solver ϕ^m. Let δt be the fine time step and $\Delta t = \Delta T = m\delta t$ be the coarse time step (we use one coarse time step for the coarse solver on each coarse time interval), in which m denotes the number of fine time steps on each coarse time interval. We note that the error propagation and convergence analysis in sections 2.3, 2.4, 2.5 are studied with the assumption that ϕ and $\phi_{\Delta T}$ can be diagonalized by the same set of eigenvectors, in cases when ϕ and $\phi_{\Delta T}$ have the same spatial discretization, as stated in [20]. Furthermore, the analysis of spatial discretization can also be found in [59]. Without loss of generality, we consider in this work the same discretization methods in both time and space for both coarse and fine solvers, namely the backward Euler in time and centered finite difference method in space. However discretizations as forward Euler, Runge-Kutta or higher order methods can also be used in the same framework , we illustrate this by using Runge-Kutta 4 for the fine solver in section 2.6.4. In this work we focus on the linear constant-coefficient partial differential equations, in particular the heat equation and the advection-reaction-diffusion equation. By sequentially applying ϕ, the linear system of equations obtained has the form:

$$AU_F := \begin{bmatrix} \mathcal{I} & & & \\ -\phi & \mathcal{I} & & \\ & \ddots & \ddots & \\ & & -\phi & \mathcal{I} \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_N \end{bmatrix} = \begin{bmatrix} \mathrm{u}_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} =: f, \tag{2.6}$$

where $A \in \mathbb{R}^{(N+1)d \times (N+1)d}$ denotes the time-stepping coefficient matrix, $\mathcal{I} \in \mathbb{R}^{d \times d}$ denotes the identity matrix and $\phi \in \mathbb{R}^{d \times d}$ denotes the discretization matrix. This system of equations can be solved by using a direct method in which the solutions $u_i, i = 0, \ldots, N$ at different time steps are obtained sequentially. This results in a complexity of N time steps, each time step being solved by using ϕ. But instead of just using ϕ, parareal combines the use of both coarse and fine solvers to result in a faster algorithm in which the fine solvers are performed in parallel.

We describe parareal by considering a simple two-level temporal mesh for which $m = 2$, as displayed in Figure 2.1. With this choice, the fine nodes are defined at all time points $\{t_0, t_1, t_2, \ldots, t_N\}$, while the coarse nodes are defined at even time points $\{t_0, t_2, \ldots, t_N\}$. At

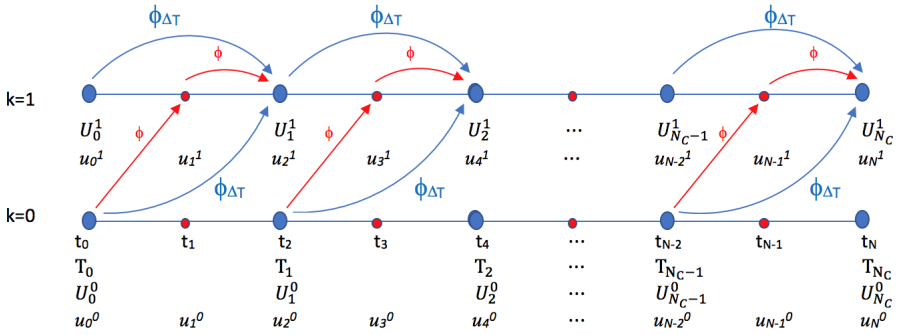

Figure 2.1: Two-level temporal mesh and parareal execution.

the initial step $k = 0$, the initial approximate of the coarse solution is obtained by applying $\phi_{\Delta T}$ sequentially and the fine solution is obtained by interpolating. Let $\mathcal{F}(T_{n+1}, T_n, U_n^k) := \phi^2 U_n^k$ be the fine propagator and $\mathcal{G}(T_{n+1}, T_n, U_n^k) := \phi_{\Delta T} U_n^k$ be the coarse propagator, parareal iteration from (2.5) becomes,

$$U_{n+1}^{k+1} = \phi_{\Delta T} U_n^{k+1} + \phi^2 U_n^k - \phi_{\Delta T} U_n^k,$$

where U_n^k corresponds to u_{2n}^k which denotes the parareal solution at coarse time point t_{2n}, $n = 0, \ldots, N/2$ and iteration k. In detail, parareal computes the approximate solutions at fine time points as follows,

$$u_i^{k+1} = \begin{cases} \mathrm{u}_0, & \text{if } i = 0, \\ \phi u_{i-1}^k, & \text{for } i = 1, 3, \ldots, N-1, \\ \phi_{\Delta T} u_{i-2}^{k+1} + \phi u_{i-1}^{k+1} - \phi_{\Delta T} u_{i-2}^k, & \text{for } i = 2, 4, \ldots, N. \end{cases} \tag{2.7}$$

As it can be seen from Figure 2.1, the fine approximate solutions can be computed in parallel based on the coarse approximate solutions from the previous iterations. Generally for arbitrary $m \geq 2$, we have similarly,

$$U_{n+1}^{k+1} = \phi_{\Delta T} U_n^{k+1} + \phi^m U_n^k - \phi_{\Delta T} U_n^k. \tag{2.8}$$

2.2.2 Expression of the standard residual correction scheme

As presented in e.g. [36], parareal algorithm can be seen as a preconditioned residual correction scheme of a reduced system representing only the coarse time solutions, which is obtained from the original system of equations (2.6). For this, a coarse matrix A_C represents the time steps of the coarse level (here we keep every second time point on each time interval), U_C represents the unknown solutions and f_C the right-hand side at coarse time points,

$$A_C U_C := \begin{bmatrix} \mathcal{I} & & & & \\ -\phi^2 & \mathcal{I} & & & \\ & \ddots & \ddots & & \\ & & -\phi^2 & \mathcal{I} & \\ & & & -\phi^2 & \mathcal{I} \end{bmatrix} \begin{bmatrix} u_0 \\ u_2 \\ \vdots \\ u_{N-2} \\ u_N \end{bmatrix} = \begin{bmatrix} \mathrm{u}_0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} =: f_C. \tag{2.9}$$

This reduced system of equations (2.9) produces exactly the same solutions as the original system (2.6) at coarse time points. A preconditioner $\tilde{M}$ which approximates the coarse matrix A is obtained by approximating each fine time integration propagator ϕ^2 by one coarse integration propagator $\phi_{\Delta T}$,

$$\tilde{M} := \begin{bmatrix} \mathcal{I} & & & & \\ -\phi_{\Delta T} & \mathcal{I} & & & \\ & \ddots & \ddots & & \\ & & -\phi_{\Delta T} & \mathcal{I} & \\ & & & -\phi_{\Delta T} & \mathcal{I} \end{bmatrix}.$$

By using the preconditioned stationary iteration at the coarse level, we obtain at iteration k,

$$U_C^{k+1} = U_C^k + \tilde{M}^{-1}(f_C - A_C U_C^k), \tag{2.10}$$

which can be written as,

$$\tilde{M}(U_C^{k+1} - U_C^k) = f_C - A_C U_C^k, \tag{2.11}$$

or explicitly written as,

$$\begin{bmatrix} \mathcal{I} & & & & \\ -\phi_{\Delta T} & \mathcal{I} & & & \\ & \ddots & \ddots & & \\ & & -\phi_{\Delta T} & \mathcal{I} & \\ & & & -\phi_{\Delta T} & \mathcal{I} \end{bmatrix} \begin{bmatrix} u_0^{k+1} - u_0^k \\ u_2^{k+1} - u_2^k \\ \vdots \\ u_{N-2}^{k+1} - u_{N-2}^k \\ u_N^{k+1} - u_N^k \end{bmatrix} = \begin{bmatrix} \mathrm{u}_0 - u_0^k \\ \phi^2 u_0^k - u_2^k \\ \vdots \\ \phi^2 u_{N-4}^k - u_{N-2}^k \\ \phi^2 u_{N-2}^k - u_N^k \end{bmatrix}. \tag{2.12}$$

It can be easily seen that the solutions u_{2i}^{k+1} for $i = 0, \ldots, N/2$ obtained by solving equation (2.12) are the same as the solutions obtained by parareal in equation (2.7).

We consider now solving the system $AU_F = f$ from equation (2.6) at the fine level. We introduce a matrix M_{SC} and we show that parareal algorithm is equivalent to solving $AU_F = f$ by using a stationary iteration preconditioned by M_{SC}^{-1}. The preconditioned stationary iteration for solving $AU_F = f$ at the fine level becomes,

$$U_F^{k+1} = U_F^k + M_{SC}^{-1}(f - AU_F^k), \tag{2.13}$$

or equivalently,

$$M_{SC}(U_F^{k+1} - U_F^k) = f - AU_F^k, \tag{2.14}$$

Note that (2.14) acts at the fine level, so M_{SC} is different from $\tilde{M}$ in (2.11). In other words, M_{SC} has to deal with both unknowns at coarse and fine time points. The matrix M_{SC} is defined in the following lemma.

Lemma 2.2.1. *Let $\mathcal{F}(T_{n+1}, T_n, u_n^k) := \phi^m u_n^k$ and $\mathcal{G}(T_{n+1}, T_n, u_n^k) := \phi_{\Delta T} u_n^k$ denote the fine and the coarse solvers, respectively. For $m \geq 2$, (2.14) is equivalent to parareal algorithm with*

M_{SC} defined as,

$$M_{SC} := \begin{bmatrix} \mathcal{I} & & & & & & & & & \\ & \mathcal{I} & & & & & & & & \\ & -\phi & \mathcal{I} & & & & & & & \\ & & & \ddots & & & & & & \\ -\phi_{\Delta T} & & & -\phi & \mathcal{I} & & & & & \\ & & & & & \ddots & & & & \\ & & & & & & \mathcal{I} & & & \\ & & & & & & -\phi & \mathcal{I} & & \\ & & & & & & & & \ddots & \\ & & & & & -\phi_{\Delta T} & & & -\phi & \mathcal{I} \end{bmatrix}. \quad (2.15)$$

md × *md*

Proof. For $m = 2$, (2.14) becomes,

$$\begin{bmatrix} \mathcal{I} & & & & & \\ & \mathcal{I} & & & & \\ -\phi_{\Delta T} & -\phi & \mathcal{I} & & & \\ & & & \ddots & & \\ & & & & \mathcal{I} & \\ & & & -\phi_{\Delta T} & -\phi & \mathcal{I} \end{bmatrix} \begin{bmatrix} u_0^{k+1} - u_0^k \\ u_1^{k+1} - u_1^k \\ u_2^{k+1} - u_2^k \\ \vdots \\ u_{N-1}^{k+1} - u_{N-1}^k \\ u_N^{k+1} - u_N^k \end{bmatrix} = \begin{bmatrix} \mathrm{u}_0 - u_0^k \\ \phi u_0^k - u_1^k \\ \phi u_1^k - u_2^k \\ \vdots \\ \phi u_{N-2}^k - u_{N-1}^k \\ \phi u_{N-1}^k - u_N^k \end{bmatrix}. \quad (2.16)$$

Simplifying (2.16) gives

$$u_i^{k+1} = \begin{cases} \mathrm{u}_0, & \text{if } i = 0, \\ \phi u_{i-1}^k, & \text{for } i = 1, 3, \ldots, N-1, \\ \phi_{\Delta T} u_{i-2}^{k+1} + \phi u_{i-1}^{k+1} - \phi_{\Delta T} u_{i-2}^k, & \text{for } i = 2, 4, \ldots, N. \end{cases}$$

Generalize for $m > 2$, for $j = 0, m, 2m, \ldots, N - m$ and by induction we have,

$$\begin{aligned} u_0^{k+1} &= \mathrm{u}_0, \\ u_{j+1}^{k+1} &= \phi u_j^k, \\ u_{j+2}^{k+1} &= \phi u_{j+1}^{k+1}, \\ &\vdots \\ u_{j+m-1}^{k+1} &= \phi u_{j+m-2}^{k+1}, \\ u_{j+m}^{k+1} &= \phi_{\Delta T} u_j^{k+1} + \phi u_{j+m-1}^{k+1} - \phi_{\Delta T} u_j^k = \phi_{\Delta T} u_j^{k+1} + \phi^m u_j^k - \phi_{\Delta T} u_j^k, \end{aligned}$$

which is identical to (2.8) and concludes the proof. □

Hence, instead of solving the system equation (2.6) by using a direct method, parareal algorithm uses the stationary iteration defined in equation (2.13) preconditioned by M_{SC} as defined in (2.15). In addition, Krylov subspace methods as GMRES can also be used

to accelerate the convergence of parareal. In the numerical experiments section 2.6.3 we present results obtained by using GMRES for solving the preconditioned linear system,

$$M_{SC}^{-1} A U_F = M_{SC}^{-1} f.$$

It will be seen that GMRES improves slightly the convergence of parareal and it allows to solve problems for which parareal has difficulty to converge, as in the case when the advection and reaction coefficients are large compared to the diffusion term for the advection-reaction-diffusion problem. However in general it does not improve drastically the convergence of parareal for our test problems, and this was also observed in previous works as [79] which studied the acceleration of waveform relaxation methods.

2.3 Interpretation of parareal as a two-level additive Schwarz in time preconditioner

In this section we present an interpretation of parareal as a two-level domain decomposition method. For this we show that the inverse of the preconditioner M_{SC} from (2.15) can be expressed as a first level additive Schwarz preconditioner that relies on using the fine propagator ϕ^m in each time subdomain, followed by a coarse time correction based on using the coarse propagator $\phi_{\Delta T}$.

We introduce first some notations. Let $A \in \mathbb{R}^{(N+1)d \times (N+1)d}$ be the time-stepping matrix as defined in section 2.2.1. The matrices $\mathbb{I} \in \mathbb{R}^{(N+1)d \times (N+1)d}$ and $\mathcal{I} \in \mathbb{R}^{d \times d}$ are identity matrices. The matrix A is decomposed into $N_C + 1$ non-overlapping subdomains $\{\Omega_i\}_{1 \le i \le N_C+1}$, where $N_C = N/m$ denotes the number of coarse time intervals. The matrix A is a block matrix, the blocks being defined as $\{A_{ij}\}_{1 \le i,j \le N+1} \in \mathbb{R}^{d \times d}$. As displayed in equation (2.6), A_{ij} can be the ϕ matrix, the identity or the zero matrix. Let $\mathfrak{N} = \{1, \dots, N+1\}$ be the set of indices of A, which corresponds to the fine time steps $\{t_0, \dots, t_N\}$. Let $\mathfrak{N}_i$, $i \in \{1, \dots, N_C + 1\}$ be the subset of $\mathfrak{N}$ such that $\mathfrak{N}_i$ represents the subset of indices of subdomain i, we define $\mathfrak{N}_i$ as,

$$\mathfrak{N}_i = \begin{cases} \{1\}, & \text{if } i = 1, \\ \{m(i-2)+2, \dots, m(i-1)+1\}, & \text{for } i = 2, \dots, N_C + 1, \end{cases} \tag{2.17}$$

the restriction matrix R_i is defined as,

$$R_i = \begin{cases} \mathcal{I}, & \text{if } i = 1, \\ \mathbb{I}(\mathfrak{N}_i, :), & \text{for } i = 2, \dots, N_C + 1, \end{cases} \tag{2.18}$$

where $\mathbb{I}(\mathfrak{N}_i, :)$ denotes the submatrix of $\mathbb{I}$ formed by the rows whose indices belong to $\mathfrak{N}_i$. The prolongation matrix R_i^T is the transpose of R_i. The subdomain matrices $\{A_i\}_{1 \le i \le N_C+1}$ are defined as,

$$A_i = \begin{cases} \mathcal{I}, & \text{if } i = 1, \\ R_i A R_i^T = \begin{bmatrix} \mathcal{I} & & & \\ -\phi & \mathcal{I} & & \\ & \ddots & \ddots & \\ & & -\phi & \mathcal{I} \\ & & & -\phi \quad \mathcal{I} \end{bmatrix}, & \text{for } i = 2, \dots, N_C + 1. \end{cases}$$

For $i \geq 2$, $A_i = R_i A R_i^T$ is an $md \times md$ block matrix. The inverse of A_i can be computed as,

$$A_i^{-1} = \begin{cases} \mathcal{I}, & \text{if } i = 1, \\ \begin{bmatrix} \mathcal{I} & & & & & \\ \phi & \mathcal{I} & & & & \\ \phi^2 & \phi & \mathcal{I} & & & \\ \phi^3 & \phi^2 & \phi & \mathcal{I} & & \\ \vdots & \vdots & & \ddots & \ddots & \\ \phi^{m-2} & \phi^{m-3} & & \dots & \phi & \mathcal{I} \\ \phi^{m-1} & \phi^{m-2} & & \dots & & \phi & \mathcal{I} \end{bmatrix}, & \text{for } i = 2, \dots, N_C + 1. \end{cases}$$

The first level additive Schwarz in time preconditioner is $\sum_{i=1}^{N_C+1} R_i^T A_i^{-1} R_i$ as presented in the introduction section. The second level coarse time correction is defined as following. Let $\mathfrak{N}_0 = \{1 + im\}_{0 \leq i \leq N_C}$ be the set of indices corresponding to coarse time points and $A_0 \in \mathbb{R}^{(N_C+1)d \times (N_C+1)d}$ be the coarse matrix that solves the reduced system from equation (2.6) at every coarse time point by using the coarse integration propagator $\phi_{\Delta T}$,

$$A_0 = \begin{bmatrix} \mathcal{I} & & & & \\ -\phi_{\Delta T} & \mathcal{I} & & & \\ & \ddots & \ddots & & \\ & & -\phi_{\Delta T} & \mathcal{I} & \\ & & & -\phi_{\Delta T} & \mathcal{I} \end{bmatrix}. \tag{2.19}$$

The coarse problem at coarse time points in the time domain is obtained by using a restriction matrix $R_0 \in \mathbb{R}^{(N_C+1)d \times (N+1)d}$, defined such that the entries of R_0 are identities at positions corresponding to the coarse time points and 0 elsewhere. In particular, R_0 is defined as,

$$R_0 = \mathbb{I}(\mathfrak{N}_0, :), \tag{2.20}$$

in which $\mathfrak{N}_0 = \{1, 1 + m, 1 + 2m, \dots, 1 + N_C m\}$ and the prolongation matrix for the coarse problem is the transpose of R_0. The inverse of A_0 can be computed as,

$$A_0^{-1} = \begin{bmatrix} \mathcal{I} & & & & & & \\ \phi_{\Delta T} & \mathcal{I} & & & & & \\ \phi_{\Delta T}^2 & \phi_{\Delta T} & \mathcal{I} & & & & \\ \phi_{\Delta T}^3 & \phi_{\Delta T}^2 & \phi_{\Delta T} & \mathcal{I} & & & \\ \vdots & \vdots & & \ddots & \ddots & & \\ \phi_{\Delta T}^{N_C-1} & \phi_{\Delta T}^{N_C-2} & & \dots & \phi_{\Delta T} & \mathcal{I} & \\ \phi_{\Delta T}^{N_C} & \phi_{\Delta T}^{N_C-1} & & \dots & & \phi_{\Delta T} & \mathcal{I} \end{bmatrix}.$$

Lemma 2.3.1. *The matrix M_{SC} defined in* (2.15) *can be factored as,*

$$M_{SC} = \left(\sum_{i=1}^{N_C+1} R_i^T A_i R_i \right) (R_0^T A_0 R_0 + \mathbb{I} - R_0^T R_0),$$

and the additive Schwarz in time preconditioner M_{SC}^{-1} is formed by the product of the additive Schwarz term $\sum_{i=1}^{N_C+1} R_i^T A_i^{-1} R_i$ and the coarse time correction term $R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0$,

$$M_{SC}^{-1} = (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \left(\sum_{i=1}^{N_C+1} R_i^T A_i^{-1} R_i \right). \tag{2.21}$$

Proof. We have

$$\left(\sum_{i=1}^{N_C+1} R_i^T A_i R_i \right) (R_0^T A_0 R_0 + \mathbb{I} - R_0^T R_0)$$

$$= \begin{bmatrix} \mathcal{I} & & & & & & & \\ & \mathcal{I} & & & & & & \\ & -\phi & \mathcal{I} & & & & & \\ & & & \ddots & & & & \\ & & & -\phi & \mathcal{I} & & & \\ & & & & & \ddots & & \\ & & & & & & \mathcal{I} & \\ & & & & & & -\phi & \mathcal{I} \end{bmatrix} \begin{bmatrix} \mathcal{I} & & & & & & & \\ & \mathcal{I} & & & & & & \\ & & \mathcal{I} & & & & & \\ & & & \ddots & & & & \\ -\phi_{\Delta T} & & & & \mathcal{I} & & & \\ & & & & & \ddots & & \\ & & & & & & \mathcal{I} & \\ & & & & & & & \ddots \\ & & & & & & & \mathcal{I} \end{bmatrix}$$

$$= \begin{bmatrix} \mathcal{I} & & & & & & & \\ & \mathcal{I} & & & & & & \\ & -\phi & \mathcal{I} & & & & & \\ & & & \ddots & & & & \\ -\phi_{\Delta T} & & & -\phi & \mathcal{I} & & & \\ & & & & & \ddots & & \\ & & & & & & \mathcal{I} & \\ & & & & & & -\phi & \mathcal{I} \end{bmatrix} = M_{SC}.$$

We observe that the matrix $(R_0^T A_0 R_0 + \mathbb{I} - R_0^T R_0)$ can be permuted to a matrix whose first diagonal block is A_0 followed by an identity matrix. Additionally the term $\sum_{i=1}^{N_C+1} R_i^T A_i R_i$ is a block diagonal matrix. Thus we obtain,

$$\begin{aligned} M_{SC}^{-1} &= (R_0^T A_0 R_0 + \mathbb{I} - R_0^T R_0)^{-1} \left(\textstyle\sum_{i=1}^{N_C+1} R_i^T A_i R_i \right)^{-1} \\ &= (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \left(\textstyle\sum_{i=1}^{N_C+1} R_i^T A_i^{-1} R_i \right). \end{aligned}$$

□

The preconditioner M_{SC}^{-1} is applied to a vector at each iteration of the residual correction scheme (2.13). The inverses A_i^{-1} and A_0^{-1} are never formed explicitly, they are applied

to a vector by using a backward solve. We refer to this preconditioner as the SC two-level additive Schwarz in time preconditioner. It is formed by the additive Schwarz preconditioner $\sum_{i=1}^{N_C+1} R_i^T A_i^{-1} R_i$, which corresponds to the use of the fine propagators computed in parallel, followed by a coarse time correction $R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0$, which corresponds to the use of the coarse propagator computed sequentially.

Corollary 2.3.1. *Solving* (2.6) *by using parareal is equivalent to using the residual correction scheme from equation* (2.13) *at the fine level, preconditioned by the SC two-level additive Schwarz in time preconditioner. Each iteration becomes:*

$$U_F^{k+1} = U_F^k + (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^k). \tag{2.22}$$

Proof. The proof is done by combining lemma 2.2.1 and lemma 2.3.1. □

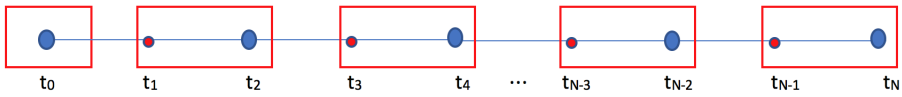

Figure 2.2: Non-overlapping time subdomains with $m = 2$. The fine nodes are defined at all time points $\{t_0, t_1, t_2, \ldots, t_N\}$ and the coarse nodes are defined at even time points $\{t_0, t_2, t_4, \ldots, t_N\}$. The first time subdomain is always defined at $\{t_0\}$, while following time subdomains are defined at $\{t_n, t_{n+1}\}$ for $n = 1, \ldots, N-1$.

We illustrate these results by considering the simple linear time dependent problem (2.4) with $m = 2$, as it can be seen in Figure 2.2. After discretization, the linear system from equation (2.6) needs to be solved. We first decompose the whole time domain into non-overlapping subdomains with indices $\mathfrak{N}_i$ given by (2.17), with the restriction matrices $R_1 \in \mathbb{R}^{d \times (N+1)d}, R_i \in \mathbb{R}^{md \times (N+1)d}$, and the prolongation matrices $R_1^T \in \mathbb{R}^{(N+1)d \times d}, R_i^T \in \mathbb{R}^{(N+1)d \times md}$ for $i = 2, \ldots, N_C + 1$ satisfy (2.18) such that their entries are $\mathcal{I}$ at positions corresponding to the i^{th} subdomain and 0 elsewhere, specifically,

$$R_1 = \begin{bmatrix} \mathcal{I} & 0 & 0 & 0 & \ldots & 0 & 0 \end{bmatrix}, R_2 = \begin{bmatrix} 0 & \mathcal{I} & 0 & 0 & 0 & \ldots & 0 \\ 0 & 0 & \mathcal{I} & 0 & 0 & \ldots & 0 \end{bmatrix}, \ldots, R_{N_C+1} = \begin{bmatrix} 0 & 0 & 0 & 0 & \ldots & \mathcal{I} & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & \mathcal{I} \end{bmatrix}.$$

The subdomain matrices $A_i = R_i A R_i^T$, for $i = 1, \ldots, N_C + 1$, become,

$$A_i = \begin{cases} \mathcal{I}, & \text{for } i = 1, \\ \begin{bmatrix} \mathcal{I} & 0 \\ -\phi & \mathcal{I} \end{bmatrix}, & \text{for } i = 2, \ldots, N_C + 1. \end{cases}$$

Let $\mathfrak{N}_0 = \{1, 3, 5, \ldots, N+1\}$ be the set of indices corresponding to coarse time points $\{t_0, t_2, \ldots t_N\}$ as displayed in Figure 2.3. Let $A_0 \in \mathbb{R}^{(N_C+1)d \times (N_C+1)d}$ be the coarse matrix as

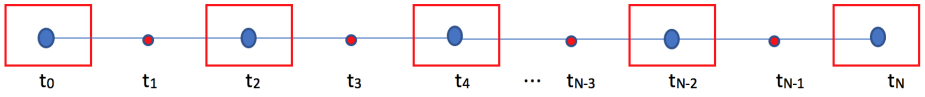

Figure 2.3: Coarse time correction defined at even time points $\{t_0, t_2, t_4, \ldots, t_N\}$.

defined in (2.19) and the restriction matrix $R_0 \in \mathbb{R}^{(N_C+1)d\times(N+1)d}$ satisfies (2.20), namely,

$$R_0 = \begin{bmatrix} \mathcal{I} & 0 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 0 & \mathcal{I} & 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & 0 & \mathcal{I} & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathcal{I} \end{bmatrix}.$$

The matrix M_{SC} becomes,

$$\begin{aligned} M_{SC} &= \sum_{i=1}^{N_C+1} R_i^T A_i R_i (R_0^T A_0 R_0 + \mathbb{I} - R_0^T R_0) \\ &= \begin{bmatrix} \mathcal{I} & & & & & & \\ 0 & \mathcal{I} & & & & & \\ -\phi & \mathcal{I} & & & & & \\ & 0 & \mathcal{I} & & & & \\ & -\phi & \mathcal{I} & & & & \\ & & \ddots & \ddots & & & \\ & & & 0 & \mathcal{I} & & \\ & & & -\phi & \mathcal{I} & & \end{bmatrix} \begin{bmatrix} \mathcal{I} & & & & & & \\ 0 & \mathcal{I} & & & & & \\ -\phi_{\Delta T} & 0 & \mathcal{I} & & & & \\ & 0 & \mathcal{I} & & & & \\ & -\phi_{\Delta T} & 0 & \mathcal{I} & & & \\ & & \ddots & & \ddots & & \\ & & & 0 & 0 & \mathcal{I} & \\ & & & & -\phi_{\Delta T} & 0 & \mathcal{I} \end{bmatrix} \\ &= \begin{bmatrix} \mathcal{I} & & & & & & \\ 0 & \mathcal{I} & & & & & \\ -\phi_{\Delta T} & -\phi & \mathcal{I} & & & & \\ & & 0 & \mathcal{I} & & & \\ & & -\phi_{\Delta T} & -\phi & \mathcal{I} & & \\ & & & \ddots & & \ddots & \\ & & & & 0 & 0 & \mathcal{I} \\ & & & & & -\phi_{\Delta T} & -\phi & \mathcal{I} \end{bmatrix}. \end{aligned} \tag{2.23}$$

It can be seen that M_{SC} from (2.23) is the same as the matrix M_{SC} defined in lemma 2.2.1 in case $m = 2$, the preconditioner M_{SC}^{-1} is computed following lemma 2.3.1,

$$M_{SC}^{-1} = (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{i=1}^{N_C+1} R_i^T A_i^{-1} R_i,$$

and then corollary 2.3.1 gives the residual correction scheme of the problem (2.4) at the fine level (2.22) with SC two-level additive Schwarz in time preconditioner M_{SC}^{-1} which is equivalent to parareal.

It was shown in a series of papers, e.g. [25, 37], that MGRIT with F-relaxation is equivalent to parareal algorithm. We show now that MGRIT with F-relaxation is also equivalent to SC two-level additive Schwarz in time preconditioner by computing the error propagation matrix at coarse time points. The error propagation of (2.22) is governed by

$$e^{k+1} = (\mathbb{I} - M_{SC}^{-1}A)e^k, \tag{2.24}$$

where $e^k := U_F - U_F^k$, and U_F, U_F^k denote the exact solution and the approximate solution, respectively. The iteration matrix has the form,

$$\mathbb{I} - M_{SC}^{-1}A = \mathbb{I} - (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{i=1}^{N_C+1} R_i^T A_i^{-1} R_i A.$$

Note that we consider M_{SC}^{-1} as a two-level additive Schwarz preconditioner in the time domain and the matrix A is not symmetric. Hence we cannot exploit the theory of Schwarz-type algorithms for symmetric positive definite matrices for which the preconditioned system $M_{SC}^{-1}A$ can be expressed as sums of orthogonal projection matrices P_i, for $i = 1, 2, \ldots, N_C + 1$, for further details see [16]. Instead we study the error propagation matrix produced in the residual correction scheme (2.22). The following lemma shows that the error propagation matrix produces exactly the same error after one iteration at coarse time points as MGRIT with F-relaxation, for which the error is given in [20, Lemma 3.1].

Remark 2.3.1. *The error propagation matrix $\mathbb{I} - M_{SC}^{-1}A$ in (2.24) describes the propagation of errors of (2.22) at both coarse and fine levels. In the following sections, e.g., lemma 2.3.2, 2.4.1, 2.4.2 and 2.4.3, for the convenience of comparison with parareal and the variants, we only consider the error propagation matrices at the coarse level. We also remark that ϕ and $\phi_{\Delta T}$ commute due to the assumption that they can be diagonalized by the same set of eigenvectors.*

Lemma 2.3.2. *Let U_F be the exact solution of (2.6), U_F^k be an approximate solution from (2.13), $e^k := U_F - U_F^k$ and denote by e_j^k the error at iteration k and time t_j with $j = 1, 2, \ldots, N$. The error at coarse time points generated at iteration $k+1$ of (2.13) with SC two-level additive Schwarz in time preconditioner defined in (2.21) satisfies:*

$$\begin{aligned} e_0^{k+1} &= 0, \\ e_{hm}^{k+1} &= \sum_{r=0}^{h-1} \phi_{\Delta T}^{h-1-r} (\phi^m - \phi_{\Delta T}) e_{rm}^k, \quad h = 1, 2, \ldots, N_C. \end{aligned} \tag{2.25}$$

Proof. We denote by $e_{C,SC}^k$ and $e_{C,SC}^{k+1}$ the errors at coarse time points at iteration k and $k+1$ respectively and by E_{SC} the error propagation matrix at coarse time points for SC two-level additive Schwarz in time preconditioner. The error propagation from equation (2.24) at coarse time points yields,

$$e_{C,SC}^{k+1} = E_{SC} e_{C,SC}^k, \tag{2.26}$$

note that ϕ and $\phi_{\Delta T}$ commute, so (2.26) can also be written as,

$$\begin{bmatrix} e_0^{k+1} \\ e_m^{k+1} \\ e_{2m}^{k+1} \\ e_{3m}^{k+1} \\ \vdots \\ e_{N_C m}^{k+1} \end{bmatrix} = (\phi^m - \phi_{\Delta T}) \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ \mathcal{I} & 0 & 0 & \dots & 0 & 0 \\ \phi_{\Delta T} & \mathcal{I} & 0 & \dots & 0 & 0 \\ \phi_{\Delta T}^2 & \phi_{\Delta T} & \mathcal{I} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi_{\Delta T}^{N_C-1} & \phi_{\Delta T}^{N_C-2} & \phi_{\Delta T}^{N_C-3} & \dots & \mathcal{I} & 0 \end{bmatrix} \begin{bmatrix} e_0^k \\ e_m^k \\ e_{2m}^k \\ e_{3m}^k \\ \vdots \\ e_{N_C m}^k \end{bmatrix}.$$

Equation (2.25) follows. □

2.4 Variants of SC two-level additive Schwarz in time preconditioner and convergence analysis

In this section we study several variants of SC two-level additive Schwarz in time preconditioner and discuss their equivalence with MGRIT with FCF-relaxation, MGRIT with F(CF)^2-relaxation or overlapping parareal. In addition, we derive a method, referred to as SCS^2 two-level additive Schwarz in time preconditioner, and discuss its suitability for exploiting parallel computing.

We first describe the SCS variant of SC two-level additive Schwarz in time preconditioner. It is obtained by first applying SC two-level additive Schwarz in time preconditioner, that is one fine solve followed by one coarse solve, and then adding one more fine solve. In detail, one iteration of the residual correction scheme is performed as follows:

$$\begin{aligned} U_F^{k+\frac{1}{2}} &= U_F^k + (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^k), \\ U_F^{k+1} &= U_F^{k+\frac{1}{2}} + \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^{k+\frac{1}{2}}). \end{aligned}$$

The error propagation matrix is defined as,

$$\left[\mathbb{I} - \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j A \right] \left[\mathbb{I} - (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j A \right].$$

The following lemma gives the error propagation of the SCS variant of SC two-level additive Schwarz in time preconditioner. It can be seen that the error propagation matrix produces exactly the same error at coarse time points after one iteration as MGRIT with FCF-relaxation. The result for MGRIT with FCF-relaxation is described in [20, Lemma 3.2].

Lemma 2.4.1. *Let U_F be the exact solution of (2.4), U_F^k be an approximate solution from (2.13), $e^k := U_F - U_F^k$ and denote by e_j^k the error at iteration k and time t_j with $j = 1, 2, \dots, N$. The error at coarse time points generated at iteration $k+1$ of the residual correction scheme*

from equation (2.13) *preconditioned by SCS two-level additive Schwarz in time preconditioner satisfies:*

$$\begin{aligned} e_0^{k+1} &= 0, \\ e_m^{k+1} &= 0, \\ e_{hm}^{k+1} &= \sum_{r=0}^{h-2} \phi_{\Delta T}^{h-2-r}(\phi^m - \phi_{\Delta T})\phi^m e_{rm}^k, \quad h = 2, 3, \ldots, N_C. \end{aligned} \tag{2.27}$$

Proof. We denote by $e_{C,SCS}^k$ and $e_{C,SCS}^{k+1}$ the errors at coarse time points at iteration k and $k+1$ respectively and by E_{SCS} the error propagation matrix at coarse time points for SCS two-level additive Schwarz in time preconditioner. We have the relation,

$$e_{C,SCS}^{k+1} = E_{SCS} e_{C,SCS}^k, \tag{2.28}$$

in which $E_{SCS} =$

$$(\phi^m - \phi_{\Delta T})\phi^m \begin{bmatrix} 0 & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 \\ \mathcal{I} & 0 & 0 & 0 & \ldots & 0 & 0 \\ \phi_{\Delta T} & \mathcal{I} & 0 & 0 & \ldots & 0 & 0 \\ \phi_{\Delta T}^2 & \phi_{\Delta T} & \mathcal{I} & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \phi_{\Delta T}^{N_C-2} & \phi_{\Delta T}^{N_C-3} & \phi_{\Delta T}^{N_C-4} & \ldots & \mathcal{I} & 0 & 0 \end{bmatrix}.$$

The relations in equation (2.27) follow. □

The SCS^2 variant of SC two-level additive Schwarz in time preconditioner is obtained by adding one more fine solve based on additive Schwarz as follows:

$$\begin{aligned} U_F^{k+\frac{1}{3}} &= U_F^k + (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^k), \\ U_F^{k+\frac{1}{2}} &= U_F^{k+\frac{1}{3}} + \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^{k+\frac{1}{3}}), \\ U_F^{k+1} &= U_F^{k+\frac{1}{2}} + \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^{k+\frac{1}{2}}). \end{aligned}$$

The error propagation matrix is defined as,

$$\left[\mathbb{I} - \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j A\right]^2 \left[\mathbb{I} - (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j A\right].$$

Lemma 2.4.2. *Let U_F be the exact solution of* (2.4)*, U_F^k be an approximate solution from* (2.13)*, $e^k := U_F - U_F^k$ and denote by e_j^k the error at iteration k and time t_j with $j = 1, 2, \ldots, N$.*

The error at coarse time points generated at iteration $k+1$ of the residual correction scheme from equation (2.13) *with SCS^2 two-level additive Schwarz in time preconditioner satisfies:*

$$\begin{aligned} e_0^{k+1} &= 0, \\ e_m^{k+1} &= 0, \\ e_{2m}^{k+1} &= 0, \\ e_{hm}^{k+1} &= \sum_{r=0}^{h-3} \phi_{\Delta T}^{h-3-r}(\phi^m - \phi_{\Delta T})\phi^{2m} e_{rm}^k, \quad h = 3, 4, \ldots, N_C. \end{aligned} \tag{2.29}$$

Proof. Let e_{C,SCS^2}^k and e_{C,SCS^2}^{k+1} be the errors at coarse time points at iteration k and $k+1$ respectively and let E_{SCS^2} be the error propagation matrix at coarse time points for SCS^2 two-level additive Schwarz in time preconditioner. We have the relation,

$$e_{C,SCS^2}^{k+1} = E_{SCS^2} e_{C,SCS^2}^k, \tag{2.30}$$

in which $E_{SCS^2} =$

$$(\phi^m - \phi_{\Delta T})\phi^{2m} \begin{bmatrix} 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ \mathcal{I} & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ \phi_{\Delta T} & \mathcal{I} & 0 & 0 & \ldots & 0 & 0 & 0 \\ \phi_{\Delta T}^2 & \phi_{\Delta T} & \mathcal{I} & 0 & \ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \phi_{\Delta T}^{N_C-3} & \phi_{\Delta T}^{N_C-4} & \phi_{\Delta T}^{N_C-5} & \ldots & \mathcal{I} & 0 & 0 & 0 \end{bmatrix}.$$

Equation (2.29) follows. ∎

A variant known in the literature as MGRIT with F(CF)$^\nu$-relaxation or overlapping parareal has been shown to converge at most after $k = [N/(\nu+1)]$ iterations [36, Theorem 5]. For the case $\nu = 2$, in the framework of domain decomposition, this variant is referred to as S(CS)^2 two-level additive Schwarz in time preconditioner and it is obtained as follows:

$$\begin{aligned} U_F^{k+\frac{1}{3}} &= U_F^k + (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^k), \\ U_F^{k+\frac{1}{2}} &= U_F^{k+\frac{1}{3}} + (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^{k+\frac{1}{3}}), \\ U_F^{k+1} &= U_F^{k+\frac{1}{2}} + \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j (f - A U_F^{k+\frac{1}{2}}). \end{aligned}$$

The error propagation matrix is defined as,

$$\left[\mathbb{I} - \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j A\right] \left[\mathbb{I} - (R_0^T A_0^{-1} R_0 + \mathbb{I} - R_0^T R_0) \sum_{j=1}^{N_C+1} R_j^T A_j^{-1} R_j A\right]^2.$$

For completeness we give in the following lemma the error of this variant.

Lemma 2.4.3. *Let U_F be the exact solution of (2.4), U_F^k be an approximate solution from (2.13), $e^k := U_F - U_F^k$ and denote by e_j^k the error at iteration k and time t_j with $j = 1, 2, \ldots, N$. The error at coarse time points generated at iteration $k+1$ of (2.13) with S(CS)^2 two-level additive Schwarz in time preconditioner satisfies:*

$$\begin{aligned} e_0^{k+1} &= 0, \\ e_m^{k+1} &= 0, \\ e_{2m}^{k+1} &= 0, \\ e_{hm}^{k+1} &= \sum_{r=0}^{h-3} (h-2-r)\phi_{\Delta T}^{h-3-r}(\phi^m - \phi_{\Delta T})^2 \phi^m e_{rm}^k, \quad h = 3, 4, \ldots, N_C. \end{aligned} \tag{2.31}$$

Proof. We denote by $e_{C,S(CS)^2}^k$ and $e_{C,S(CS)^2}^{k+1}$ the errors at coarse time points at iteration k and $k+1$ respectively and by $E_{S(CS)^2}$ the error propagation matrix at coarse time points for S(CS)^2 two-level additive Schwarz in time preconditioner. We obtain the relation,

$$e_{C,S(CS)^2}^{k+1} = E_{S(CS)^2} e_{C,S(CS)^2}^k, \tag{2.32}$$

in which $E_{S(CS)^2} =$

$$(\phi^m - \phi_{\Delta T})^2 \phi^m \begin{bmatrix} 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ \mathcal{I} & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ 2\phi_{\Delta T} & \mathcal{I} & 0 & 0 & \ldots & 0 & 0 & 0 \\ 3\phi_{\Delta T}^2 & \phi_{\Delta T} & \mathcal{I} & 0 & \ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ (N_C-2)\phi_{\Delta T}^{N_C-3} & (N_C-3)\phi_{\Delta T}^{N_C-4} & (N_C-4)\phi_{\Delta T}^{N_C-5} & \ldots & \mathcal{I} & 0 & 0 & 0 \end{bmatrix}$$

from which equation (2.31) follows. □

These different variants have different costs in a parallel environment. Given that the fine solve phase based on additive Schwarz is done in parallel and the coarse solve phase has to be done in sequential, the coarse solve is the major limiting factor. For that reason, the SCS^2 variant becomes more noticeable since it uses more fine solve phases based on additive Schwarz while it just performs one coarse solve phase which is done in sequential. The impact of additional fine or coarse solve phases in the preconditioner to the error convergence as well as the computational costs will be discussed in more detail in the next section.

2.5 Convergence estimate

In this section we estimate the convergence of SC two-level additive Schwarz in time preconditioner and its variants by computing the norms of the error propagation matrices. The convergence is estimated based on an eigenvalue analysis for which the coarse and the

fine propagators must have the same eigenvectors. As the assumption in section 2.2 that ϕ and $\phi_{\Delta T}$ have the same eigenvectors, there exists a unitary matrix X, e.g., $X^*X = XX^* = \mathcal{I}$ such that ϕ and $\phi_{\Delta T}$ can be diagonalized as,

$$\begin{aligned} \Lambda &= X^*\phi X = diag(\lambda_1, \lambda_2, \ldots, \lambda_d), \\ \Lambda_{\Delta T} &= X^*\phi_{\Delta T} X = diag(\mu_1, \mu_2, \ldots, \mu_d), \end{aligned}$$

with $|\lambda_i| < 1$ and $|\mu_i| < 1$ for $i = 1, 2, \ldots, d$ since ϕ and $\phi_{\Delta T}$ are stable time-stepping methods.

The matrix E_{SC} defined in equation (2.26) is the error propagation matrix corresponding to SC two-level additive Schwarz in time preconditioner, each element of this matrix is a block matrix of dimension $d \times d$. The error propagation matrices $E_{SCS}, E_{SCS^2}, E_{S(CS)^2}$ corresponding to the variants of SC two-level additive Schwarz in time preconditioner are defined in (2.28), (2.30), (2.32). We then have,

$$\begin{aligned} ||E_{SC}||_1 = ||E_{SC}||_\infty &\leq \max_{1\leq j\leq N_C} \left(\sum_{i=1}^{N_C} ||(E_{SC})_{ij}||_1 \right) = \sum_{i=0}^{N_C-1} ||\phi_{\Delta T}^i(\phi^m - \phi_{\Delta T})||_1 \\ &\leq ||\phi^m - \phi_{\Delta T}||_1 \sum_{i=0}^{N_C-1} ||\phi_{\Delta T}||_1^i = ||\phi^m - \phi_{\Delta T}||_1 \frac{1 - ||\phi_{\Delta T}||_1^{N_C}}{1 - ||\phi_{\Delta T}||_1}, \end{aligned} \tag{2.33}$$

where $(E_{SC})_{ij}$ refers to the block component at row i and column j of E_{SC}. On the other hand, we also have,

$$\begin{aligned} ||E_{SC}||_2 \leq \sqrt{||E_{SC}||_1 ||E_{SC}||_\infty} &\leq ||\phi^m - \phi_{\Delta T}||_1 \frac{1 - ||\phi_{\Delta T}||_1^{N_C}}{1 - ||\phi_{\Delta T}||_1} \\ &\leq ||X(\Lambda^m - \Lambda_{\Delta T})X^*)||_1 \frac{1 - ||X\Lambda_{\Delta T}^{N_C} X^*||_1}{1 - ||X\Lambda_{\Delta T} X^*||_1} \\ &\leq \max_{1\leq j\leq d} \left\{ |\lambda_j^m - \mu_j| \frac{1 - |\mu_j|^{N_C}}{1 - |\mu_j|} \right\}. \end{aligned} \tag{2.34}$$

Similarly we have,

$$||E_{SCS}||_2 \leq \max_{1\leq j\leq d} \left\{ |\lambda_j^m - \mu_j| \frac{1 - |\mu_j|^{N_C-1}}{1 - |\mu_j|} |\lambda_j|^m \right\}, \tag{2.35}$$

$$||E_{SCS^2}||_2 \leq \max_{1\leq j\leq d} \left\{ |\lambda_j^m - \mu_j| \frac{1 - |\mu_j|^{N_C-2}}{1 - |\mu_j|} |\lambda_j|^{2m} \right\}, \tag{2.36}$$

$$||E_{S(CS)^2}||_2 \leq \max_{1\leq j\leq d} C_j, \tag{2.37}$$

in which

$$C_j = (\lambda_j^m - \mu_j)^2 \frac{1 - (N_C - 1)|\mu_j|^{N_C-2} + (N_C - 2)|\mu_j|^{N_C-1}}{(1 - |\mu_j|)^2} |\lambda_j|^m. \tag{2.38}$$

The 2-norm of the errors is estimated in the following theorem.

Theorem 2.5.1. *Let ϕ and $\phi_{\Delta T}$ be simultaneously diagonalizable by the same unitary matrix X and be stable time-stepping methods with eigenvalues λ_i and μ_i respectively , e.g., $|\lambda_i| < 1$ and $|\mu_i| < 1$ for $i = 1, \ldots, d$. The error at coarse time points generated at iteration $k+1$ of* (2.13) *satisfy:*

$$||e^{k+1}_{C,SC}||_2 \leq \max_{1\leq j\leq d}\left\{|\lambda_j^m - \mu_j|\frac{1-|\mu_j|^{N_C}}{1-|\mu_j|}\right\}||e^k_{C,SC}||_2, \tag{2.39}$$

$$||e^{k+1}_{C,SCS}||_2 \leq \max_{1\leq j\leq d}\left\{|\lambda_j^m - \mu_j|\frac{1-|\mu_j|^{N_C-1}}{1-|\mu_j|}|\lambda_j|^m\right\}||e^k_{C,SCS}||_2, \tag{2.40}$$

$$||e^{k+1}_{C,SCS^2}||_2 \leq \max_{1\leq j\leq d}\left\{|\lambda_j^m - \mu_j|\frac{1-|\mu_j|^{N_C-2}}{1-|\mu_j|}|\lambda_j|^{2m}\right\}||e^k_{C,SCS^2}||_2, \tag{2.41}$$

$$||e^{k+1}_{C,S(CS)^2}||_2 \leq \max_{1\leq j\leq d} C_j||e^k_{C,S(CS)^2}||_2, \tag{2.42}$$

where C_j is defined in (2.38).

Proof. Combining (2.33)-(2.34), (2.35), (2.36) and (2.37) leads to the desired results. □

The convergence bounds for SC from (2.39) and SCS from (2.40) are already given in the context of MGRIT with F-relaxation and with FCF-relaxation, see [20], in which the authors estimate the convergence by using the eigenvector expansion of the error to compute the error norm for each eigenmode. In this chapter, we estimate the convergence of SC two-level additive Schwarz in time preconditioner and its variants by computing directly the norms of the error propagation matrices generated at iteration $k+1$ of the residual correction scheme from equation (2.13). The theoretical convergence bounds we obtained for SC and SCS two-level additive Schwarz in time preconditioner are exactly the same with those for MGRIT with F-relaxation and with FCF-relaxation. This once again confirms the equivalence between parareal, MGRIT with F-relaxation and SC two-level additive Schwarz in time preconditioner.

As the work presented in [20], those estimates have a removable singularity that is when $|\mu_j|$ tends to 1. They are also shown to be bounded independently of N_C in many applications. Furthermore, the nominator $1-|\mu_j|^{N_C}$ can be replaced by 1 since the estimates hold for all N_C.

As mentioned in the end of the previous section, these variants have different computational costs for implementation. To make this clear, we follow our setting in section 2.2 to recall the speedup of parareal algorithm from [74] as,

$$\mathcal{S}(N) = \frac{Nm\tau_f}{N\tau_C + K(N\tau_C + m\tau_f)}, \tag{2.43}$$

in which the numerator describes the runtime for the fine propagator over N coarse time intervals while the denominator shows the runtime of parareal algorithm with N processors and K iterations, and τ_C, τ_f denote the computational cost of one step of the coarse and fine propagator. Depend on the number of coarse and fine propagator phases in the preconditioner, we then have different speedup of the variants, more precisely,

$$\mathcal{S}_{SCS}(N) = \frac{Nm\tau_f}{N\tau_C + K(N\tau_C + 2m\tau_f)}, \tag{2.44}$$

$$\mathcal{S}_{SCS^2}(N) = \frac{Nm\tau_f}{N\tau_C + K(N\tau_C + 3m\tau_f)}, \tag{2.45}$$

$$\mathcal{S}_{S(CS)^2}(N) = \frac{Nm\tau_f}{N\tau_C + K(2N\tau_C + 3m\tau_f)}. \tag{2.46}$$

It is obvious that the speedup becomes less efficient as the number of coarse or fine propagator phases increases. However those fine propagator phases are totally performed in parallel, this is a very important characteristic that we can exploit. By adding one or two additional fine propagation steps in the preconditioner, the convergence of parareal from (2.39) can be reduced by a factor of $|\lambda_j|^m$ or $|\lambda_j|^{2m}$ as it can be seen in (2.40), (2.41), especially in the case when the eigenvalues are very small and the number of fine time step per time slice m is very large. Since the S(CS)^2 variant consists of two coarse propagation steps which are performed sequentially, it is not very efficient in a parallel environment. On the other hand, SCS^2 can really off-set the added computational cost since we can take advantage of the parallel computing of the fine solver.

2.6 Numerical results

In this section we first discuss results that show the equivalence between parareal and SC two-level additive Schwarz in time preconditioner for three different problems, Dahlquist problem, heat equation, and advection-reaction-diffusion equation. Numerical experiments investigate the behavior of the convergence rates on short and long time intervals when N_C and m vary. We then discuss the convergence of different variants of two-level domain decomposition preconditioners in time. A comparison between parareal or SC two-level additive Schwarz in time preconditioner and parareal with GMRES acceleration is also conducted. The three linear test cases considered here are the Dahlquist problem with $a_0 = -1, \mathrm{u}_0 = 1$,

$$\frac{du}{dt} = a_0 u, \quad u(0) = \mathrm{u}_0, \quad t \in [0, T], \tag{2.47}$$

the heat equation with $a^* = 3, L = 1, \Delta x = 0.1$, the exact solution $u_{exact} = x(L - x)^2 \exp(-2t)$,

$$\begin{cases} \frac{\partial u}{\partial t} &= a^* \frac{\partial^2 u}{\partial x^2} + f \quad in \ (0, L) \times (0, T), \\ u(x, 0) &= u_0(x) \qquad x \in (0, L), \\ u(0, t) &= u(L, t) = 0 \qquad t \in (0, T), \end{cases} \tag{2.48}$$

and the advection-reaction-diffusion equation with $a = 1, b = 1, c = 1, L = 1, \Delta x = 0.1$, the exact solution $u_{exact} = \sin(2\pi x) \exp(-2t)$,

$$\begin{cases} \frac{\partial u}{\partial t} &= a\frac{\partial^2 u}{\partial x^2} - b\frac{\partial u}{\partial x} + cu + f \quad in \ (0, L) \times (0, T), \\ u(x, 0) &= u_0(x) \qquad x \in (0, L), \\ u(0, t) &= u(L, t) = 0 \qquad t \in (0, T), \end{cases} \tag{2.49}$$

in which the unknowns $u(x, t)$ in (2.48) and (2.49) are considered in $(0, L) \times (0, T) \subset \mathbb{R}^d \times \mathbb{R}$, where d is the space dimension. The source term is denoted by f and is chosen to

obtain the desired exact solution. For simplicity we consider the Dirichlet boundary condition, however the periodic boundary condition is also used in 2.6.4. Note that the same discretization methods are used for both coarse and fine solvers, namely centered finite difference in space and backward Euler in time except the end of section 2.6.4 in which Runge-Kutta 4 is used for the fine solver.

2.6.1 Equivalence between parareal and SC two-level additive Schwarz in time preconditioner

In order to study the short time interval behavior, we use $N_C = 20, T = 1$, while for the long time interval behavior, we use $N_C = 100, T = 100$. With time steps $\Delta t = T/N_C, \delta t = \Delta t/m$, for $m = 20$, the 2-norm (spectral norm) of the error between the approximate solution and the fine sequential solution (obtained by sequentially using the fine solver) is displayed in figure 2.4 for the three test cases. We observe that the convergence curves of parareal and SC two-level additive Schwarz in time preconditioner are almost the same, except for the last iterations, when this may happen because of round-off errors. The bound for SC two-level additive Schwarz in time preconditioner derived in equation (2.39) is sharp, in particular for long time intervals.

For the Dahlquist problem, the errors of parareal and SC two-level additive Schwarz in time preconditioner are in superlinear convergence regime on short time intervals and in linear convergence regime on long time intervals. This behavior is also outlined in [37]. In particular on short time intervals they reach 10^{-13} after 5 iterations while with the same number of iterations, the attained error is 10^{-4} on long time interval.

For the heat equation, a convergence to 10^{-16} is observed for short time interval after 18 iterations. For long time interval, both parareal and SC two-level additive Schwarz in time preconditioner converge to an error of 10^{-17} after 10 iterations.

For the advection-reaction-diffusion equation, in particular for short time interval, both parareal and SC two-level additive Schwarz in time preconditioner converge to an error of 10^{-14} after 18 iterations. For long time interval, both parareal and SC two-level additive Schwarz in time preconditioner converge to an error of 10^{-16} after 15 iterations.

2.6.2 Comparison between variants of SC two-level additive Schwarz in time preconditioner

Numerical experiments are performed to study the convergence of several variants of the SC two-level additive Schwarz in time preconditioner that use additional coarse or fine propagation steps. Similarly to the previous section, the short time interval behavior uses $N_C = 20$ and $T = 1$, and the long time interval behavior uses $N_C = 100$ and $T = 100$. Figures 2.5, 2.6, and 2.7 display the error, in 2-norm, between the approximate solution and the fine sequential solution, with time steps $\Delta t = T/N_C, \delta t = \Delta t/m$, and $m \in \{2, 20\}$, for the Dahlquist problem, heat equation, and advection-reaction-diffusion equation, respectively.

For the Dahlquist problem, Figure 2.5, on short time interval the improvement of SCS, SCS^2 is not very important compared to SC two-level additive Schwarz in time

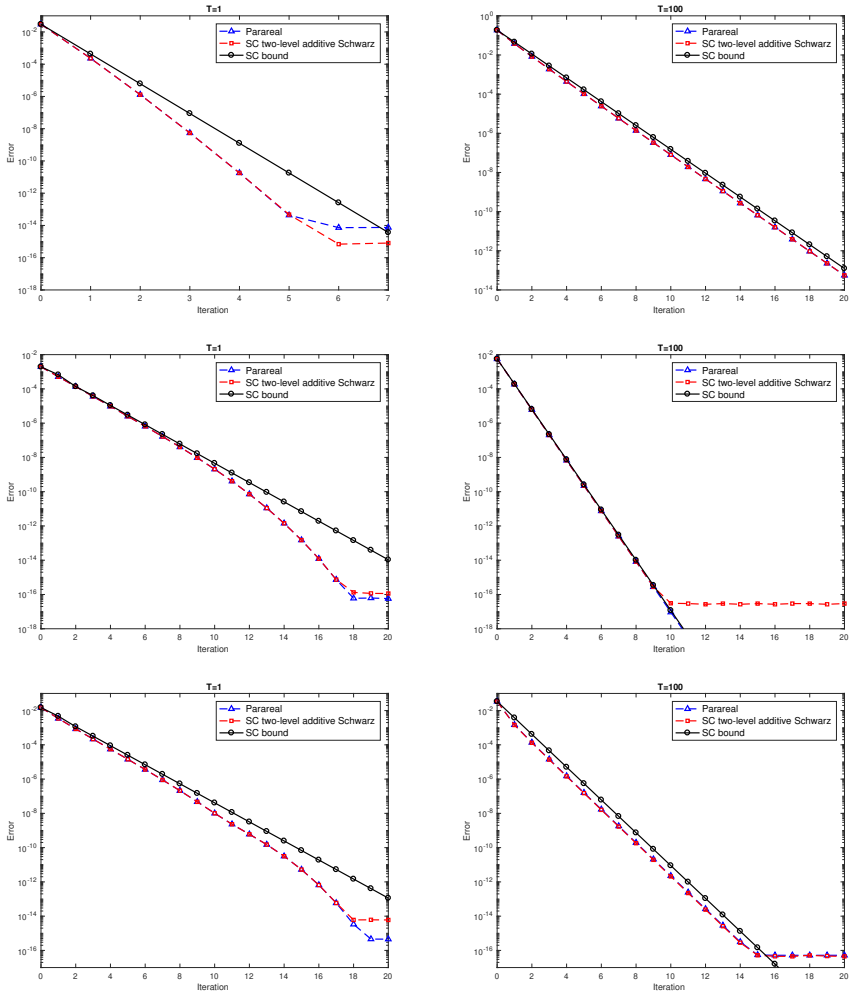

Figure 2.4: Error between approximate solution and fine sequential solution with $m = 20$ for Dahlquist problem (top), heat equation (middle) and advection-reaction-diffusion equation (bottom), $T = 1, N_C = 20$ (first column), and $T = 100, N_C = 100$ (second column).

preconditioner except the S(CS)^2 which converges faster than the others. However on long time interval, the improvement becomes more important. In particular for $m = 2$, SC two-level additive Schwarz in time preconditioner converges to an error of 10^{-16} after 16 iterations, while SCS and SCS^2 converge in 11 and 9 iterations respectively, and S(CS)^2 converges in just 7 iterations to an error of 10^{-17}. For $m = 20$, after 20 iterations SC two-level additive Schwarz in time preconditioner converges to an error of 10^{-13}, while SCS, S(CS)^2, and SCS^2 converge to much smaller errors, below 10^{-16}.

For the heat problem, Figure 2.6, on short time interval SCS, SCS^2 and S(CS)^2 converge faster than SC two-level additive Schwarz in time preconditioner. In particular for $m = 2$, SC converges to an error around 10^{-16} after 13 iterations, while SCS, SCS^2, and S(CS)^2 need 9, 6, and 5 iterations, respectively. For $m = 20$, the improvement becomes more important,

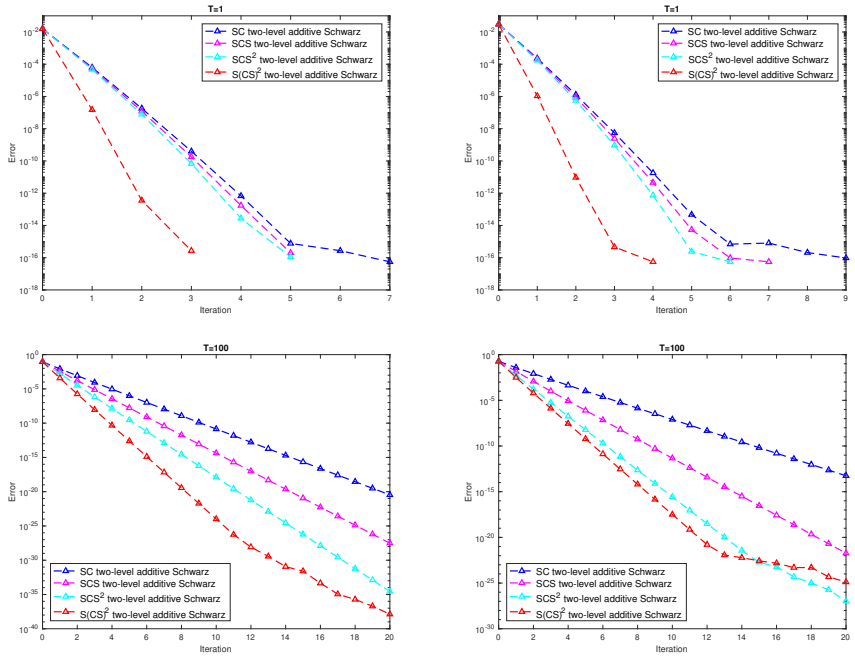

Figure 2.5: Error between approximate solution and fine sequential solution for Dahlquist problem with $m = 2$ (first column) and $m = 20$ (second column), $T = 1, N_C = 20$ (first row), and $T = 100, N_C = 100$ (second row).

specifically it takes 18 iterations for SC two-level additive Schwarz in time preconditioner to converge to an error of 10^{-16}, while SCS reaches this error in 10 iterations, and both SCS^2 and S(CS)^2 require only 7 iterations. We also observe that SCS^2 has a convergence rate close to S(CS)^2. On long time interval, both SCS, SCS^2 have a convergence rate close to the one of S(CS)^2, and SCS^2 converges faster than the other variants. In particular for $m = 2$, SC two-level additive Schwarz in time preconditioner converges to an error of 10^{-17} after 10 iterations, while it takes 4 iterations for SCS and 3 iterations for both SCS^2 and S(CS)^2 to converge to the same error. For $m = 20$, SCS^2 converges to an error around 10^{-17} after one iteration, SCS and S(CS)^2 converge to the same error in 2 iterations, while SC two-level additive Schwarz in time preconditioner requires 10 iterations.

For the advection-reaction-diffusion problem, Figure 2.7, the convergence behavior is similar to the heat equation for $m = 2$. For short time interval and $m = 20$, SC two-level additive Schwarz in time preconditioner converges to an error of 10^{-14} in 18 iterations, while SCS, SCS^2, and S(CS)^2 converge to the same error in 9, 7, and 6 iterations, respectively. On long time interval and $m = 20$, it takes 15 iterations for SC two-level additive Schwarz in time preconditioner to converge to an error of 10^{-16}, while SCS, S(CS)^2, and SCS^2 reach the same error in 4, 3, and 2 iterations, respectively.

In summary, the SC two-level additive Schwarz in time preconditioner with no additional coarse or fine propagation steps has a slower convergence than the other variants for all our test cases. This indicates that the usage of additional coarse or fine propagation steps leads to a more efficient preconditioner. The S(CS)^2 variant, corresponding to overlapping

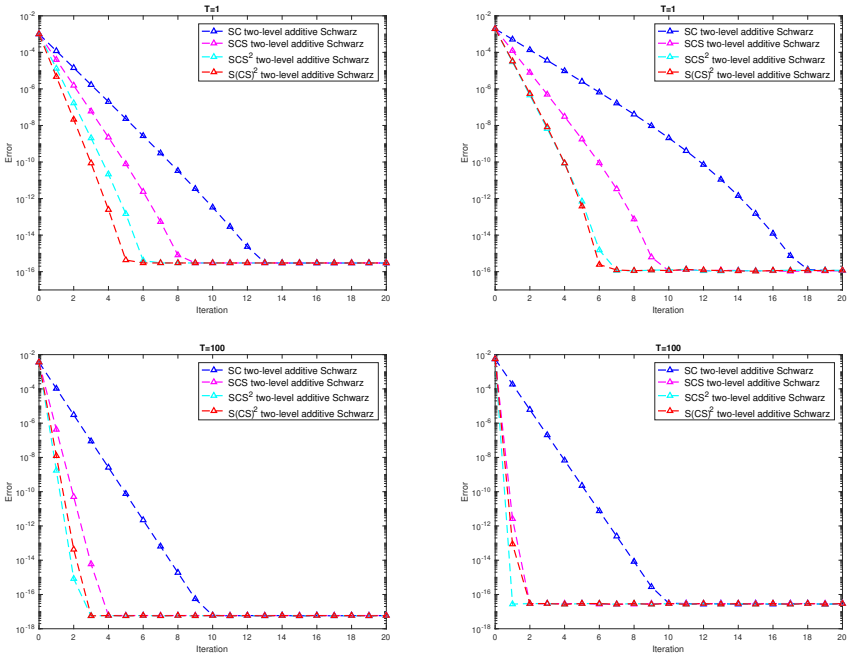

Figure 2.6: Error between approximate solution and fine sequential solution for heat equation with $m = 2$ (first column) and $m = 20$ (second column), $T = 1, N_C = 20$ (first row), and $T = 100, N_C = 100$ (second row).

parareal or MGRIT with F(CF)^2-relaxation, converges faster than the other variants in case of short time interval simulation. However, it costs 2 coarse propagation steps, which are computed sequentially, and hence in a parallel environment this leads to a less efficient algorithm. The SCS^2 variant converges faster than SCS for all our test cases. It is close to the convergence rate of S(CS)^2 for short time interval simulation, and it is even faster than S(CS)^2 for the heat equation (2.48) and the advection-reaction-diffusion equation (2.49) on long time interval. It is efficient when m increases, for example for $m = 20$, it reaches an error of 10^{-17} after one iteration. Since it has only one coarse propagation step, and the additional steps are performed in parallel, it allows to exploit efficiently a parallel computer.

2.6.3 Parareal with GMRES acceleration

In this section we discuss the results obtained by parareal with GMRES acceleration. The tolerance for GMRES is set to 10^{-16}. For Dahlquist problem, the 2-norm of the error between the approximate solution and the fine sequential solution and of the relative residual are displayed in Figure 2.8 for $N_C = 20, T = 1$, and for $N_C = 100, T = 100$. In both tests, $\Delta t = T/N_C, \delta t = \Delta t/m, m \in \{5, 20\}$. On short time interval, we observe that while the relative residual of parareal with GMRES acceleration is slightly smaller, the convergence rate of the error is not improved much. However, on long time interval, GMRES improves the convergence rate of parareal. For example for $m = 20$, parareal

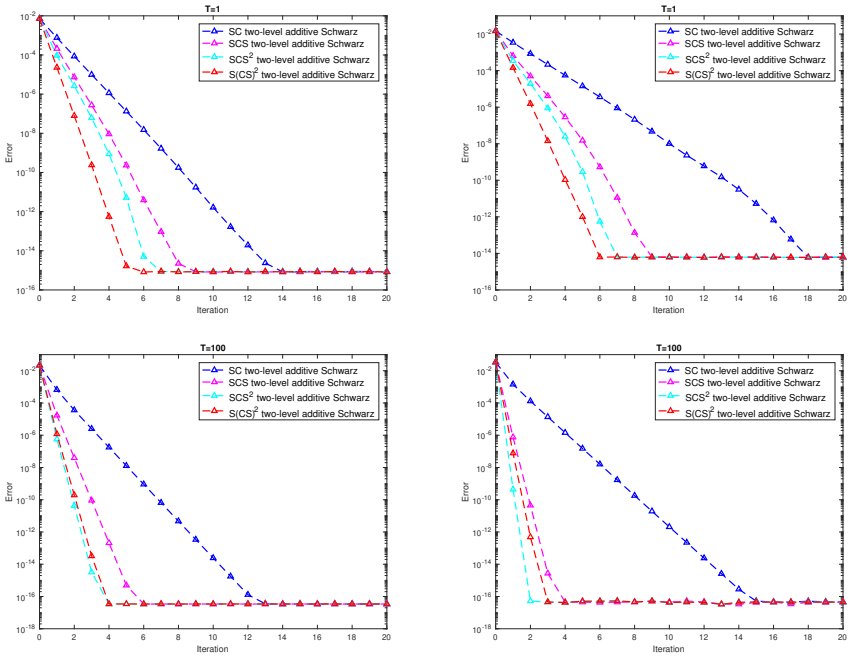

Figure 2.7: Error between approximate solution and fine sequential solution for advection-reaction-diffusion equation with $m = 2$ (first column) and $m = 20$ (second column), $T = 1, N_C = 20$ (first row), and $T = 100, N_C = 100$ (second row).

with GMRES acceleration converges to an error of 10^{-15} after 19 iterations, while parareal converges to an error of 10^{-13} after 20 iterations. For the relative residual, parareal with GMRES acceleration converges to 10^{-15} after 20 iterations, while parareal converges to 10^{-11} after the same numbers of iterations.

Since the convergence behavior of the error and the relative residual is similar for the heat equation and the advection-reaction-diffusion equation, we present only the convergence results for the latter equation. They are displayed in Figure 2.9 for short and long time intervals. On short time interval, we observe that the convergence rate of parareal with GMRES acceleration is slightly improved for both the error and the relative residual. Parareal with GMRES acceleration allows to reach the same error as parareal, while requiring 2 iterations less. For example for $m = 20$, parareal with GMRES acceleration converges to an error of 10^{-14} after 16 iterations, while parareal converges to the same error after 18 iterations. On long time interval, the improvement is even less important.

It can be seen that GMRES improves slightly the convergence of parareal for the three test cases as mentioned in the end of section 2.2.2.

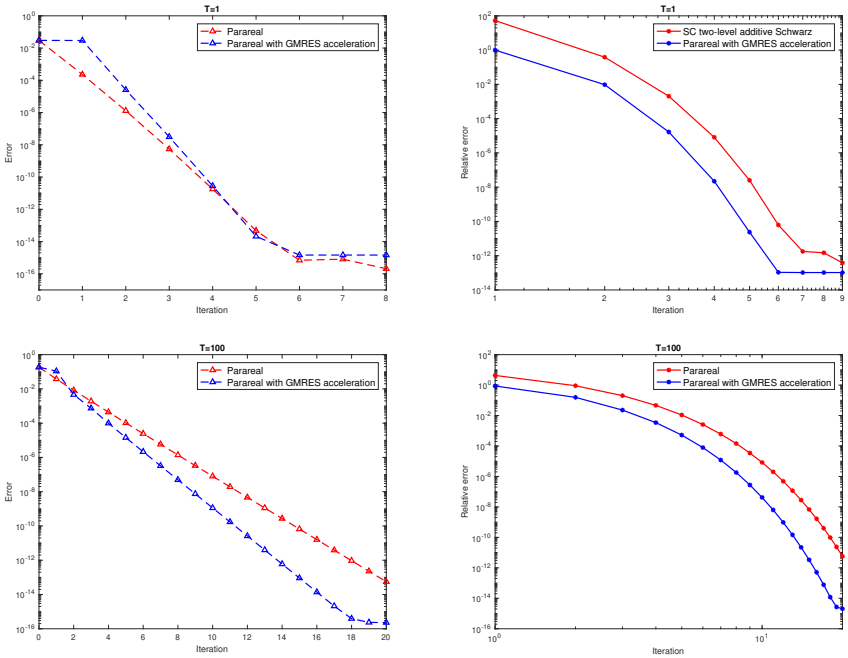

Figure 2.8: Error between approximate solution and fine sequential solution (first column) and relative residual (second column) in 2-norm for Dahlquist problem, $T = 1, N_C = 20$ (first row), $T = 100, N_C = 100$ (second row) with $m = 20$ in both cases.

2.6.4 Impact of GMRES acceleration for the advection-reaction-diffusion equation with different coefficients

We study in this section the convergence of parareal with GMRES acceleration for the advection-reaction-diffusion equation with different coefficients than in the beginning of section 2.6. We consider the following setting: $L = 1, T = 1$, $a = 0.01, b = 0.5, c = 100$, $N_C = 20, \Delta x \in \{0.2, 0.05\}$, $\Delta t = T/N_C, \delta t = \Delta t/m, m = 2$. The exact solution is $u_{exact} = \sin(2\pi x)\exp(-2t)$. The 2-norm of the error between the approximate solution and the fine sequential solution and of the relative residual are displayed in Figure 2.10. We observe that both parareal and parareal with GMRES acceleration converge within 20 iterations. However the error and relative residual of parareal seem to stagnate ($\Delta x \in \{0.2, 0.05\}$) and even increase ($\Delta x = 0.05$), while those of parareal with GMRES acceleration always decrease. Specifically for $\Delta x = 0.2$, parareal converges slowly within the first 5 iterations, then stagnates, and continues to converge after 16 iterations. Hence GMRES acceleration provides a more robust approach on short time interval $T = 1$. However, on long time interval $T = 100$, both methods converge with the same rate.

In this section we also present numerical experiments for the advection-reaction-diffusion equation in two cases, diffusion dominated and advection dominated. For both cases, we consider the advection-reaction-diffusion equation (2.49) with the periodic

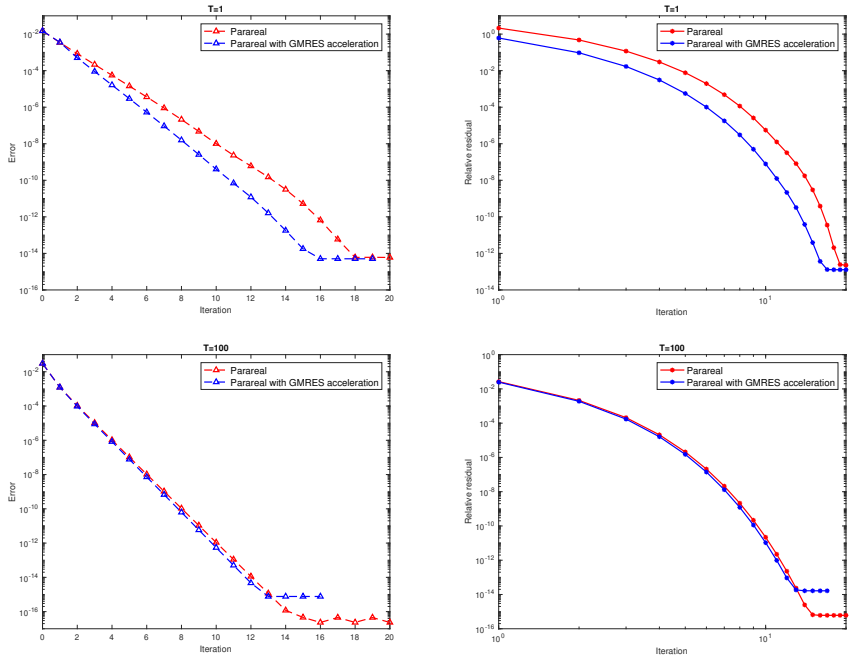

Figure 2.9: Error between approximate solution and fine sequential solution (first column) and relative residual (second column) in 2-norm for advection-reaction-diffusion equation, $T = 1, N_C = 20, m = 20$ (first row), and $T = 100, N_C = 100, m = 5$ (second row).

boundary condition

$$\begin{cases} u(0,t) &= u(L-\Delta x,t), \\ u(L,t) &= u(\Delta x,t), \end{cases}$$

with $L = 1, T = 1, N_C = 20, \Delta x = 0.025, \Delta t = T/N_C, \delta t = \Delta t/m, m = 5$ and the exact solution $u_{exact} = \sin(2\pi(x - bt))\exp(-2t)$. For the advective case, we consider $a = 0.0005, b = 1, c = 1$, and for the diffusive case, we consider $a = 1, b = 0.0005, c = 1$. The 2-norm of the error between the approximate solution and the fine sequential solution and of the relative residual are displayed in Figure 2.11. We observe that parareal with GMRES acceleration always converges faster than the plain parareal in both cases. In particular for the advective case, parareal converges to the error of 10^{-12} after 20 iterations while parareal with GMRES acceleration converges to the same error after 19 iterations. For the diffusive case, after 18 iterations, parareal with GMRES acceleration converges to the error of 10^{-9} while parareal only converges to the error of 10^{-6}. It can be seen that GMRES again improves slightly the convergence of parareal as the numerical results in section 2.6.3.

Additionally in the end of this section, we show the convergence behaviors of parareal and parareal with GMRES acceleration with a different method for the fine propagator. In particular, we keep using backward Euler in time for the coarse propagator but Runge-Kutta 4 for the fine propagator. For the discretization in space, we keep the same centered finite difference method for both coarse and fine propagators. Follow the same setting with the periodic boundary condition and $\Delta x = 0.1$, the convergence results are shown in

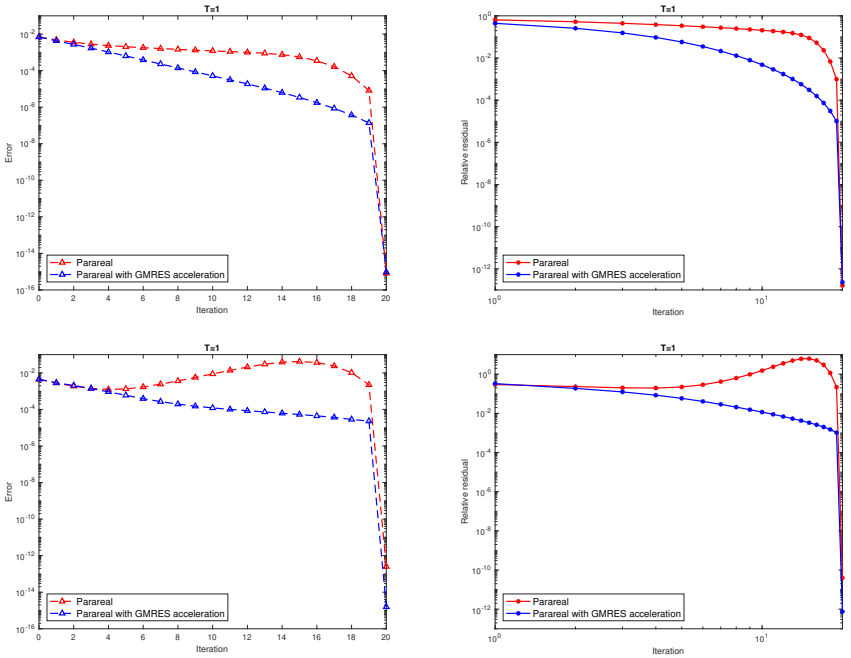

Figure 2.10: Error between approximate solution and fine sequential solution (first column) and relative residual (second column) in 2-norm for advection-reaction-diffusion equation with the Dirichlet boundary condition, $T = 1, N_C = 20$, $m = 2$, $\Delta x = 0.2$ (first row), and $\Delta x = 0.05$ (second row), with backward Euler for both propagators.

Figure 2.12. In particular for the advective case, parareal converges to the error of 10^{-14} after 20 iterations while parareal with GMRES acceleration converges to the same error after 18 iterations. For the diffusive case, we observe that both parareal and parareal with GMRES acceleration converge with a faster rate than the advective case. In particular parareal with GMRES acceleration needs 16 iterations to converge to the error of 10^{-14} while parareal needs 18 iterations to converge to the same error. We observe that parareal with GMRES acceleration slightly improves the convergence in both cases as the previous results in Figure 2.11. Moreover, with the more accurate discretization for the fine propagator Runge-Kutta 4, the convergence curves are slightly faster than the ones using backward Euler in Figure 2.11. Specifically in the diffusive case, parareal with GMRES acceleration converges to the error of 10^{-14} after 16 iterations while it needs 20 iterations to converge to the same error in case of using backward Euler for the fine propagator as it can be seen in Figure 2.11.

2.7 Conclusions and perspectives

In this chapter, we propose an interpretation of parareal algorithm based on a domain decomposition strategy, that we refer to as SC two-level additive Schwarz in time precon-

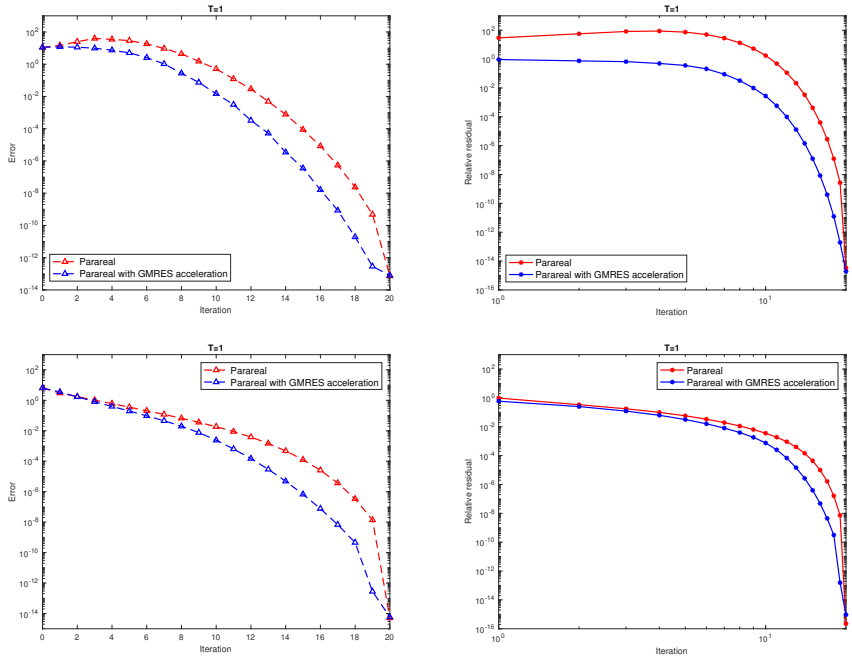

Figure 2.11: Error between approximate solution and fine sequential solution (first column) and relative residual (second column) in 2-norm for advection-reaction-diffusion equation with the periodic boundary condition, $T = 1$, $N_C = 20$, $m = 5$ for advective case (first row), and for diffusive case (second row), with backward Euler for both propagators.

ditioner. This preconditioner in time is equivalent to MGRIT with F-relaxation. We study variants of this preconditioner and show that additional fine or coarse propagation steps lead to MGRIT with FCF-relaxation, MGRIT with F(CF)^2-relaxation or overlapping parareal. We also find that SCS^2 two-level additive Schwarz in time preconditioner converges faster than MGRIT with F(CF)^2-relaxation or overlapping parareal on long time interval and with a large number of subdomains. This allows to exploit parallelism while having only one sequential coarse propagation step. Theoretical convergence bounds are verified and numerical results show that they are sharp especially for long time interval simulation. We also propose using Krylov subspace method, especially GMRES, to accelerate the parareal algorithm. We find that for a specific case of the advection-reaction-diffusion equation in which the advection and reaction coefficients are large compared to the diffusion term, the error of parareal stagnates or even increases for the first iterations, while GMRES provides a faster decrease of the error. This phenomena as well as the convergence analysis of parareal with GMRES acceleration will be studied in our future work.

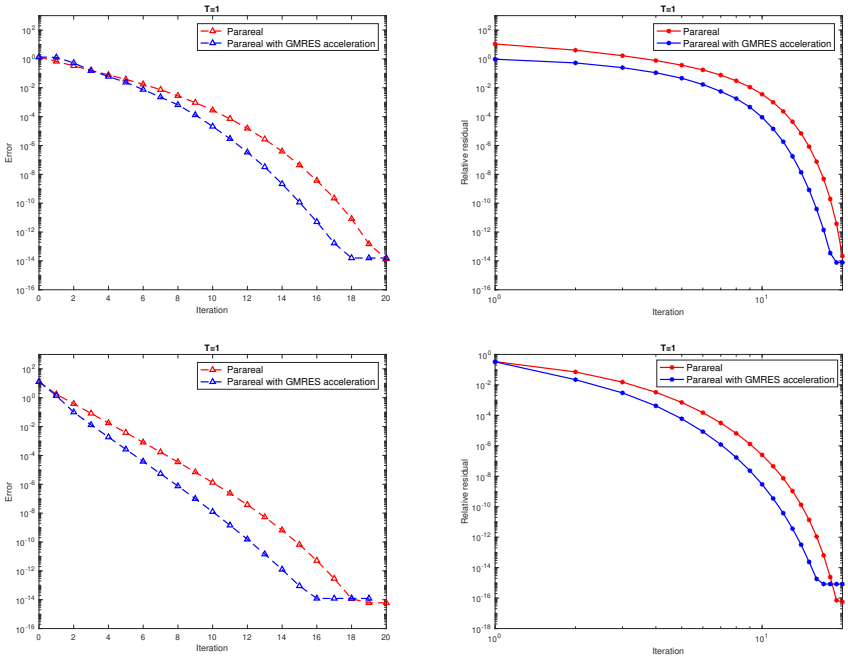

Figure 2.12: Error between approximate solution and fine sequential solution (first column) and relative residual (second column) in 2-norm for advection-reaction-diffusion equation with the periodic boundary condition, $T = 1, N_C = 20, m = 5$ for advective case (first row), and for diffusive case (second row), with backward Euler for the coarse propagator and Runge-Kutta 4 for the fine propagator.

3 Reduced Model-Based Parareal Simulations of Oscillatory Singularly Perturbed Ordinary Differential Equations

Contents

This chapter corresponds to the published journal paper, L. Grigori, S. A. Hirstoaga, V. T. Nguyen and J. Salomon. *Reduced model-based parareal simulations of oscillatory singularly perturbed ordinary differential equations.* In: *Journal of Computational Physics*, Volume 436, 1 July 2021, 110282, ISSN 0021-9991, https://doi.org/10.1016/j.jcp.2021.110282.
Acknowledgement: Laura Grigori, Van-Thanh Nguyen and Julien Salomon thank the funding by the French National Research Agency (ANR) Contract ANR-15-CE23-0019 (project CINE-PARA). The authors wish to express their gratitude to the anonymous referees for their valuable remarks.

Abstract

We propose a new strategy for solving by the parareal algorithm highly oscillatory ordinary differential equations which are characteristics of a six-dimensional Vlasov equation. For the coarse solvers we use reduced models, obtained from the two-scale asymptotic expansions in [28]. Such reduced models have a low computational cost since they are free of high oscillations. The parareal method allows to improve their accuracy in a few iterations through corrections by fine solvers of the full model. We demonstrate the accuracy and the efficiency of the strategy in numerical experiments of short time and long time simulations of charged particles submitted to a large magnetic field. In addition, the convergence of the parareal method is obtained uniformly with respect to the vanishing stiff parameter.

Keywords: parareal algorithm, two-scale expansion, multi-scale models, Vlasov characteristics, electric and magnetic fields.

3.1 Introduction

In this paper we propose a new coupling strategy in the parareal framework [68, 71] to efficiently solve the following six dimensional dynamical system for $0 < \varepsilon \ll 1$

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{x}_\varepsilon}{\mathrm{d}t} = \mathbf{v}_\varepsilon, & \mathbf{x}_\varepsilon(s) = \mathbf{x}, \\ \dfrac{\mathrm{d}\mathbf{v}_\varepsilon}{\mathrm{d}t} = \dfrac{1}{\varepsilon}\big(\mathbf{v}_\varepsilon \times \mathbf{B}_\varepsilon(\mathbf{x}_\varepsilon)\big) + \mathbf{E}(t, \mathbf{x}_\varepsilon), & \mathbf{v}_\varepsilon(s) = \mathbf{v}, \end{cases} \tag{3.1}$$

where $(\mathbf{x}, \mathbf{v})$ is an initial condition given at the initial time $t = s$. The system in (3.1) models the dynamics of a charged particle under the influence of an external electro-magnetic field. This is a typical characteristic curve of the Vlasov equation. In this context, $\mathbf{x}_\varepsilon : \mathbb{R} \to \mathbb{R}^3$ stands for the position unknown, $\mathbf{v}_\varepsilon : \mathbb{R} \to \mathbb{R}^3$ for the velocity unknown, and $\mathbf{E} : \mathbb{R} \times \mathbb{R}^3 \to \mathbb{R}^3$ and $\frac{1}{\varepsilon}\mathbf{B}_\varepsilon : \mathbb{R}^3 \to \mathbb{R}^3$ for a given electro-magnetic field. We assume $|\mathbf{B}_\varepsilon| = 1$ and that the mass and the charge particle are both equal to 1. The parameter $1/\varepsilon$ in front of the $\mathbf{v}_\varepsilon \times \mathbf{B}_\varepsilon(\mathbf{x}_\varepsilon)$ term means that the magnetic field is assumed high with respect to the electric term, in view of plasma confinement considerations [54]. More precisely, $1/\varepsilon$ denotes the strength of the magnetic field and thus, since the charge-to-mass ratio is assumed to be 1, the cyclotron frequency is also equal to $1/\varepsilon$. The difficulty of the problem is that the large magnetic field introduces a rapid time scale, the rotation of particles around the magnetic field line, which is much smaller than the one driven by the electric field. We are thus faced with a multi-scale problem whose numerical solution by standard methods requires high computational cost, since a standard but accurate enough numerical integrator requires time steps that are of the order of the fastest oscillation.

This is an issue to be avoided in applications, and therefore, in this paper we are interested in solving equations in (3.1) with a time step which is not constraint by ε.

The parareal algorithm is an efficient method performing real time simulations with the help of parallel computing, for the numerical solving of a very large class of time dependent equations. The literature is huge, we cite only [27, 34, 68, 71]. The method involves a fine expensive solver that is only applied in parallel, and a coarse but cheap solver which is used in sequential. A basic way to apply parareal in practice consists in taking large time steps Δt for a coarse solver and in refining the solutions in parallel using smaller time steps δt. This can reduce the computational time if the parareal iterations converge rapidly and if the ratio $\Delta t/\delta t$ is large.

However, when solving stiff equations like (3.1), regardless of the numerical scheme used for the coarse solver, the time step should satisfy $\Delta t \sim \varepsilon$ to achieve enough accuracy leading to a rapid convergence of the parareal scheme [37]. This is due to the high oscillations in time (with period of order ε) in the solution. Therefore, it can be interesting to use a different model to define the coarse solver in such a way that it remains computationally cheap but with a time step satisfying $\Delta t \gg \varepsilon$. Eventually, it is also important that the coarse solver be accurate enough so that the parareal iterations converge rapidly. In the case of equation (3.1), it is crucial for the coarse solver to provide an accurate approximation of the high oscillations, since otherwise the solver accumulates large errors, parareal requires a large number of iterations and thus the computational speed-up deteriorates. The purpose of our work is to obtain a convergent parareal algorithm with a large ratio $\Delta t/\delta t$ and a small number of iterations ($k \ll N$, see section 3.2 for notation).

In this paper, we use the parareal algorithm to efficiently integrate equation (3.1), by using a reduced model to define the coarse solver. Roughly speaking, such a reduced model reads

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{Y}}{\mathrm{d}t} = f(\mathbf{Y},\mathbf{U}), & \mathbf{Y}(s) = \mathbf{x}, \\ \dfrac{\mathrm{d}\mathbf{U}}{\mathrm{d}t} = g(\mathbf{Y},\mathbf{U}), & \mathbf{U}(s) = \mathbf{v}, \end{cases} \tag{3.2}$$

where $\mathbf{Y}$, $\mathbf{U}$ are used to approximate $\mathbf{x}_\varepsilon$, $\mathbf{v}_\varepsilon$ thanks to

$$\big(\mathbf{x}_\varepsilon(t), \mathbf{v}_\varepsilon(t)\big) \sim Z\Big((t-s)/\varepsilon, \big(\mathbf{Y}(t), \mathbf{U}(t)\big)\Big) \quad \text{when } \varepsilon \to 0,$$

and where Z is an operator for which an explicit form is to be derived in practice.
Specifically, we illustrate the idea above with an example in a similar context, as detailed in [53]. Thus, if instead of equation (3.1) we consider

$$\frac{\mathrm{d}\mathbf{u}_\varepsilon}{\mathrm{d}t} + \frac{1}{\varepsilon} L\mathbf{u}_\varepsilon = N(\mathbf{u}_\varepsilon), \quad \mathbf{u}_\varepsilon(0) = u_0,$$

where L is a linear operator and $N(\mathbf{u}_\varepsilon)$ is a specific nonlinear term, then it is well-known that under suitable assumptions, the solution $\mathbf{u}_\varepsilon(t)$ has the asymptotic approximation $\mathbf{u}_\varepsilon(t) = \exp(-\frac{t}{\varepsilon}L)\overline{\mathbf{u}}(t) + O(\varepsilon)$, where the slowly varying function $\overline{\mathbf{u}}$ is the solution to the reduced model

$$\frac{\mathrm{d}\overline{\mathbf{u}}}{\mathrm{d}t} = \overline{N}(\overline{\mathbf{u}}), \quad \overline{\mathbf{u}}(0) = u_0, \tag{3.3}$$

where the $\overline{N}(\overline{\mathbf{u}})$ term is obtained by time averaging

$$\overline{N}(\overline{\mathbf{u}}(t)) = \lim_{T\to\infty} \frac{1}{T} \int_0^T e^{\theta L} N(e^{-\theta L}\overline{\mathbf{u}}(t))d\theta.$$

It is important to note that $\overline{\mathbf{u}}(t)$ and its derivatives are formally bounded independently of ε and therefore, large time steps $\Delta t \gg \varepsilon$ can be taken to solve (3.3) (see [53]).

In our approach, the reduced model is obtained through a two-scale asymptotic expansion and is proved to provide an accurate approximation of the initial equation when the small parameter ε vanishes [28]. More precisely, we use either a zero-th order or a first order two-scale model, depending on the availability of practical equations. Indeed, it is possible that the first order model is too complex to be solved, analytically or numerically and in this case, only the zero-th order model will be considered. These models have two advantages: the low computational cost and the capability to give a good approximation of the high frequency oscillations through the operator Z enclosing the smallest scale, under the assumption that these oscillations are periodic and can be analytically computed [28].

The idea of using a different model for the coarse solver is not new. As an example, a similar approach has been used in chemical kinetics [71], where a reduction of a linear kinetic system with multiple scales was applied for the coarse solving. In [67], a slow manifold projector is used as coarse solver for solving ordinary differential equations with dissipative stiffness. We also mention two contributions closely related to our work. First, a parareal method for PDEs with linear high oscillating term is proposed in [53]. On the basis of a classical averaged model, the method needs exact knowledge of the fast variable to obtain a convergent parareal algorithm. Though our strategy also assumes the period of the fastest motion to be known, the high oscillating term is not necessarily linear in our case (see section 3.5). Additionally, we consider first-order asymptotic terms which provide a more accurate averaged model and can accelerate the convergence of the parareal method. Second, in the frame of models similar to (3.1), a multi-scale method for solving the slow evolution was successfully used as coarse solver in [3], without requiring explicit knowledge of the fast and slow variables. However a tedious alignment algorithm is required to achieve convergence of parareal. Specifically, the method needs to make the alignment of the fast phases of the coarse and fine solvers with sufficient efficiency and accuracy. On the contrary, our reduced model accurately synchronizes phase with the fine solution, which allows us to avoid such an alignment algorithm to get the numerical convergence of the parareal method. The drawback of our approach is that the small period of the fast oscillation must to be known. However, this particular framework covers several models which solve interesting problems in plasma physics, as illustrated in section 3.6.

The paper is organized as follows. In section 3.2, we briefly present the parareal method and describe our strategy as applied in the context of stiff ordinary differential equations. In section 3.3, we introduce the two-scale asymptotic expansion at the base of the reduced models and we justify their use as coarse solvers of our parareal algorithm. In sections 3.4 and 3.5, we present three ODE models which enter into the general form (3.1) and derive their first order and zero-th order reduced equations following [28]. The full equations under consideration apply to different models in plasma physics: isotope separation by

ion cyclotron resonance, storing of charged particles in a Penning trap, and an example of charged particle confinement by strong variable magnetic field. For these models, we present in section 3.6 numerical experiments that show that the parareal strategy provides accurate results together with computational efficiency through parallelism.

3.2 The parareal algorithm

Introduced in 2001 [68], the parareal (parallel in real time) algorithm displays its advantage by covering various fields of applications where it exploits very efficiently parallel computing over a large number of processors to solve problems in real time constraint context. Since its conception, the algorithm has been intensively analyzed [6, 33, 37, 97]. Let us briefly recall this approach. Consider the simple time dependent problem

$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u) \text{ in } (0,T),\ u(0) = \mathrm{u}_0. \tag{3.4}$$

The time interval $[0,T]$ is decomposed into N uniform time slices $[T_n, T_{n+1}]$, for $n \in \{0,\dots,N-1\}$. Let $\mathcal{F}(T_{n+1}, T_n, U_n)$ denote the fine solver, which gives a very accurate approximation of the solution at time T_{n+1} with the initial solution U_n at time T_n and let $\mathcal{G}(T_{n+1}, T_n, U_n)$ denote the coarse solver, which gives a coarse approximation of the solution at time T_{n+1} also with the initial solution U_n at time T_n. The coarse solver is to be chosen in such a way that, its cost is much lower than the one of the fine solver. A popular strategy consists in using the approximation method considered in the fine solver but with a larger time step [37]. Alternatively, one can use an approximation method with lower accuracy, or even use a different model from the original problem as long as it can give a reasonable coarse and fast approximation of the solution of the original problem [71].
In this paper, we follow the latter approach and focus on the idea of using a reduced model of the original problem for the coarse solver. For that reason, the coarse solver $\mathcal{G}(T_{n+1}, T_n, U_n)$ is always assigned to the solution of the reduced model (3.2) and the fine solver $\mathcal{F}(T_{n+1}, T_n, U_n)$ is always assigned to the (approximated) solution of the original problem (3.1). In addition, we let the coarse propagator perform a single time step per time slice $[T_n, T_{n+1}]$.
The parareal algorithm aims at computing a sequence $(U_n^k)_{k,n}$ of approximations of $u(T_n)$ for $n \in \{0,\dots,N\}$ for every k in the following way. At the first step, the initial approximation U_n^0 at coarse time points $0 = T_0 < T_1 < \cdots < T_N = T$ can be computed sequentially using the coarse solver that reads

$$U_{n+1}^0 = \mathcal{G}(T_{n+1}, T_n, U_n^0),\ U_0^0 = u_0,$$

and then for $k = 0, 1, \dots$ with $U_0^{k+1} = u_0$, the parareal algorithm computes a more accurate approximation

$$U_{n+1}^{k+1} = \mathcal{G}(T_{n+1}, T_n, U_n^{k+1}) + \mathcal{F}(T_{n+1}, T_n, U_n^k) - \mathcal{G}(T_{n+1}, T_n, U_n^k).$$

In this iteration, the terms $\mathcal{F}(T_{n+1}, T_n, U_n^k)$ have the largest computational cost. Therefore, all these fine computations could be performed in parallel over each interval $[T_n, T_{n+1}]$, the

main goal of parareal being to speed up the computing time. However, in order to achieve a real speed-up, the algorithm should converge in a number of iterations significantly smaller than the number of time intervals.

3.3 Two-scale asymptotic expansion

In this section, we summarize the principles and the main result of two-scale asymptotic expansion allowing to obtain reduced models. The equation (3.1) is a particular instance of the more general singularly perturbed dynamical system

$$\frac{\mathrm{d}\mathcal{X}_\varepsilon}{\mathrm{d}t} = \mathbf{a}(t, \mathcal{X}_\varepsilon) + \frac{1}{\varepsilon}\mathbf{b}(t, \mathcal{X}_\varepsilon), \qquad \mathcal{X}_\varepsilon(s) = \mathcal{X}, \tag{3.5}$$

where $\mathcal{X}_\varepsilon : \mathbb{R} \to \mathbb{R}^d$ and $\mathbf{a}$ and $\mathbf{b}$ are given fields satisfying suitable assumptions and s plays the role of the initial time. Following [28], we briefly recall the asymptotic two-scale expansion method in order to approximate the solution $\mathcal{X}_\varepsilon(t)$ when $\varepsilon \to 0$. Under regularity assumptions on $\mathbf{a}$ and $\mathbf{b}$ and assuming the solution $\mathbf{Z}(t; \theta, \mathbf{z})$ to equation

$$\frac{\mathrm{d}\mathbf{Z}}{\mathrm{d}\theta} = \mathbf{b}(t, \mathbf{Z}), \qquad \mathbf{Z}(t; 0, \mathbf{z}) = \mathbf{z} \tag{3.6}$$

to be periodic in θ, for every $t \in \mathbb{R}$ and every $\mathbf{z} \in \mathbb{R}^d$, it is proved in [28] that $\mathcal{X}_\varepsilon$ admits the following two-scale expansion in time

$$\mathcal{X}_\varepsilon(t) = \mathcal{X}^0\Big(t, \frac{t-s}{\varepsilon}\Big) + \varepsilon\,\mathcal{X}^1\Big(t, \frac{t-s}{\varepsilon}\Big) + \varepsilon^2\,\mathcal{X}^2\Big(t, \frac{t-s}{\varepsilon}\Big) + \dots \tag{3.7}$$

when $\varepsilon \to 0$ and where the functions $\mathcal{X}^i(t, \theta)$ are periodic in θ for every $i \in \mathbb{N}$. In this setting, ordinary differential equations characterizing the terms of the expansion (3.7) are derived in [28, Theorems 1.1 & 1.3]. In addition, strong convergence theorems are proved, justifying the approximation results asserting that, *e.g.*, at the zero-th order we have

$$\mathcal{X}_\varepsilon(t) \sim \mathcal{X}^0\Big(t, \frac{t-s}{\varepsilon}\Big), \quad \text{when } \varepsilon \to 0,$$

and at the first order,

$$\mathcal{X}_\varepsilon(t) \sim \mathcal{X}^0\Big(t, \frac{t-s}{\varepsilon}\Big) + \varepsilon\,\mathcal{X}^1\Big(t, \frac{t-s}{\varepsilon}\Big), \quad \text{when } \varepsilon \to 0.$$

For the sake of completeness, we give below the result concerning the two-scale limit model or the zero-th order approximation [28, Theorem 1.1] in the case of a six dimensional space ($d = 6$).

Theorem 1.1. *We assume that*[1] $\mathbf{a} \in \big(C^1_b(\mathbb{R} \times \mathbb{R}^6)\big)^6$ *and* $\mathbf{b} \in \big(C^2_b(\mathbb{R} \times \mathbb{R}^6)\big)^6$. *Assume also that the solution of* (3.6) *is* 2π*-periodic in* θ*, for every* $t \in \mathbb{R}$ *and every* $\mathbf{z} \in \mathbb{R}^6$. *Then, for*

[1] C^m_b stands for the space of continuous functions with bounded derivatives to the order m.

every initial condition $\mathcal{X} \in \mathbb{R}^6$*, every* $\varepsilon > 0$*, and every* $\Delta S > 0$*, the solution* $\mathcal{X}_\varepsilon$ *of* (3.5) *exists on* $[s, s + \Delta S]$*, is unique and satisfies*

$$\lim_{\varepsilon \to 0} \sup_{t \in [s, s+\Delta S]} \left| \mathcal{X}_\varepsilon(t) - \mathcal{X}^0\left(t, \frac{t-s}{\varepsilon}\right) \right| = 0, \tag{3.8}$$

where $|\cdot|$ *stands for the Euclidean norm on* $\mathbb{R}^6$ *and* $\mathcal{X}^0$ *satisfies*

$$\mathcal{X}^0(t, \theta) = \mathbf{Z}\big(t; \theta, \mathcal{Y}^0(t)\big) \tag{3.9}$$

and where $\mathcal{Y}^0$ *is the solution to*

$$\frac{\mathrm{d}\mathcal{Y}^0}{\mathrm{d}t} = \alpha(t, \mathcal{Y}^0), \qquad \mathcal{Y}^0(s) = \mathcal{X}, \tag{3.10}$$

with α *defined by*

$$\alpha(t, \mathcal{Y}) = \frac{1}{2\pi} \int_0^{2\pi} \{\nabla \mathbf{Z}(t; \theta, \mathcal{Y})\}^{-1} \left\{ \mathbf{a}(t, \mathbf{Z}(t; \theta, \mathcal{Y})) - \frac{\partial \mathbf{Z}}{\partial t}(t; \theta, \mathcal{Y}) \right\} d\theta.$$

Remark 3.3.1. *We remark that the limit model in* (3.10) *does not contain high oscillations in time so that cheap numerical schemes can be used to compute* $\mathcal{Y}^0$*. Then, when* $\mathbf{Z}$ *is known in* (3.6)*, we obtain the term* $\mathcal{X}^0$ *by* (3.9)*, as an approximation of the solution* $\mathcal{X}_\varepsilon$ *in the sense of* (3.8)*. Though obtained at a low computational cost, the approximation* $\mathcal{X}^0$ *still contains information about the high oscillations in the solution, through the operator* $\mathbf{Z}$*.*
These facts underline that the solution to the limit model given by (3.9)-(3.10) *is a good candidate for a coarse solving in the parareal framework.*

In the subsequent sections, we develop this framework for equations of the type of equation (3.1), by using the notation $\mathcal{X} = (\mathbf{x}, \mathbf{v})^T$, where, as in classical mechanics, $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)^T$ stands for the position vector and $\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)^T$ for the velocity vector. In this setting, it is important to note the particular form of the system (3.6). The solution $\mathbf{Z}$ captures only the rotation of the particle velocity following the magnetic field: $\mathbf{Z} = (\mathbf{x}_\mathbf{Z}, \mathbf{v}_\mathbf{Z})^T$ is the solution to

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{x}_\mathbf{Z}}{\mathrm{d}\theta} = 0, & \mathbf{x}_\mathbf{Z}(0) = \mathbf{x}, \\ \dfrac{\mathrm{d}\mathbf{v}_\mathbf{Z}}{\mathrm{d}\theta} = \mathbf{v}_\mathbf{Z} \times \mathbf{B}(\mathbf{x}_\mathbf{Z}), & \mathbf{v}_\mathbf{Z}(0) = \mathbf{v}. \end{cases}$$

This motion is assumed to be 2π-periodic in the theory we use. We denote in the sequel the cyclotron period in time by $P = 2\pi\varepsilon$ and the cyclotron frequency by $1/\varepsilon$, which are associated to the full system (3.1).

3.4 The case of a constant magnetic field

In this section we consider equation (3.1) provided with a constant magnetic field $\mathbf{B}_\varepsilon = \overrightarrow{e_1}$, where $\{\overrightarrow{e_1}, \overrightarrow{e_2}, \overrightarrow{e_3}\}$ is the frame of $\mathbb{R}^3$ and with a given external electric field. In this way, the

term $\mathbf{v}_\varepsilon \times \mathbf{B}_\varepsilon(\mathbf{x}_\varepsilon)$ in the velocity equation of (3.1) writes $(\mathbf{v}_\varepsilon)^\perp = (0, (\mathbf{v}_\varepsilon)_3, -(\mathbf{v}_\varepsilon)_2)^T$. Thus, we can see that the basic assumption of periodicity of the solution of (3.6) is satisfied. The common feature of the test cases we treat in this section is that we can compute analytically the solutions of equation (3.1) and of the corresponding reduced model. Therefore, when applying the parareal algorithm we will be able to use the exact flows for the fine and the coarse solvers.

3.4.1 A uniform time varying electric field

In this section, we take an electric field which is only highly oscillating in time. In this case, system (3.1) writes

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{x}_\varepsilon}{\mathrm{d}t} = \mathbf{v}_\varepsilon, & \mathbf{x}_\varepsilon(s) = \mathbf{x}, \\ \dfrac{\mathrm{d}\mathbf{v}_\varepsilon}{\mathrm{d}t} = \dfrac{1}{\varepsilon}(\mathbf{v}_\varepsilon)^\perp + \mathbf{E}\Big(\dfrac{t}{\varepsilon}\Big), & \mathbf{v}_\varepsilon(s) = \mathbf{v}, \end{cases} \tag{3.11}$$

where $\mathbf{E}$ has the form $\mathbf{E}(\tau) = \big(E_1, E_2(\tau), E_3(\tau)\big)^T$, with $E_1 \in \mathbb{R}$ and E_2, E_3 are 2π-periodic functions, see [28, Section 3.1]. This system can be used for modelling ion cyclotron resonance with application in isotope separation in plasmas, see [29] and the references therein. In magnetized plasmas, the ions are heated by an oscillating perpendicular electric field at frequencies corresponding to the ion cyclotron frequency. Thus, the cyclotron resonance leads to a growth of the amplitude of motion in time. In the sequel, we consider for illustration the following electric field

$$E_1 = 0, \quad E_2(\tau) = \sin(\tau), \quad E_3(\tau) = \cos(\tau). \tag{3.12}$$

However, all the following results can be derived in a similar form for a general electric field with the above properties. Next, we need the following matrices denoted by

$$P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, R(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{pmatrix}, \mathcal{R}(\theta) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \sin\theta & 1-\cos\theta \\ 0 & \cos\theta - 1 & \sin\theta \end{pmatrix}. \tag{3.13}$$

It is convenient to put the solution of (3.11)-(3.12) in the form

$$\begin{pmatrix} \mathbf{x}_\varepsilon(t) \\ \mathbf{v}_\varepsilon(t) \end{pmatrix} = \mathcal{A} \begin{pmatrix} \mathbf{x} \\ \mathbf{v} \end{pmatrix} + \mathcal{B}, \tag{3.14}$$

where the 6×6 matrix $\mathcal{A}$ and the vector $\mathcal{B}$ are given by

$$\mathcal{A} = \begin{pmatrix} I_3 & (t-s)P + \varepsilon\mathcal{R}\big(\frac{t-s}{\varepsilon}\big) \\ O_3 & R\big(\frac{t-s}{\varepsilon}\big) \end{pmatrix} \text{ and}$$

$$\mathcal{B} = \begin{pmatrix} \varepsilon(t-s)\big(0, -\cos(t/\varepsilon), \sin(t/\varepsilon)\big)^T + \varepsilon^2\big(0, \sin(t/\varepsilon) - \sin(s/\varepsilon), \cos(t/\varepsilon) - \cos(s/\varepsilon)\big)^T \\ (t-s)\big(0, \sin(t/\varepsilon), \cos(t/\varepsilon)\big)^T \end{pmatrix},$$

with I_3 the 3×3 identity matrix and O_3 the 3×3 zero matrix.

Next, we derive the reduced model for equation (3.11). We apply [28, Theorems 3.1, 3.2] to equations (3.11)-(3.12) to obtain the first order two-scale model. The approximation of the solution is

$$\mathbf{G}(t) = \begin{pmatrix} \mathbf{x}^0\Big(t, \dfrac{t-s}{\varepsilon}\Big) \\ \mathbf{v}^0\Big(t, \dfrac{t-s}{\varepsilon}\Big) \end{pmatrix} + \varepsilon \begin{pmatrix} \mathbf{x}^1\Big(t, \dfrac{t-s}{\varepsilon}\Big) \\ \mathbf{v}^1\Big(t, \dfrac{t-s}{\varepsilon}\Big) \end{pmatrix}, \tag{3.15}$$

where the terms in the expansion are given by

$$\begin{pmatrix} \mathbf{x}^0(t,\theta) \\ \mathbf{v}^0(t,\theta) \end{pmatrix} = \begin{pmatrix} \mathbf{y}^0(t) \\ R(\theta)\,\mathbf{u}^0(t) \end{pmatrix} \tag{3.16}$$

and

$$\begin{pmatrix} \mathbf{x}^1(t,\theta) \\ \mathbf{v}^1(t,\theta) \end{pmatrix} = \begin{pmatrix} \mathbf{y}^1(t) + \mathcal{R}(\theta)\,\mathbf{u}^0(t) \\ R(\theta)\,\mathbf{u}^1(t) + R(\theta)\Big(\displaystyle\int_0^\theta d\sigma - \frac{\theta}{2\pi}\int_0^{2\pi} d\sigma\Big)\big(R(-\sigma)\mathbf{E}(\sigma)\big) \end{pmatrix}. \tag{3.17}$$

Then, in the particular case of the electric field in (3.12), we have that $\big(\mathbf{y}^0(t), \mathbf{u}^0(t)\big)$ is solution to

$$\frac{\mathrm{d}\mathbf{y}^0}{\mathrm{d}t} = \begin{pmatrix} (\mathbf{u}^0)_1 \\ 0 \\ 0 \end{pmatrix}, \frac{\mathrm{d}\mathbf{u}^0}{\mathrm{d}t} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \text{ and } \begin{cases} \mathbf{y}^0(s) = \mathbf{x}, \\ \mathbf{u}^0(s) = \mathbf{v}, \end{cases} \tag{3.18}$$

with $(\mathbf{x}, \mathbf{v})$ the initial condition in (3.11) and that $\big(\mathbf{y}^1(t), \mathbf{u}^1(t)\big)$ is solution to

$$\frac{\mathrm{d}\mathbf{y}^1}{\mathrm{d}t} = \begin{pmatrix} (\mathbf{u}^1)_1 \\ -1 \\ 0 \end{pmatrix}, \frac{\mathrm{d}\mathbf{u}^1}{\mathrm{d}t} = 0 \text{ and } \begin{cases} \mathbf{y}^1(s) = 0, \\ \mathbf{u}^1(s) = 0. \end{cases} \tag{3.19}$$

Equations (3.18)-(3.19) are easy to solve, their solutions are

$$\begin{cases} \mathbf{y}^0(t) = \big(\mathbf{v}_1(t-s) + \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\big)^T, \\ \mathbf{u}^0(t) = \big(\mathbf{v}_1, \mathbf{v}_2, (t-s) + \mathbf{v}_3\big)^T, \end{cases}$$

and respectively

$$\begin{cases} \mathbf{y}^1(t) = \big(0, -(t-s), 0\big)^T, \\ \mathbf{u}^1(t) = \big(0, 0, 0\big)^T. \end{cases}$$

Replacing these formulas in (3.16)-(3.17) and getting the result in (3.15) we obtain the analytical form of the first-order two-scale approximation $\mathbf{G}(t)$. However, it is interesting to write $\mathbf{G}$ as the solution of the original system was derived in equation (3.14). We have

$$\mathbf{G}(t) = \mathcal{A}\begin{pmatrix} \mathbf{x} \\ \mathbf{v} \end{pmatrix} + C, \tag{3.20}$$

where the matrix $\mathcal{A}$ is as in (3.14) and C is given by

$$C = \begin{pmatrix} \varepsilon(t-s)\big(0, -\cos((t-s)/\varepsilon), \sin((t-s)/\varepsilon)\big)^T \\ (t-s)\big(0, \sin((t-s)/\varepsilon), \cos((t-s)/\varepsilon)\big)^T \end{pmatrix}.$$

Remark 3.4.1. *We notice that in the general case where* $\mathbf{E}$ *has the form*

$$\mathbf{E}(\tau) = \big(E_1, E_2(\tau), E_3(\tau)\big)^T, \quad \textit{with } E_1 \in \mathbb{R} \textit{ and } 2\pi\textit{-periodic functions } E_2, E_3,$$

the solutions of the full model and of the reduced one keep similar expressions to those in (3.14) *and* (3.20) *respectively. More precisely, the matrix* $\mathcal{A}$ *will be the same, the difference appearing in the vectors* $\mathcal{B}$ *and* C *which will contain averages in the fast variable against* $\sin(\cdot)$ *and* $\cos(\cdot)$ *of the functions* E_2 *and* E_3.

3.4.2 A non uniform stationary electric field

In this part, we consider an electric field which is not dependent of time but of space and we use the framework in [28, Section 3.2]. In this case, system (3.1) writes

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{x}_\varepsilon}{\mathrm{d}t} = \mathbf{v}_\varepsilon, & \mathbf{x}_\varepsilon(s) = \mathbf{x}, \\ \dfrac{\mathrm{d}\mathbf{v}_\varepsilon}{\mathrm{d}t} = \dfrac{1}{\varepsilon}(\mathbf{v}_\varepsilon)^\perp + \mathbf{E}(\mathbf{x}_\varepsilon), & \mathbf{v}_\varepsilon(s) = \mathbf{v}. \end{cases} \tag{3.21}$$

Then, following a standard strategy we can find an explicit form of a linear application $\mathbf{E}$ leading to highly oscillating solution but which is bounded in time. Nevertheless, by taking in (3.21) the electric field given by

$$\mathbf{E}(\mathbf{x}) = c\begin{pmatrix} -\mathbf{x}_1 \\ \mathbf{x}_2/2 \\ \mathbf{x}_3/2 \end{pmatrix}, \tag{3.22}$$

with an arbitrary constant $c > 0$, the system describes the dynamics of a charged particle in an ideal Penning trap [65] (we fix to 1 both the charge and the mass of the particle). Under the condition $\varepsilon < \sqrt{1/(2c)}$, the solution of (3.21)-(3.22) is

$$\begin{aligned} \mathbf{x}_\varepsilon(t) =& \begin{pmatrix} c_1\cos(\sqrt{c}\,(t-s)) + c_2\sin(\sqrt{c}\,(t-s)) \\ a_1\sin(a_\varepsilon(t-s)) - a_2\cos(a_\varepsilon(t-s)) + b_1\sin(b_\varepsilon(t-s)) - b_2\cos(b_\varepsilon(t-s)) \\ a_1\cos(a_\varepsilon(t-s)) + a_2\sin(a_\varepsilon(t-s)) + b_1\cos(b_\varepsilon(t-s)) + b_2\sin(b_\varepsilon(t-s)) \end{pmatrix}, \\ \mathbf{v}_\varepsilon(t) =& \frac{\mathrm{d}\mathbf{x}_\varepsilon}{\mathrm{d}t}(t), \end{aligned} \tag{3.23}$$

where

$$a_\varepsilon = \frac{1+\sqrt{1-2c\varepsilon^2}}{2\varepsilon}, \quad b_\varepsilon = \frac{1-\sqrt{1-2c\varepsilon^2}}{2\varepsilon}, \tag{3.24}$$

and $a_1, a_2, b_1, b_2, c_1, c_2$ are constants to be found from the initial condition.

Remark 3.4.2.

A Penning trap is a device for storing charged particles using a homogeneous magnetic field and an inhomogeneous quadrupole electric field. The constant c in (3.22) *entails the geometry of the trap and the voltage between the electrodes, while* $1/\varepsilon$ *is the magnitude of the magnetic field. The condition for having a stable periodic trajectory [65] is*

$$\frac{1}{\varepsilon} > \sqrt{2c}. \tag{3.25}$$

Otherwise, the particle escapes from the trap due to a magnetic field which is weaker than the electric field. This corresponds to a solution with growing amplitude of motion in time.

2. *We notice that the three frequencies* $\sqrt{c}$, a_ε, *and* b_ε *are denoted in literature [65] by* ω_x, ω_+, *and* ω_- *respectively, and they verify the relation*

$$\omega_\pm = \frac{1}{2}\left(\omega_{cy} \pm \sqrt{\omega_{cy}^2 - 2\omega_x^2}\right),$$

where ω_{cy} *is the cyclotron frequency. In our notation* $\omega_{cy} = 1/\varepsilon$.

3. *It is clear that the motion in the* $\overrightarrow{e_1}$ *direction is decoupled from the motion in the other two directions. More precisely, a charged particle performs in an ideal Penning trap three independent motions with characteristic frequencies: a modified cyclotron motion (at frequency* ω_+*), the axial motion (at frequency* ω_x*), and the magnetron motion or the* $\mathbf{E}\times\mathbf{B}$ *drift (at frequency* ω_-*).*

4. *We have* $a_\varepsilon \sim \frac{1}{\varepsilon}$ *and* $b_\varepsilon \sim \varepsilon$ *when* $\varepsilon \longrightarrow 0$. *Therefore, the solution in* (3.23) *oscillates in time at three scales,* $2\pi\varepsilon$, 1 *and* $2\pi/\varepsilon$. *In addition, we can identify initial conditions leading to solutions which are oscillating at the desired scale(s) by equating to zero the corresponding coefficients.*

Next, we derive the reduced model for equation (3.21). More precisely, we apply [28, Theorem 3.3] to write the specific first order two-scale model to the system (3.21)-(3.22). Recalling the formula in [28, Theorem 3.3], the first-order approximation of the solution to the model (3.21)-(3.22) is given by

$$\mathbf{G}(t) = \begin{pmatrix} \mathbf{x}^0\left(t, \dfrac{t-s}{\varepsilon}\right) \\ \mathbf{v}^0\left(t, \dfrac{t-s}{\varepsilon}\right) \end{pmatrix} + \varepsilon \begin{pmatrix} \mathbf{x}^1\left(t, \dfrac{t-s}{\varepsilon}\right) \\ \mathbf{v}^1\left(t, \dfrac{t-s}{\varepsilon}\right) \end{pmatrix}, \tag{3.26}$$

where, as in section 3.4.1, the terms in the expansion are given by

$$\begin{pmatrix} \mathbf{x}^0(t,\theta) \\ \mathbf{v}^0(t,\theta) \end{pmatrix} = \begin{pmatrix} \mathbf{y}^0(t) \\ R(\theta)\,\mathbf{u}^0(t) \end{pmatrix} \tag{3.27}$$

and

$$\begin{pmatrix} \mathbf{x}^1(t,\theta) \\ \mathbf{v}^1(t,\theta) \end{pmatrix} = \begin{pmatrix} \mathbf{y}^1(t) + \mathcal{R}(\theta)\,\mathbf{u}^0(t) \\ R(\theta)\,\mathbf{u}^1(t) + \mathcal{R}(\theta)\,\mathbf{E}(\mathbf{y}^0(t)) \end{pmatrix}. \tag{3.28}$$

Then, in the particular case of the electric field in (3.22), we have that $\big(\mathbf{y}^0(t), \mathbf{u}^0(t)\big)$ is solution to

$$\frac{\mathrm{d}\mathbf{y}^0}{\mathrm{d}t} = \begin{pmatrix} (\mathbf{u}^0)_1 \\ 0 \\ 0 \end{pmatrix}, \frac{\mathrm{d}\mathbf{u}^0}{\mathrm{d}t} = \begin{pmatrix} -c(\mathbf{y}^0)_1 \\ 0 \\ 0 \end{pmatrix} \text{ and } \begin{cases} \mathbf{y}^0(s) = \mathbf{x}, \\ \mathbf{u}^0(s) = \mathbf{v}, \end{cases} \tag{3.29}$$

with $(\mathbf{x}, \mathbf{v})$ the initial condition in (3.21) and that $\big(\mathbf{y}^1(t), \mathbf{u}^1(t)\big)$ is solution to

$$\frac{\mathrm{d}\mathbf{y}^1}{\mathrm{d}t} = \begin{pmatrix} (\mathbf{u}^1)_1 \\ \frac{c}{2}(\mathbf{y}^0)_3 \\ -\frac{c}{2}(\mathbf{y}^0)_2 \end{pmatrix}, \frac{\mathrm{d}\mathbf{u}^1}{\mathrm{d}t} = \begin{pmatrix} -c(\mathbf{y}^1)_1 \\ -\frac{c}{2}(\mathbf{u}^0)_3 \\ \frac{c}{2}(\mathbf{u}^0)_2 \end{pmatrix} \text{ and } \begin{cases} \mathbf{y}^1(s) = 0, \\ \mathbf{u}^1(s) = 0. \end{cases} \tag{3.30}$$

Equations (3.29)-(3.30) are easy to solve, their solutions are

$$\begin{cases} \mathbf{y}^0(t) = \big(\mathbf{x}_1 \cos(\sqrt{c}\,(t-s)) + \frac{\mathbf{v}_1}{\sqrt{c}} \sin(\sqrt{c}\,(t-s)),\ \mathbf{x}_2,\ \mathbf{x}_3\big)^T, \\ \mathbf{u}^0(t) = \big(-\mathbf{x}_1\sqrt{c}\sin(\sqrt{c}\,(t-s)) + \mathbf{v}_1\cos(\sqrt{c}\,(t-s)),\ \mathbf{v}_2,\ \mathbf{v}_3\big)^T, \end{cases}$$

and respectively

$$\begin{cases} \mathbf{y}^1(t) = \big(0,\ \frac{c}{2}\mathbf{x}_3(t-s),\ -\frac{c}{2}\mathbf{x}_2(t-s)\big)^T, \\ \mathbf{u}^1(t) = \big(0,\ -\frac{c}{2}\mathbf{v}_3(t-s),\ \frac{c}{2}\mathbf{v}_2(t-s)\big)^T. \end{cases}$$

Replacing (3.27)-(3.28) in (3.26), we obtain

$$\mathbf{G}(t) = \begin{pmatrix} \mathbf{y}^0(t) \\ R(\frac{t-s}{\varepsilon})\mathbf{u}^0(t) \end{pmatrix} + \varepsilon \begin{pmatrix} \mathbf{y}^1(t) + \mathcal{R}(\frac{t-s}{\varepsilon})\mathbf{u}^0(t) \\ R(\frac{t-s}{\varepsilon})\mathbf{u}^1(t) + \mathcal{R}(\frac{t-s}{\varepsilon})\mathbf{E}(\mathbf{y}^0(t)) \end{pmatrix}, \tag{3.31}$$

and thus, getting the analytic expressions of $\mathbf{y}^0, \mathbf{u}^0, \mathbf{y}^1, \mathbf{u}^1, \mathbf{E}$ and of matrices R and $\mathcal{R}$ in the above formula leads to the analytic form of the approximation $\mathbf{G}(t)$ to the solution $\big(\mathbf{x}_\varepsilon(t), \mathbf{v}_\varepsilon(t)\big)$ when ε is small enough and at any time $t \in [s, s+\Delta S]$. The obtained formula will be used in section 3.6 for the coarse solver.

3.5 The case of a variable magnetic field

In this section we study the case of a magnetic field with a strong part which is variable and a bounded part which is constant (see [28, Section 3.4]). In addition, we restrict to the case without electric field. More precisely, we consider equation (3.1) in the form

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{x}_\varepsilon}{\mathrm{d}t} = \mathbf{v}_\varepsilon, & \mathbf{x}_\varepsilon(s) = \mathbf{x}, \\ \dfrac{\mathrm{d}\mathbf{v}_\varepsilon}{\mathrm{d}t} = \dfrac{1}{\varepsilon}\big(\mathbf{v}_\varepsilon \times \mathcal{M}(\mathbf{x}_\varepsilon)\big) + \mathbf{v}_\varepsilon \times \overrightarrow{e_3}, & \mathbf{v}_\varepsilon(s) = \mathbf{v}, \end{cases} \tag{3.32}$$

where

$$\mathcal{M}(\mathbf{x}) = \frac{1}{\sqrt{\mathbf{x}_1{}^2 + \mathbf{x}_2{}^2}} \begin{pmatrix} -\mathbf{x}_2 \\ \mathbf{x}_1 \\ 0 \end{pmatrix}.$$

We first notice that the assumption on the 2π-periodicity of the solution $\mathbf{Z}$ to equation (3.6) is satisfied. Then, unlike the test cases in section 3.4, we do not have an analytic expression for the solution of equation (3.32). The reduced model we will use in the parareal method for this case, is the two-scale limit and not the first order approximation. The reason is that using the first order term in the asymptotic expansion becomes almost impossible due to its complex form (see [28, Theorem 3.6 & Appendix A]).

Next, we detail the two-scale limit model approximating equation (3.32) when $\varepsilon \longrightarrow 0$. Following [28, Theorem 3.6], the limit term in the expansion is given by

$$\begin{pmatrix} \mathbf{x}^0(t,\theta) \\ \mathbf{v}^0(t,\theta) \end{pmatrix} = \begin{pmatrix} \mathbf{y}^0(t) \\ \mathcal{Z}_{\mathbf{v}}\big(t;\theta,\mathbf{y}^0(t),\mathbf{u}^0(t)\big) \end{pmatrix},$$

where the components of $\mathcal{Z}(t;\theta,\mathbf{x},\mathbf{v}) = \big(\mathcal{Z}_{\mathbf{x}}(t;\theta,\mathbf{x},\mathbf{v}), \mathcal{Z}_{\mathbf{v}}(t;\theta,\mathbf{x},\mathbf{v})\big)^T$ are $\mathcal{Z}_{\mathbf{x}}(t;\theta,\mathbf{x},\mathbf{v}) = \mathbf{x}$ and $\mathcal{Z}_{\mathbf{v}}(t;\theta,\mathbf{x},\mathbf{v}) = C(\theta,\mathbf{x})\mathbf{v}$, with

$$C(\theta,\mathbf{x}) = \begin{pmatrix} \frac{\mathbf{x}_1{}^2\cos\theta+\mathbf{x}_2{}^2}{\mathbf{x}_1{}^2+\mathbf{x}_2{}^2} & \frac{\mathbf{x}_1\mathbf{x}_2(\cos\theta-1)}{\mathbf{x}_1{}^2+\mathbf{x}_2{}^2} & -\frac{\mathbf{x}_1\sin\theta}{\sqrt{\mathbf{x}_1{}^2+\mathbf{x}_2{}^2}} \\ \frac{\mathbf{x}_1\mathbf{x}_2(\cos\theta-1)}{\mathbf{x}_1{}^2+\mathbf{x}_2{}^2} & \frac{\mathbf{x}_2{}^2\cos\theta+\mathbf{x}_1{}^2}{\mathbf{x}_1{}^2+\mathbf{x}_2{}^2} & -\frac{\mathbf{x}_2\sin\theta}{\sqrt{\mathbf{x}_1{}^2+\mathbf{x}_2{}^2}} \\ \frac{\mathbf{x}_1\sin\theta}{\sqrt{\mathbf{x}_1{}^2+\mathbf{x}_2{}^2}} & \frac{\mathbf{x}_2\sin\theta}{\sqrt{\mathbf{x}_1{}^2+\mathbf{x}_2{}^2}} & \cos\theta \end{pmatrix}$$

and where $\big(\mathbf{y}^0(t),\mathbf{u}^0(t)\big)$ is solution to

$$\frac{\mathrm{d}\mathbf{y}^0}{\mathrm{d}t} = \overline{A}(\mathbf{y}^0)\mathbf{u}^0, \quad \frac{\mathrm{d}\mathbf{u}^0}{\mathrm{d}t} = \overline{\beta}(\mathbf{y}^0,\mathbf{u}^0) \quad \text{and} \quad \begin{cases} \mathbf{y}^0(s) = \mathbf{x}, \\ \mathbf{u}^0(s) = \mathbf{v}, \end{cases} \tag{3.33}$$

with $(\mathbf{x},\mathbf{v})$ the initial condition in (3.32) and with

$$\overline{A}(\mathbf{y}) = \frac{1}{\mathbf{y}_1{}^2+\mathbf{y}_2{}^2}\begin{pmatrix} \mathbf{y}_2{}^2 & -\mathbf{y}_1\mathbf{y}_2 & 0 \\ -\mathbf{y}_1\mathbf{y}_2 & \mathbf{y}_1{}^2 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \ \overline{\beta}(\mathbf{y},\mathbf{u}) = \begin{pmatrix} \frac{\mathbf{u}_2(\mathbf{u}_1\mathbf{y}_2-\mathbf{u}_2\mathbf{y}_1)}{\mathbf{y}_1{}^2+\mathbf{y}_2{}^2} \\ \frac{\mathbf{u}_1(\mathbf{u}_2\mathbf{y}_1-\mathbf{u}_1\mathbf{y}_2)}{\mathbf{y}_1{}^2+\mathbf{y}_2{}^2} \\ 0 \end{pmatrix}.$$

Thus, in this case, the approximation $\mathbf{G}(t)$ to the solution $\big(\mathbf{x}_\varepsilon(t),\mathbf{v}_\varepsilon(t)\big)$ when ε is small enough, is obtained first by solving the system (3.33) and then

$$\mathbf{G}(t) = \begin{pmatrix} \mathbf{y}^0(t) \\ C\big(\frac{t-s}{\varepsilon},\mathbf{y}^0(t)\big)\mathbf{u}^0(t) \end{pmatrix}. \tag{3.34}$$

3.6 Numerical results

First, in section 3.6.1 we analyze the time interval of validity and the accuracy of the reduced models for each test case. Then, we present numerical experiments illustrating the convergence of the parareal algorithm. In all the cases we consider, we obtained the numerical convergence with a number of parareal iterations K much smaller than the number N of the time slices of the interval $[0,T]$.

The reduced models that we use are zero-th or first order approximations of the initial stiff equation until a final time of order 1. The parareal algorithm allows us to perform simulations in long times, of order $1/\varepsilon$ or larger, by using the reduced model on time intervals where the latter is proved to be valid. For each case, we study the convergence of the algorithm when ε is fixed and also when making the parameter ε vanishing. This last issue is meaningful from the application viewpoint, since in realistic plasma physics phenomena such parameters are not fixed to a single value during the simulation but they can decrease in time.

3.6.1 Validity of the reduced models

The theorems from [28] prove convergence over time intervals of length 1 of the original models to the reduced models when the parameter ε vanishes. Therefore we cannot expect, in theory, that the approximation be valid over intervals of length $1/\varepsilon$ or larger. In addition, to the best of our knowledge, there are no estimates for the rate of convergence. In this section, we consequently assess numerically the quality of approximation of the reduced models in valid final times, *i.e.* in times of order $O(1)$. We then check how large the final time can be such that the reduced models still provide satisfactory approximations. To this end, we plot the relative error

$$\text{Error}(T_n) = \frac{\|\mathcal{G}_n - \mathcal{X}(T_n)\|_1}{\|\mathcal{X}(T_n)\|_1}, \tag{3.35}$$

where $\|\cdot\|_1$ stands for the ℓ_1 norm in $\mathbb{R}^6$, $\mathcal{G}_n$ stands for the reduced model solution at time T_n and $\mathcal{X}(T_n)$ stands for the original model solution at time T_n. Recall that $\mathcal{G}_n$ and $\mathcal{X}(T_n)$ have analytic forms for the test case in section 3.4.2, whereas numerical approximations are used for both for the test case in section 3.5. Next, we do not discuss the case described in section 3.4.1 since writing the solutions of the original and the reduced models in the forms (3.14) and (3.20) respectively, shows that convergence occurs after one iteration (see next section).

For the case considered in section 3.4.2 we recall that both the original and the first-order reduced models have analytic solutions given by (3.23) and (3.31). First, we remark that the exact solutions corresponding to the values of $\varepsilon \in \{0.1, 0.04\}$ are not well-approximated by the reduced model, see Fig. 3.3. We can see that beyond the final time $T = 50$ the approximations are not acceptable anymore. In contrast, for $\varepsilon = 0.01$ or smaller, the relative error is below 0.1 till the final time $T = 2500$, meaning that ε is small enough so that the reduced model provides a good approximation. Thus, if $\varepsilon = 0.01$, we obtain an acceptable relative error at the final time $T = 2500$, which means almost 40000 cyclotron orbits.

We now consider the test case of section 3.5. Here, we solve both models numerically, since no analytic expressions of their solutions are available. More precisely, we solve the system (3.32) by the symmetric and volume-preserving method G^4 of order 4 described in [56] and the limit model in (3.33) by the explicit Runge-Kutta 4 method. We use for the limit model approximation a time step equal to 0.625, whereas for the original dynamics we use a time step about $2\pi\varepsilon/80$ to accurately solve the cyclotron motion. We consider

two initial conditions

$$\mathbf{x} = (0, 1, 1)^T, \quad \mathbf{v} = (1, \varepsilon, 0)^T \tag{3.36}$$

and

$$\mathbf{x} = (1, 1, 1)^T, \quad \mathbf{v} = (1, \varepsilon, 0)^T. \tag{3.37}$$

Let us do some qualitative remarks about the trajectories of both particles. First, notice that the solutions obtained with these initial conditions behave differently: for the first particle, the solution oscillates at two time scales (a rapid oscillation of order ε and a slower oscillation of order 1) whereas for the second one, the solution entails additionally a slow motion, consisting of a linear drift in the $\overrightarrow{e_3}$ direction (see Fig. 3.1). Also, the amplitude of the rapid oscillation in position in the $\overrightarrow{e_3}$ direction is of order ε^2 for one particle and of order ε for the other.

Then, as we can deduce from (3.33), the two-scale limit model does not capture the motion in the $\overrightarrow{e_3}$ direction, providing only an approximation of the projected motion on the perpendicular plane to $\overrightarrow{e_3}$. Thus, the limit model misses the $\overrightarrow{e_3}$-drift motion of the particle in (3.37). Eventually, the right panel in Fig. 3.1 shows that the planar angular velocity of the particle in (3.36) is larger than that of the particle in (3.37). We plot the

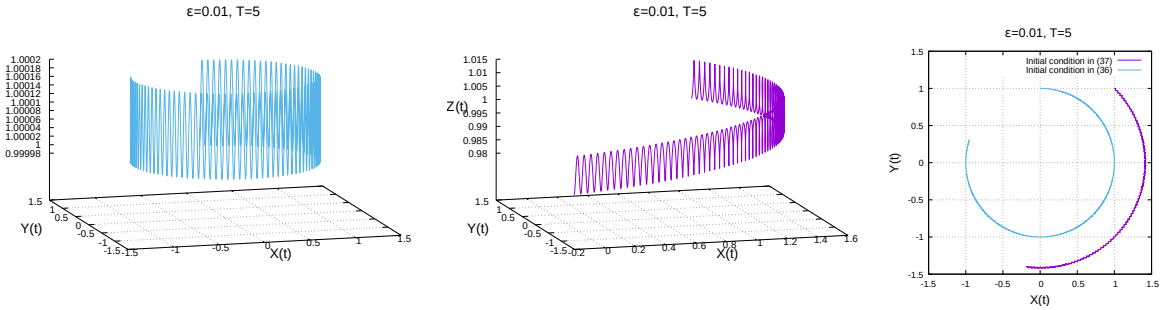

Figure 3.1: The position trajectories until final time 5 of two particles: (3.36) at the left panel and (3.37) at the center, following the model in (3.32). The projection of their motion on the perpendicular plane to $\vec{e}_3$ is at the right panel.

relative error of the reduced (limit) model for several values of ε in Fig. 3.4. We can see that the behaviour of the error displays significant difference between these two initial conditions. More precisely, we observe that at final time $T = 100$, the reduced model does not provide a good approximation of the original model when $\varepsilon \in \{0.1, 0.05\}$ in the case of the initial condition given by (3.37). On the contrary, when the initial condition is given by (3.36), the error is acceptable. However, for both particles, we deduce from Fig. 3.4 that the errors are large for times of order 2500, for any value of ε. In addition, when diminishing the time step for the numerical solver of the reduced model, we observe that the error drastically decreases when using the initial condition in (3.36). This result does not hold for the initial condition in (3.37), see Fig. 3.5.

In conclusion, we obtained for the first test case that the reduced model accurately approximates the original model in large times if ε is sufficiently small. For the second test case the reduced model fails to approximate the original dynamics in long times for every

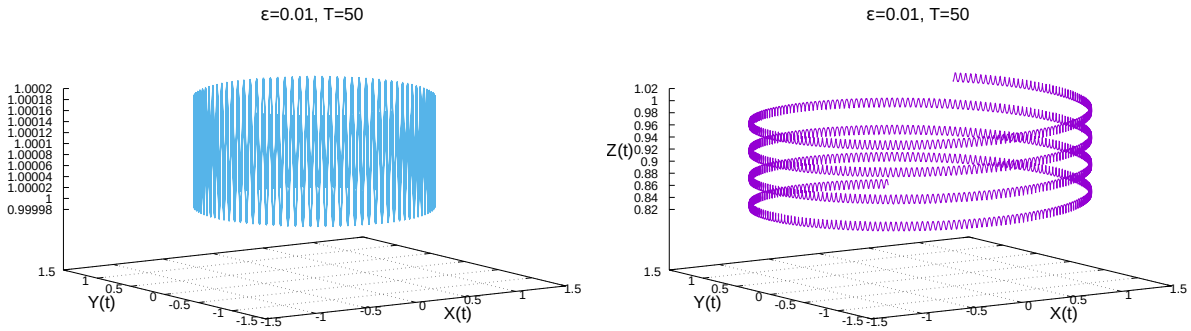

Figure 3.2: The position trajectories until final time 50 of two particles: (3.36) at the left panel and (3.37) at the right, following the model in (3.32).

considered value of ε. We show in the next sections that the parareal algorithm allows to enhance the situation.

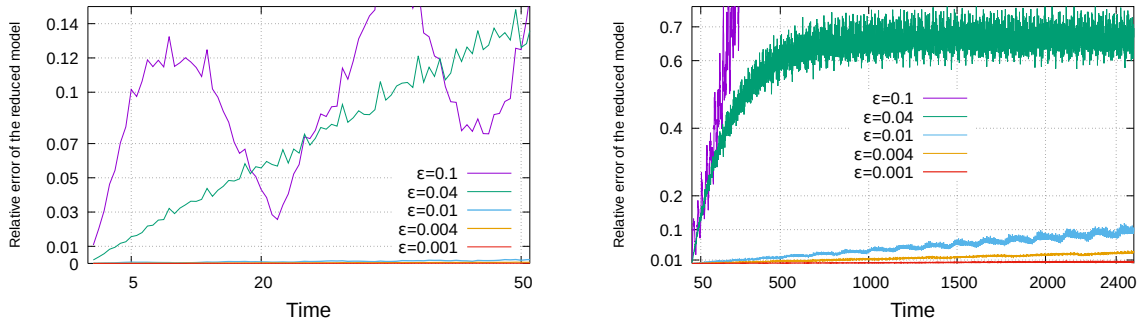

Figure 3.3: Evolution with respect to time of the relative errors of the reduced model solution in (3.31) with respect to the solution in (3.23) with the initial condition in (3.38), in short time (at left) and long time (at right), for several values of ε.

3.6.2 The test cases with strong constant magnetic field

First, we discuss about the case in section 3.4.1 of a uniform but time varying electric force in equation (3.11). Assume we fix an initial condition and we fix ε to a small value, say $\varepsilon = 0.01$. Then we use the exact solutions in (3.14) and (3.20) for the fine propagator $\mathcal{F}$ and respectively the coarse solver $\mathcal{G}$. We observe that the parareal algorithm writes in this case

$$U_{n+1}^{k+1} = \mathcal{F}(T_{n+1}, T_n, U_n^{k+1}), \quad \forall n \in \{0, \ldots, N-1\}, \forall k \geq 0.$$

In particular, for $k = 1$ we have

$$U_{n+1}^{1} = \mathcal{F}(T_{n+1}, T_n, U_n^{1}), \quad \forall n \in \{0, \ldots, N-1\}$$

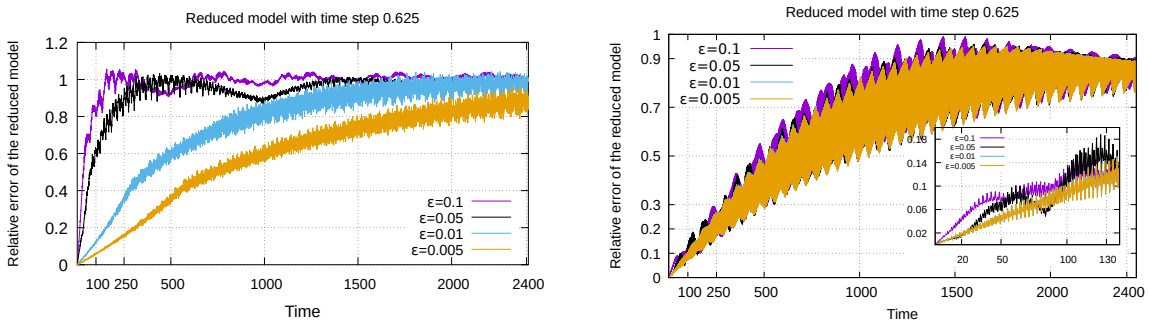

Figure 3.4: Evolution with respect to time of the relative errors of the numerical approximation of the reduced model in (3.34) with respect to the numerical solution of (3.32) with the initial condition in (3.37) (at left) and that in (3.36) (at right), for several values of ε.

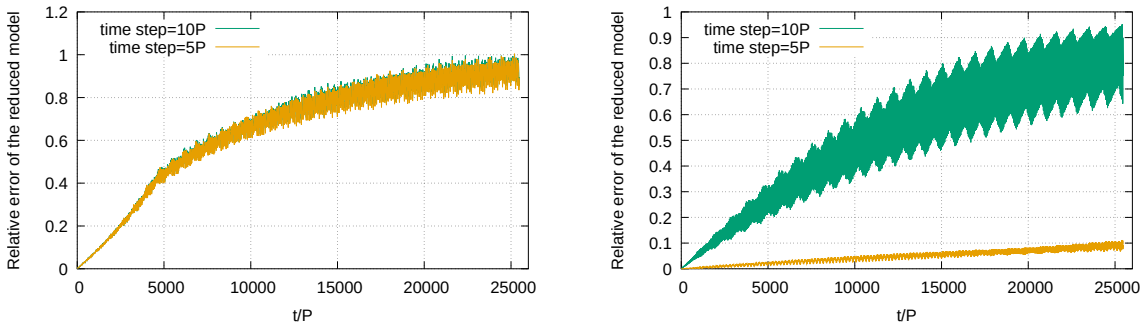

Figure 3.5: Evolution with respect to time of the relative errors of the reduced model in (3.34) with respect to the solution of (3.32) with the initial condition in (3.37) (at left) and that in (3.36) (at right). The fast cyclotron period is denoted by $P = 2\pi\varepsilon$ where $\varepsilon = 0.01$. Two time steps for the reduced model are used: $0.625 \sim 10P$ and $0.3125 \sim 5P$.

and therefore, since the exact flows are used for the propagators $\mathcal{F}$ and $\mathcal{G}$, the parareal algorithm provides an exact solution in one iteration. Though easy to solve, this test case underlines the strength of the strategy: thanks to the writing of the original and reduced flows as (3.14) and (3.20) respectively, the use of the reduced model through the parareal algorithm leads to high accuracy in one iteration, whereas the error of the reduced model alone is very large (of order 1, following our simulations when ε is fixed to $\varepsilon = 0.01$).

We now treat the case in section 3.4.2. We consider the initial condition

$$\mathbf{x} = (1, 1, 1)^T, \quad \mathbf{v} = (1, 1, 1)^T \tag{3.38}$$

for solving the model (3.21)-(3.22). We set $c = 2$ and we vary ε verifying (3.25). The solution issued from this initial condition oscillates at three definite time scales (see Remark 3.4.2).

• We first fix $\varepsilon = 0.01$. As a first approach, we apply the parareal method in a standard way, meaning that we use for the coarse propagator $\mathcal{G}$ the classical Runge-Kutta 4 method

for the initial model, with a bigger time step than that for the fine propagator. However, the coarse time step still needs to solve the smallest scale in order to have stability and reasonable accuracy. In this case, we have only to investigate the needed number of the parareal iterations for achieving convergence. More precisely, we first set the final time $T = 2\pi\varepsilon$ (one rapid oscillation), $N \in \{8, 16\}$ (larger N is not interesting), $\Delta t = T/N$, the number of coarse time steps on each time slice $MG = 1$ and the number of fine time steps on each time slice $MF = 80/N$. Thus, the fine time step $\delta t = T/80$ is fixed with respect to N and additionally is small enough for capturing the smallest scale. We plot at the top of Fig. 3.6 the relative error (in $L^\infty[0, T]$) between the solution $\mathcal{X}(t_n)$ obtained with the fine solver and the parareal solution U_n^k, as a function of the number k of parareal iterations

$$\text{Error}(k) = \frac{\max_{n\in\{1,\dots,N\}} \|U_n^k - \mathcal{X}(t_n)\|_1}{\max_{n\in\{1,\dots,N\}} \|\mathcal{X}(t_n)\|_1}, \tag{3.39}$$

where $\|\cdot\|_1$ stands for the ℓ_1 norm in $\mathbb{R}^6$. We obtain convergence of the algorithm for small k (4 or 5), however in a case of a too small T from the application point of view. When taking a larger final time $T = 8\pi\varepsilon$ (4 oscillations) with $MF = 320/N$ and $\delta t = T/320$, we have convergence of parareal for k very close to N (see the bottom of Fig. 3.6 for $N \in \{8, 16\}$). We can conclude that this parareal strategy provides convergence after $k \lesssim N$ iterations and with a ratio $\Delta t/\delta t \sim 1$, which is not an interesting approach.

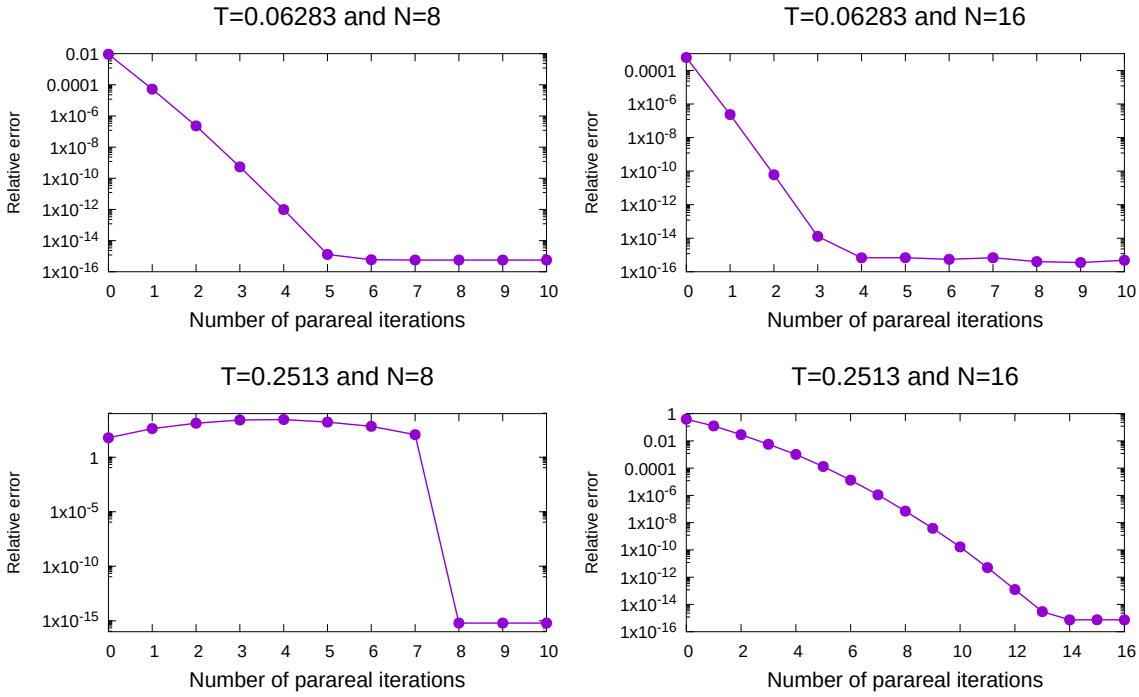

Figure 3.6: Convergence rate of standard parareal algorithm for the test case in section 3.4.2.

• We now propose to use for the coarse solver $\mathcal{G}$ the reduced model in section 3.4.2 and we thus make use of the analytic expression in (3.31). In addition, we use for the fine

	$N = 2$	$N = 4$	$N = 8$	$N = 16$	$N = 32$	$N = 64$	$N = 128$
$\Delta t/P$	39.79	19.89	9.95	4.97	2.49	1.24	0.62

Table 3.1: Numbers of cyclotron periods ($P = 2\pi/a_\varepsilon$) enclosed in a time step of the coarse solver for several values of N. We have $T = 5$, $\Delta t = T/N$, and $\varepsilon = 0.01$.

solver $\mathcal{F}$ the explicit form of the solution in (3.23). We start by illustrating the convergence of the algorithm in short time simulations. We fix the final time $T = 5$ and the interval $[0, T]$ is partitioned in $N \in \{2, 4, 8, 16, 32, 64, 128\}$ sub-intervals. The big time step is thus $\Delta t = T/N$. It is interesting to note the size of the coarse time step Δt with respect to the small cyclotron period P, when N varies (see Table 3.1). Larger is $\Delta t/P$, larger is the ratio $\Delta t/\delta t$ and thus, cheaper is the coarse propagator.

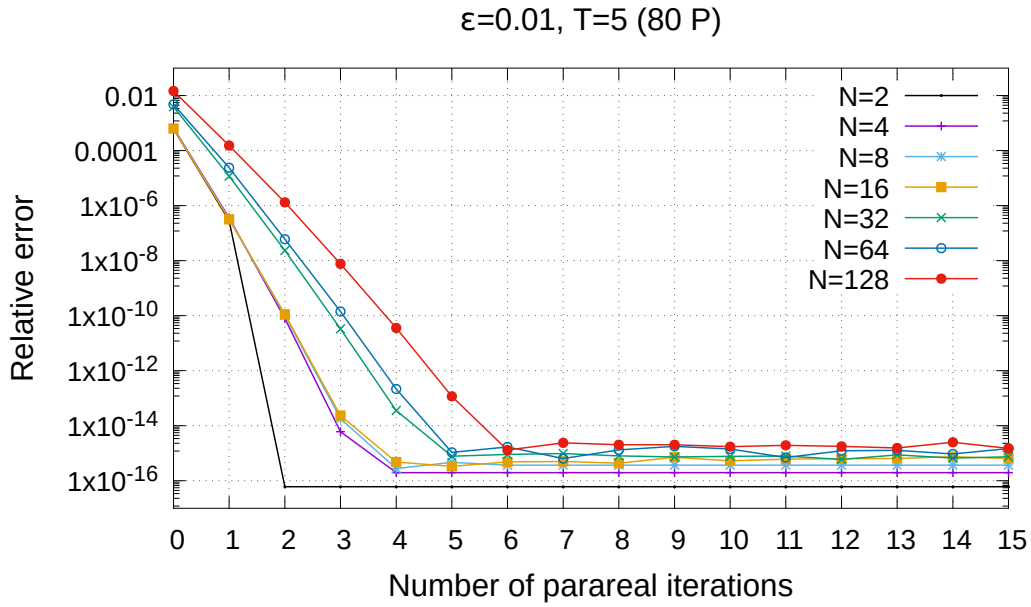

Figure 3.7: Convergence of the parareal algorithm for the Penning trap test case at short final time $T = 5$. The fast cyclotron period is denoted by $P \sim 2\pi\varepsilon$.

We plot in Fig. 3.7 the relative error in $L^\infty[0, T]$ defined in (3.39) by taking the solution in (3.23) for $\mathcal{X}(t_n)$. The case $k = 0$ corresponds to the relative error of the solution of the reduced model with respect to the exact solution of the original model. We obtained convergence of the algorithm after a maximum of 6 iterations for all the considered values of N.

• We now analyze the behaviour of the parareal algorithm when ε decreases, in which case the reduced model becomes a more accurate approximation for the initial equation. We display in Fig. 3.10 the relative errors illustrating the convergence of the parareal algorithm. We plot for each value of N in the set $\{8, 16, 32, 64\}$ the errors for several values of ε at final

time $T = 500\varepsilon$ which corresponds to approximately 80 cyclotron periods. As expected, the initial errors of the parareal method (*i.e.* $k = 0$) are decreasing when ε becomes smaller. Also, the smaller is ε, the faster is the convergence of the parareal algorithm since the better is the approximation of the reduced model. We already observed in section 3.6.1 that for $\varepsilon \in \{0.1, 0.04\}$, the reduced model induces a much bigger error than for the other smaller values of ε. However, with the parareal strategy we obtain acceptable convergence results for $\varepsilon = 0.1$, which entails a $O(1)$ error for the reduced model: the parareal method convergences after $k = 10$ (resp. $k = 14$) iterations when $N = 32$ (resp. $N = 64$). Except for the values of $\varepsilon \in \{0.1, 0.04\}$, the convergence of the parareal algorithm for all the considered values of N is obtained after a maximum of $k = 5$ iterations. We also emphasize the achievement of an uniform error with respect to ε.

• Then, we consider the more challenging case of a long time simulation (of order $1/\varepsilon$). We fix the final time $T = 600 \sim 2\pi/b_\varepsilon$, where b_ε is defined in (3.24) and we take N in the set $\{120, 240, 480, 960\}$. As previously, we plot in Fig. 3.8 the relative errors between the exact solution and the parareal solution, as a function of the number k of parareal iterations. For this case, we can conclude with underlying the strength of using the parareal algorithm. The reduced model is not proved to be an approximation of the initial model in time of order $1/\varepsilon$. However, in a few number of parareal iterations we obtain high accuracy by applying the reduced model on valid intervals. Thus, if parallelism is to be used, the computational cost in the case of $N = 480$ (resp. $N = 960$) could drastically be reduced, achieving a round-off error in only 7 parareal iterations. In this case, a time slice includes approximately 20 (resp. 10) rapid oscillations.

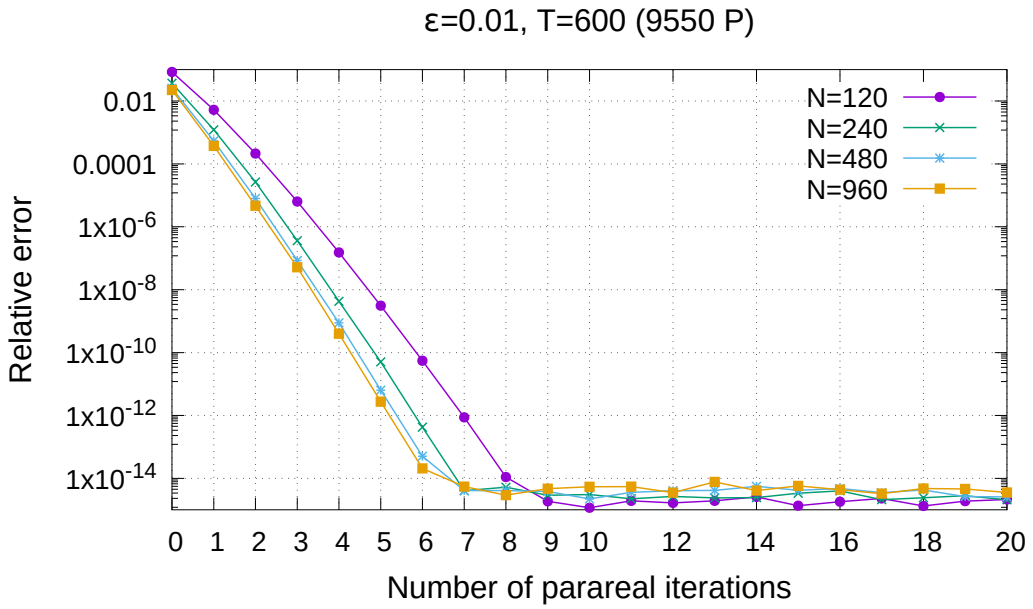

Figure 3.8: Convergence of the parareal algorithm for the Penning trap test case at final time $T = 600$. The fast cyclotron period is denoted by $P \sim 2\pi\varepsilon$.

• Finally, we show the outcome of much longer simulations, where we keep the coarse time step fixed while the final time is increased with N. This framework is relevant for applications where one needs to integrate over very long times. We fix $\varepsilon = 0.01$ and the coarse time step to $\Delta t = 1.25$. The final time T is chosen in the set $\{2000, 4000, 8000, 16000, 32000\}$, see Fig. 3.9. Setting $N = 25600$, we observe that when $T = 32000$, *i.e.* T larger than 500000 cyclotron orbits, the convergence of the parareal algorithm is obtained in $k = 21$ iterations, with an error around 10^{-13}. In our opinion this is an excellent result which is due to the accuracy of the reduced model. Being of first order, the model provides good approximations of the slow motion and of the fast oscillation.

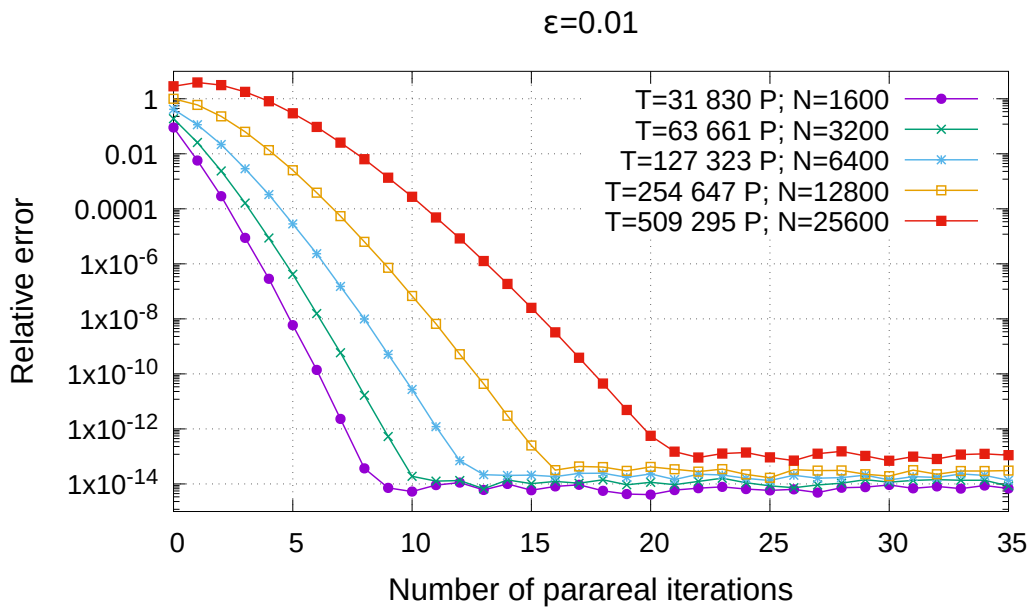

Figure 3.9: Convergence of the parareal algorithm for the Penning trap test case when the coarse time step is kept constant to 1.25 and the final time T is increasing with N. The fast cyclotron period is denoted by $P \sim 2\pi\varepsilon$.

3.6.3 The test case with strong variable magnetic field

We now consider solving the problem in (3.32) with the initial conditions in (3.36) and (3.37). As in the previous section, we discuss the results of our simulations when $\varepsilon = 0.01$ in final times of order 1 and $1/\varepsilon$, and then we perform simulations in short final times by varying the values of ε. We recall that we use a symmetric and volume-preserving scheme and the classical Runge-Kutta 4 method for the models in (3.32) and (3.33) respectively, for the fine and respectively the coarse propagators. However, while the $\mathcal{F}$ propagator needs a time step δt which is a fraction of the rapid oscillation ($P \sim 2\pi\varepsilon$), the $\mathcal{G}$ propagator is computed with a time step Δt much larger than $2\pi\varepsilon$ (see typical values in Table 3.1).

• We first set $\varepsilon = 0.01$. We fix the final time $T = 5$ and we partitioned the interval $[0, T]$ in $N \in \{2, 4, 8, 16, 32, 64, 128\}$ sub-intervals. The coarse time step is $\Delta t = T/N$ and

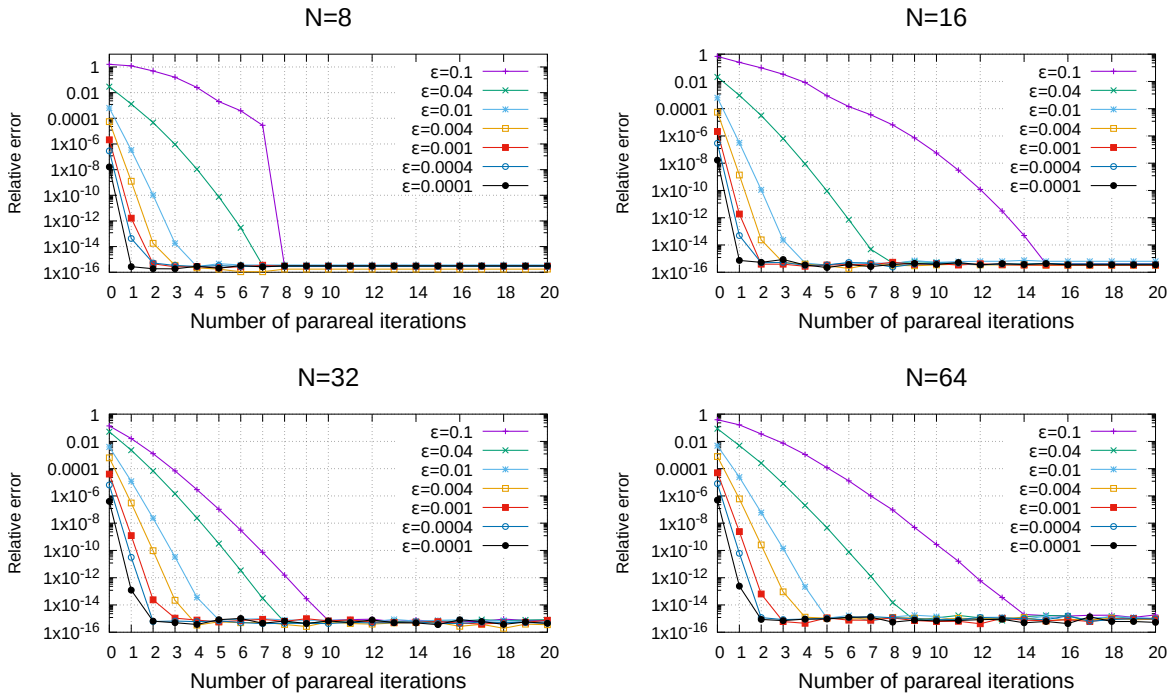

Figure 3.10: Convergence of the parareal algorithm in short final times, $T = 500\varepsilon \sim 80P$, for several small values of ε, for the test case in section 3.4.2.

the fine time step is fixed to $\delta t = T/6400$, which is sufficiently small to solve the rapid oscillation. We plot at the top of Fig. 3.11 the relative error defined in (3.39) of the parareal solution with respect to the reference one computed with the $\mathcal{F}$ propagator. We observe convergence after a maximum of $k = 9$ iterations when $N \in \{32, 64, 128\}$ which could lead to satisfactory speed-up if parallel processing is set up.

• Then, we plot in Fig. 3.12 the errors of the parareal algorithm when ε vanishes. For a fixed number of time slices of $[0, T = 500\varepsilon]$ we perform simulations when ε goes in $\{0.01, 0.04, 0.01, 0.004, 0.001, 0.0004, 0.0001\}$. We find numerically the property of smaller errors with smaller ε, for every k, due to the smaller error of the reduced model with respect to the initial equation. We obtain unsatisfactory results when $\varepsilon = 0.1$, since we recall from the left panel of Fig. 3.4 that the reduced model is not a good approximation at $T = 50$ for this case. On the contrary, the value of $\varepsilon = 0.04$ leads to satisfactory parareal results, when $N \in \{32, 64\}$. For the other smaller values of the parameter, we observe convergence of the algorithm for all N after a maximum of $k = 9$ iterations. As for the test case in the previous section, we obtain uniform error with respect to ε.

• However, the most interesting problem is that of a long time simulation. We now fix $T = 50$ and we take N in the set $\{20, 40, 80, 160\}$. A bigger value of T can be treated similarly, since the trajectories of both particles evolve as until $T = 50$, with a linear drift in the $\overrightarrow{e_3}$ direction for the particle in (3.37) (see Fig. 3.2). The fine time step is set to $\delta t = T/64000$. We plot the relative error at the bottom of Fig. 3.11. We obtained when N is small much larger errors of the parareal algorithm for the particle in (3.36) because of

its larger perpendicular angular velocity, as mentioned above. Indeed, when N is small, *i.e.* when the time step is big, the error of the limit model is too large so that the parareal method (or the fine solver) can catch a convenient accuracy in a small number of iterations. At the top of Fig. 3.11, $N = 2$ means a coarse time step of almost 40 rapid oscillations; we have the same remark for $N = 20$ at the bottom of the figure. Nevertheless, we note that in the interesting case of $N = 80$ (resp. $N = 160$), we achieve convergence after only $k = 9$ iterations. This value of N corresponds to a coarse time step of almost 10 (resp. 5) rapid oscillations and to a ratio $\Delta t/\delta t$ of 800 (resp. 400). Thus, coupled to the parallel computations of the fine solver, this strategy could be very effective in terms of computational costs.

• Finally, we perform longer simulations, by keeping the coarse time step fixed while the final time increases with N. The obtained results (see Fig. 3.13) are not as good as those reported in the previous section, as a consequence of the accuracy of the approximation of the reduced model (compare Fig. 3.4 to Fig. 3.3). We fix $\varepsilon = 0.01$, the coarse time step to $\Delta t = 1.25$ and we use the initial condition in (3.37); smaller values of Δt do not significantly improve the error of the reduced model. As for the particle in (3.36), we fix $\Delta t = 0.625$, in order to approximate the slow circular motion with a similar accuracy as for the other particle. When considering the initial condition in (3.37), the results of the parareal algorithm are not fully satisfactory when the final time is large (see Fig. 3.13): for example, when $T = 1000$, *i.e.* almost 16000 cyclotron periods, we obtain an error of order 10^{-5} after less than $k = 60$ iterations (recall $N = 800$) but afterwards, the error decays very slowly, a quite large number of parareal iterations being necessary to achieve a much smaller error. The situation is completely different when using the initial condition in (3.36). We obtain good convergence results of the parareal algorithm in large times: at $T = 1000$, for $N = 1600$ time slices, we achieve a 10^{-10} error after $k = 25$ iterations.

To further understand the rationale behind the slow convergence of the algorithm for this test case, we assess the long-term energy error, which is a major issue in applications. First, we verified that the volume-preserving numerical scheme G^4, used as fine solver, preserves the Hamiltonian of the system (3.32), at the accuracy of the machine precision. More precisely, the Hamiltonian is $\mathcal{H}(\mathbf{x}, \mathbf{v}) = |\mathbf{v}|^2/2$, since there is no electric term. Following [34], we plot in Fig. 3.14 the error in the energy

$$\mathcal{H}(\mathbf{x}_n^k, \mathbf{v}_n^k) - \mathcal{H}(\mathbf{x}_0, \mathbf{v}_0) \tag{3.40}$$

where $(\mathbf{x}_0, \mathbf{v}_0)$ is the initial condition and $(\mathbf{x}_n^k, \mathbf{v}_n^k)$ is the k-th iterate of the parareal algorithm. We display, for both initial conditions given by (3.36) and (3.37), the energy error corresponding to the first 6 parareal iterations and then, the $k = 10$-th iterate respectively the $k = 50$-th iterate, taking into account when the parareal convergence is achieved (see Fig. 3.13). We can see that the energy error of the particle in (3.36) has no large amplitude oscillations in time and the convergence is quite fast, unlike the particle in (3.37). This is in accordance with the results in Fig. 3.13.

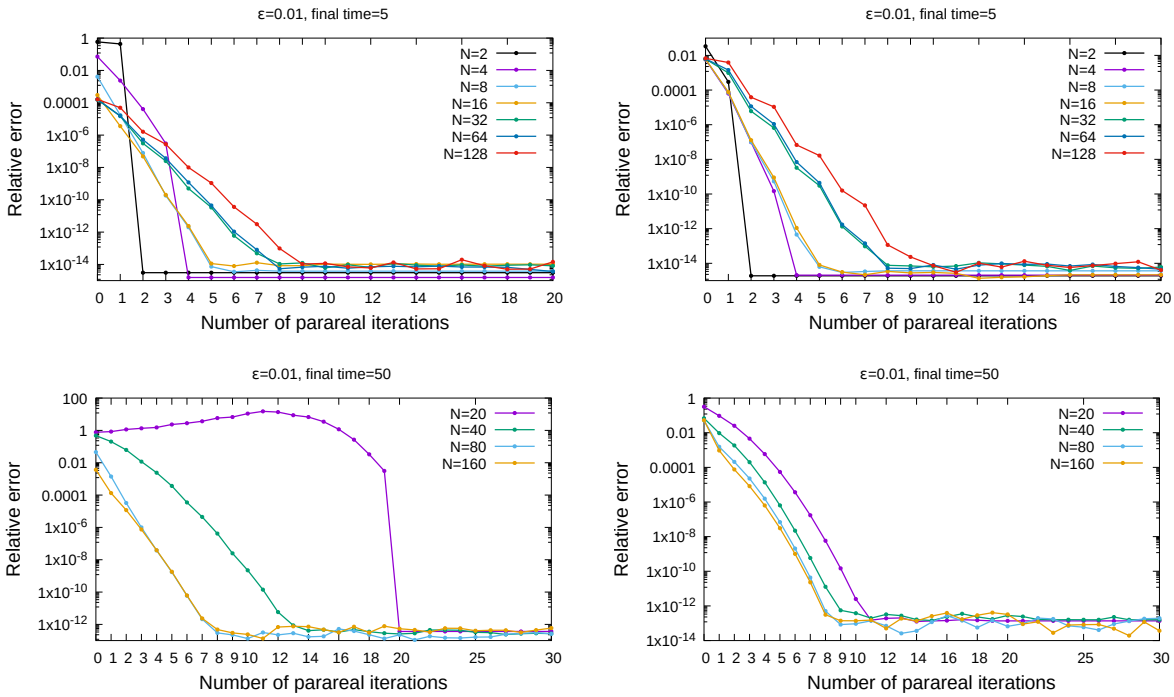

Figure 3.11: Convergence of the parareal algorithm for the test case in section 3.5 in short final time $T = 5 \sim 80P$ (at the top) for the initial conditions in (3.36) (left panel) and (3.37) (right panel) and in longer time $T = 50$ (at the bottom).

3.7 Conclusions and perspectives

In this paper, we proposed a coupling strategy using the limit model based on the two-scale asymptotic expansion for the coarse propagator in the parareal algorithm, for efficiently solving oscillatory singularly perturbed ODEs which are characteristics of a six-dimensional Vlasov equation. Rapid convergence of the parareal algorithm is obtained in simulations of charged particles, tests of a Penning trap and of isotope separation by ion cyclotron resonance and test of an example of strong variable magnetic field. The method was shown to be accurate and efficient with a uniform convergence rate. The convergence analysis as well as the Vlasov-Poisson equation using the same strategy will be studied in the future work.

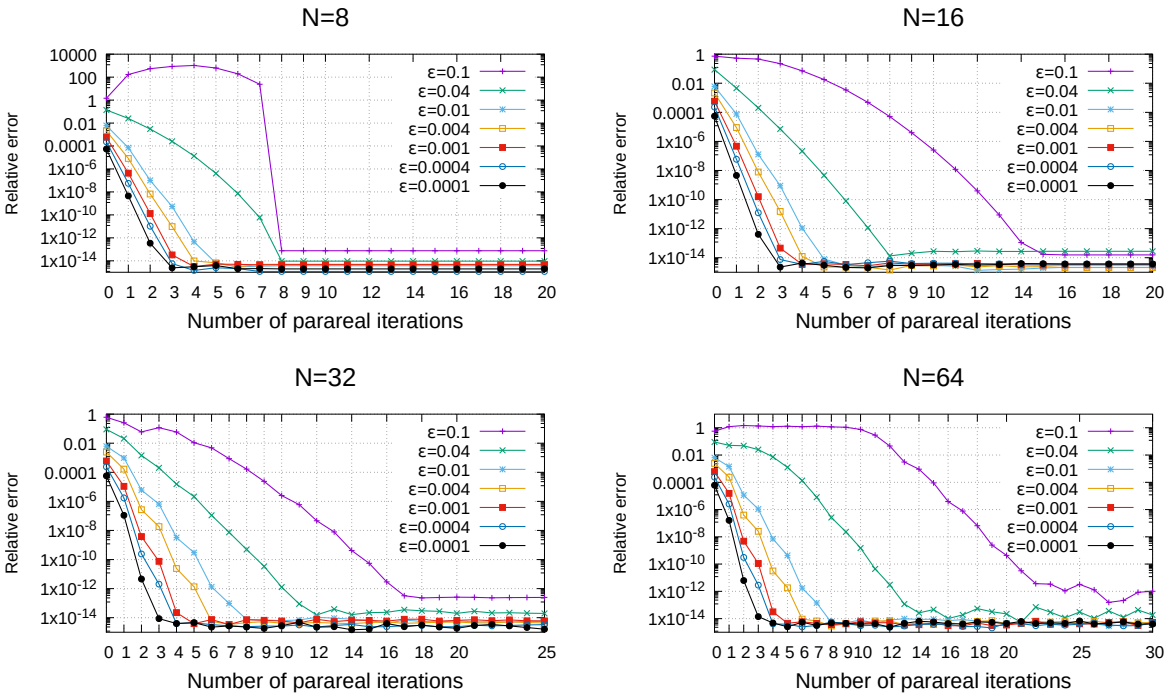

Figure 3.12: Convergence of the parareal algorithm in short final times, $T = 500\varepsilon \sim 80P$, for several small values of ε, for the initial condition in (3.37), for the test case in section 3.5.

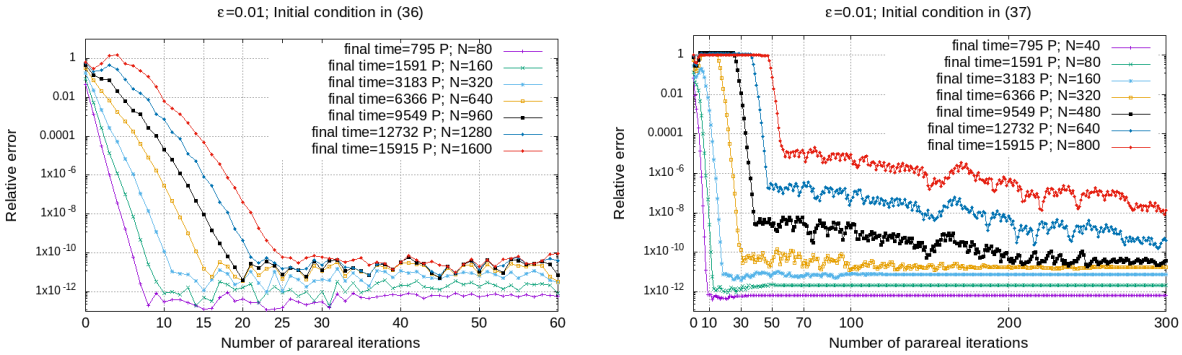

Figure 3.13: Convergence of the parareal algorithm for the test case in section 3.5 when the coarse time step Δt is kept constant and the final time is increasing with N. At the left panel: the initial condition in (3.36) and $\Delta t = 0.625$. At the right panel: the initial condition in (3.37) and $\Delta t = 1.25$. The fast cyclotron period is denoted by $P = 2\pi\varepsilon$.

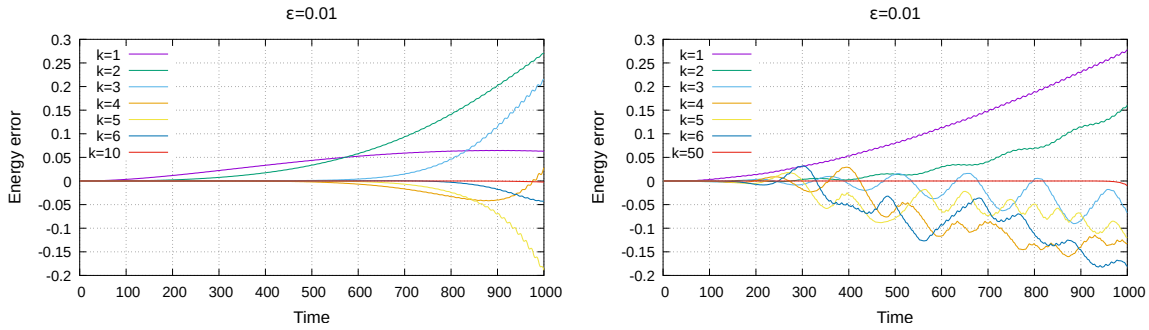

Figure 3.14: Evolution of the energy error defined in (3.40) of the parareal algorithm for the test case in section 3.5, until the final time $T = 1000 = 15915P$ with $P = 2\pi\varepsilon$. Left panel: the initial condition in (3.36) with $\Delta t = 0.625$ for the coarse solver. Right panel: the initial condition in (3.37) with $\Delta t = 1.25$ for the coarse solver.

4 Deflation Technique of the Smallest Singular Values based on Nested Dissection Partition and Sparse QR Factorization with Tournament Pivoting

Contents

Abstract

We construct a deflation preconditioner based on sparse QR factorization with tournament pivoting strategy and nested dissection partitioning. This preconditioner aims at replacing the smallest singular values which are close to zero by larger ones in order to improve the convergence rate of Krylov methods, namely GMRES. We combine the deflation preconditioner with block Jacobi preconditioner to obtain better convergence rate of GMRES. We also present the deflation of singular vectors based on QR factorization and discuss the comparison between strong RRQR deflated GMRES and SVD deflated GMRES.

Keywords: QR factorization, strong RRQR factorization, tournament pivoting, nested dissection, deflation preconditioner, GMRES, SVD, block Jacobi preconditioner.

4.1 Introduction

In the context of parareal algorithm, two solvers are used to propagate the solution from one time point to the next, these are the coarse and the fine solvers. The coarse solver relies on a low order approximation that gives a coarse approximate for the solution sequentially. In the opposite, the fine solver relies on a high order approximation and gives a very accurate approximate for the solution. Despite of being computed in parallel, the fine solver is often very expensive terms of computational cost, especially when the size of the problem becomes large. It requires solving large sparse linear systems arising from the discretization of the problem, which are often ill-conditioned, slowly convergent and require a lot of computational efforts. For this, a promising approach is to exploit the information generated from previous iterations, particularly in [38] the authors used information generated from the fine solver of previous iterations to obtain a more accurate approximation for the coarse solver by constructing an enhanced Krylov subspace of known fine evolution. We know from [24, 78, 89, 99] that small eigenvalues of the coefficient matrix can badly affect the convergence of the solver. The linear system changes from one time step to the next, hence information generated from previous iterations can be exploited, so recycling Krylov subspace solvers, see [82], namely the GCRO-DR [18] or GMRES-DR [78] method can be used for the fine solver. Those methods are the Krylov subspace solvers as Conjugate gradient and GMRES, but instead of executing alone, they are combined with some deflation strategy to remove the smallest eigenvalues, and are designed to be restarted to keep only some useful information for memory requirement purposes, for more details see [24]. Following the work in [1], motivated by the work of Simoncini studying the relation between the singular vectors associated to the smallest singular values and the convergence of some restarted Krylov methods, see [96], in this work, we also aim at studying the deflation of the singular vectors associated to the smallest singular values. But instead of using the SVD factorization, we use the strong

rank revealing QR (RRQR) factorization combined with tournament pivoting strategy and nested dissection partitioning on $\boldsymbol{A}^T\boldsymbol{A}$. Strong RRQR factorization was shown to efficiently approximate the largest singular values, see [10, 19]. We show in our study that it also provides good approximations for the smallest singular values with a reasonable computational cost compared to the SVD factorization. By combining with the tournament pivoting strategy, our goal is to obtain an efficient and highly parallel computation of the space to be deflated in GMRES method. This strategy results in constructing a deflation preconditioner which replaces the smallest singular values close to zero by much larger ones so that GMRES will not suffer from bad convergence due to the impact of those smallest singular values in the spectrum. Then by using additionally the block Jacobi preconditioner, GMRES results in a much faster convergence rate. This chapter is organized as follows, we recall the strong RRQR factorization in section 4.2 and the tournament pivoting strategy in section 4.3. The combination between strong RRQR and tournament pivoting strategy is presented in section 4.4. Section 4.5 is dedicated to the construction of the deflation preconditioner. In section 4.6 we briefly recall the block Jacobi preconditioner and the combination between the deflation and block Jacobi preconditioners. We give in section 4.7 the deflation of singular vectors based on strong RRQR factorization and the numerical results are given in section 4.8.

4.2 Strong Rank Reavealing QR (Strong RRQR) factorization

Since introduced in [42] and the first algorithm so-called QR with column pivoting (QRCP) was developed in [13], RRQR factorization shows its efficiency in many applications such as low rank approximations, least square computations, nonsymmetric eigenproblems and regularization (see [14, 15, 42, 45, 66]). However, RRQR is not a stable algorithm since the elements of the term $R_{11}^{-1}R_{12}$ in the factorization can be very large as stated in [48]. To overcome this problem, the authors introduced in [48] a new factorization so-called strong RRQR. One of the most important application of strong rank revealing QR factorization is to build a basis for the approximate right null space of $\boldsymbol{A}$, especially in rank-deficient least-squares problems, subspace selection and linear dependency analysis or subspace tracking,(see [11, 43, 44, 45, 62, 104]). In this work we mainly focus on the rank revealing of the QR factorization to approximate the smallest singular values of $\boldsymbol{A}$. Our main results come from the strong rank revealing QR (RRQR) factorization definition following [48, Theorem 3.2] and [19, Theorem 2.4] as,

Theorem 4.2.1. *([48, Theorem 3.2] and [19, Theorem 2.4]) Let $\boldsymbol{A}$ be an $m \times n$ matrix and $1 \leq k \leq \min(m, n)$. Let $f > 1$ and Π be a permutation matrix such that the decomposition*

$$\boldsymbol{A}\Pi = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}, \tag{4.1}$$

verifies for all $(i, j) \in [1, k] \times [1, n-k]$,

$$\gamma_j^2(R_{11}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq kf^2. \tag{4.2}$$

Then for any $1 \leq j \leq n-k$ *and* $1 \leq i \leq k$,

$$1 \leq \frac{\sigma_i(\boldsymbol{A})}{\sigma_i(R_{11})} \leq \sqrt{1+kf^2(n-k)},\ 1 \leq \frac{\sigma_j(R_{22})}{\sigma_{k+j}(\boldsymbol{A})} \leq \sqrt{1+kf^2(n-k)}, \tag{4.3}$$

where $\Pi \in \mathbb{R}^{n\times n}, R_{11} \in \mathbb{R}^{k\times k}, R_{12} \in \mathbb{R}^{k\times(n-k)}, R_{22} \in \mathbb{R}^{(m-k)\times(n-k)}, Q_1 \in \mathbb{R}^{m\times k}$ and $Q_2 \in \mathbb{R}^{m\times(n-k)}, \gamma_j(R_{22})$ is the 2-norm of the jth column of R_{22}, $\sigma_i(\boldsymbol{A}), 1 \leq i \leq \min(m,n)$ denotes the ith singular value of $\boldsymbol{A}$ and $\sigma_{\min}(R_{11})$ is the smallest singular value of R_{11}. We note in (4.3) that the lower bounds always hold for any permutation Π thank to the interlacing property of singular values.

Since in this work we put our interest in approximations for the smallest singular values of R_{22} and we restrict the problem in a square matrix $\boldsymbol{A}$, which results in remark 4.2.2 a slightly different version of Theorem 4.2.1 but the results and the proof remain the same as in Theorem [48, Theorem 3.2] and [19, Theorem 2.4],

Remark 4.2.2. *Let* $\boldsymbol{A}$ *be an* $n\times n$ *matrix and* $1 \leq k \leq n$. *Let* $f > 1$ *and* Π *be a permutation matrix such that the decomposition*

$$A\Pi = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}, \tag{4.4}$$

verifies for all $(i,j) \in [1, n-k] \times [1,k]$,

$$\gamma_j^2(R_{11}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq (n-k)f^2. \tag{4.5}$$

Then for any $1 \leq j \leq k$ *and* $1 \leq i \leq n-k$,

$$1 \leq \frac{\sigma_i(\boldsymbol{A})}{\sigma_i(R_{11})} \leq \sqrt{1+kf^2(n-k)},\ 1 \leq \frac{\sigma_j(R_{22})}{\sigma_{n-k+j}(\boldsymbol{A})} \leq \sqrt{1+kf^2(n-k)}, \tag{4.6}$$

where $\Pi \in \mathbb{R}^{n\times n}, R_{11} \in \mathbb{R}^{(n-k)\times(n-k)}, R_{12} \in \mathbb{R}^{(n-k)\times k}, R_{22} \in \mathbb{R}^{k\times k}, Q_1 \in \mathbb{R}^{n\times(n-k)}$ and $Q_2 \in \mathbb{R}^{n\times k}, \gamma_j(R_{22})$ is the 2-norm of the jth column of R_{22}, $\sigma_i(\boldsymbol{A}), 1 \leq i \leq n$ denotes the ith singular value of $\boldsymbol{A}$ and $\sigma_{\min}(R_{11})$ is the smallest singular value of R_{11}.

In the following discussion we describe how to combine sparse QR, especially strong RRQR with tournament pivoting strategy to build a deflation preconditioner of the smallest singular values of $\boldsymbol{A}$.

4.3 Tournament pivoting strategy

In the need of minimizing the communication cost as well as effectively exploit the parallel computing in such algorithms as LU, QR, RRQR,... tournament pivoting strategy was introduced and became a very promising candidate. Introduced in [19, 46] as a communication avoiding technique for LU and strong RRQR factorizations, tournament pivoting aims to select k columns from the input matrix $\boldsymbol{A}$. Let us briefly recall the tournament pivoting strategy following [10]. Consider a matrix $\boldsymbol{A}$ which can be partitioned into 4 blocks of columns, $\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \end{bmatrix}$. We first perform strong RRQR to select k columns from each column block $\boldsymbol{A}_{1i},\ 1 \leq i \leq 4$, we denote by $I_{i0},\ 1 \leq i \leq 4$

the set of indices of selected columns corresponding to each column block $\boldsymbol{A}_{1i}$. Next we concatenate the sets of selected columns from the previous step together in pair following a reduction binary tree then perform strong RRQR to select k columns from each pair. In particular in the second step, we perform strong RRQR to select k columns from $\boldsymbol{A}(:, I_{10} \cup I_{20})$ and k columns from $\boldsymbol{A}(:, I_{30} \cup I_{40})$, where $\boldsymbol{A}(:, I)$ denotes the submatrix whose columns come from the indices I of $\boldsymbol{A}$. We denote by $I_{i1},\ 1 \leq i \leq 2$ the sets of selected columns from the second step. The $2k$ selected columns from the second step are then concatenated as $\boldsymbol{A}(:, I_{11} \cup I_{21})$ and we perform strong RRQR again to select the final set of k columns which is denoted by I_{12}. We summarize all steps of the tournament pivoting technique to select k columns from the columns of matrix $\boldsymbol{A}$ in the following Algorithm 7, which was recalled in Chapter 1,

Algorithm 7: Tournament pivoting for 1D column partitioned matrices, one reduction step to select the k columns

Input: $\boldsymbol{A}_1, \cdots, \boldsymbol{A}_p$ submatrices of rank k approximation

1 Perform strong RRQR to select k columns from each $\boldsymbol{A}_i$, indices of selected columns are denoted by I_i, $1 \leq i \leq p$

2 Selected columns are concatenated in $\tilde{\boldsymbol{A}} = [\boldsymbol{A}_1(:, I_1) \cdots \boldsymbol{A}_p(:, I_p)]$

3 Perform again strong RRQR of $\tilde{\boldsymbol{A}}$ to select k columns

Output: indices of k rank revealing columns of $\boldsymbol{A}$

Although tournament pivoting was introduced to approximate the largest singular values and associated singular vectors for example see [10], in this work the algorithm is modified to approximate the smallest singular values and associated singular vectors, which will be described in more detail in the next section 4.4.

4.4 Approximation of the smallest singular values of A by sparse QR with tournament pivoting and nested dissection

In this section we present our strategy by performing sparse QR with tournament pivoting and nested dissection partitioning to select k columns that allow to approximate the k smallest singular values and corresponding singular vectors of A. Nested dissection [40, 41, 69] is a fill-reducing ordering technique that relies on a divide-and-conquer strategy of the graph of a symmetric matrix. The original graph is first partitioned into two subgraphs that are connected by a separator. The bisection is then recursively executed on each subgraph. Specifically in this work, the matrix $\boldsymbol{A}$ is partitioned by applying nested dissection on the graph of $\boldsymbol{A}^T\boldsymbol{A}$ using METIS library, see [63]. The standard nested dissection requires the input matrix to be symmetric, this explains why we use the nested dissection partition on $\boldsymbol{A}^T\boldsymbol{A}$ for $\boldsymbol{A}$. In the following, we consider a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ that is partitioned by using nested dissection on $\boldsymbol{A}^T\boldsymbol{A}$ as,

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11} & & \boldsymbol{A}_{13} \\ & \boldsymbol{A}_{22} & \boldsymbol{A}_{23} \\ \boldsymbol{A}_{31} & \boldsymbol{A}_{32} & \boldsymbol{A}_{33} \end{bmatrix},$$

where $\boldsymbol{A}_{ii} \in \mathbb{R}^{n_i\times n_i}$, $i = 1:3$, and $n_1 + n_2 + n_3 = n$.

The nested dissection partitioning is necessary for the convenient computation of the inverse of some blocks, this will be explained in more detail later. We note that each $\boldsymbol{A}_{ii}^{j}, i, j = 1:2$ is a subdomains of the nested dissection partition, for that reason with a nested partition of 4 subdomains, we can write $\boldsymbol{A}$ as,

$$\boldsymbol{A} = \begin{bmatrix} A_{11}^1 & & A_{13}^1 & & & & A_{10} \\ & A_{22}^1 & A_{23}^1 & & & & A_{20} \\ A_{31}^1 & A_{32}^1 & A_{33}^1 & & & & A_{30} \\ & & & A_{11}^2 & & A_{13}^2 & A_{40} \\ & & & & A_{22}^2 & A_{23}^2 & A_{50} \\ & & & A_{31}^2 & A_{32}^2 & A_{33}^2 & A_{60} \\ A_{10}^T & A_{20}^T & A_{30}^T & A_{40}^T & A_{50}^T & A_{60}^T & A_{70} \end{bmatrix},$$

where $A_{i3}^1, i = 1:3$ denote the local interface corresponding to the first and the second subdomains, $A_{i3}^2, i = 1:3$ denote the local interface corresponding to the third and the fourth subdomains and $A_{i0}, i = 1:7$ denote the global interface of $\boldsymbol{A}$. In addition, we denote by $I_p,\ p = 1:4$ the indices of columns corresponding to the pth subdomain of $\boldsymbol{A}$, by I_{ij} the indices of the local interface corresponding to the ith, jth subdomains and by I_0 the indices of columns corresponding to the global interface.

We first perform strong RRQR on a subdomain plus the columns corresponding to the interfaces to select the last k columns, e.g. for the first subdomain A_{11}^1 we consider the matrix,

$$\boldsymbol{A}(:, I_1 \cup I_{12} \cup I_0)\Pi_1^1 = \begin{bmatrix} A_{11}^1 & A_{13}^1 & A_{10} \\ & A_{23}^1 & A_{20} \\ A_{31}^1 & A_{33}^1 & A_{30} \\ & & A_{40} \\ & & A_{50} \\ & & A_{60} \\ A_{10}^T & A_{30}^T & A_{70} \end{bmatrix} \Pi_1^1.$$

For the second subdomain A_{22}^1, we consider the matrix

$$\boldsymbol{A}(:, I_2 \cup I_{12} \cup I_0)\Pi_1^2 = \begin{bmatrix} & A_{13}^1 & A_{10} \\ A_{22}^1 & A_{23}^1 & A_{20} \\ A_{32}^1 & A_{33}^1 & A_{30} \\ & & A_{40} \\ & & A_{50} \\ & & A_{60} \\ A_{20}^T & A_{30}^T & A_{70} \end{bmatrix} \Pi_1^2,$$

and for the third and the fourth subdomains, we have

$$\boldsymbol{A}(:, I_3 \cup I_{34} \cup I_0)\Pi_2^1 = \begin{bmatrix} & & A_{10} \\ & & A_{20} \\ & & A_{30} \\ A_{11}^2 & A_{13}^2 & A_{40} \\ & A_{23}^2 & A_{50} \\ A_{31}^2 & A_{33}^2 & A_{60} \\ A_{40}^T & A_{60}^T & A_{70} \end{bmatrix} \Pi_2^1,$$

$$\boldsymbol{A}(:, I_4 \cup I_{34} \cup I_0)\Pi_2^2 = \begin{bmatrix} & & A_{10} \\ & & A_{20} \\ & & A_{30} \\ & A_{13}^2 & A_{40} \\ A_{22}^2 & A_{23}^2 & A_{50} \\ A_{32}^2 & A_{33}^2 & A_{60} \\ A_{50}^T & A_{60}^T & A_{70} \end{bmatrix} \Pi_2^2.$$

Here $\Pi_i^j,\ i, j = 1:2$ correspond to the permutation matrices determined by strong RRQR factorization. This allows to select the last k columns from each subdomain and the interfaces. Each k selected columns are then concatenated in pairs following a reduction binary tree and strong RRQR factorization is performed until the final set of k columns is selected.

However in this work we do not use a binary reduction tree, the selected columns from the first step are all concatenated and then strong RRQR is performed to select the final set of k columns, known as a flat reduction tree. More specifically, after selecting the last k columns from the matrices formed by $A_{11}^1, A_{22}^1, A_{11}^2, A_{22}^2$ plus the interfaces, we then put all the selected columns together and perform a strong RRQR to select the last k columns. Let $I_{0j},\ j = 1:4$ denote the indices of the k selected columns from each subdomain and the interfaces in the previous step, we have

$$\boldsymbol{A}(:, I_{01} \cup I_{02} \cup I_{03} \cup I_{04})\Pi_b = \begin{bmatrix} A_{10}^1 & A_{10}^2 & A_{10}^3 & A_{10}^4 \\ A_{20}^1 & A_{20}^2 & A_{20}^3 & A_{20}^4 \\ A_{30}^1 & A_{30}^2 & A_{30}^3 & A_{30}^4 \\ A_{40}^1 & A_{40}^2 & A_{40}^3 & A_{40}^4 \\ A_{50}^1 & A_{50}^2 & A_{50}^3 & A_{50}^4 \\ A_{60}^1 & A_{60}^2 & A_{60}^3 & A_{60}^4 \\ A_{70}^1 & A_{70}^2 & A_{70}^3 & A_{70}^4 \end{bmatrix} \Pi_b,$$

where $A_{i0}^p,\ p = 1:4,\ i = 1:7$ denote the selected columns from each pth subdomain plus the interfaces and Π_b denotes the permutation matrix obtained from the strong RRQR factorization. All steps of combining the sparse QR factorization with tournament pivoting strategy and nested dissection on $\boldsymbol{A}^T\boldsymbol{A}$ are described in Algorithm 8.

Algorithm 8: Sparse QR with tournament pivoting and nested dissection to approximate the smallest singular values

Input: matrix $\boldsymbol{A} \in \mathbf{R}^{n\times n}$, number of subdomains p

1 Partition the matrix $\boldsymbol{A}$ using nested dissection on $\boldsymbol{A}^T\boldsymbol{A}$

2 Perform strong RRQR to select the last k columns from each matrix corresponding to each subdomain $\boldsymbol{A}_{ii}^l, i = 1:2, l = 1:p/2$ plus the interfaces, indices of selected columns are denoted by $I_{0j}, 1 \leq j \leq p$

3 Selected columns are concatenated in $\tilde{\boldsymbol{A}} = [\boldsymbol{A}(:, I_{01}) \cdots \boldsymbol{A}(:, I_{0p})]$

4 Perform again strong RRQR of $\tilde{\boldsymbol{A}}$ to select k columns

Output: indices of the last k columns corresponding to the last k smallest singular values of $\boldsymbol{A}$

We refer to Algorithm 8 as sparse QR with tournament pivoting and nested dissection to approximate the smallest singular values. The cost of performing strong RRQR on a matrix formed by each subdomain plus the interfaces in step 2 is $O(nn_hk)$ flops, where $n_h, h = 1:p$ denotes the size of columns corresponding to subdomain $\boldsymbol{A}_{ii}^l$ and the interfaces. We can consider that n_h approaches n/P when the size of the interfaces is much smaller than the size of the subdomains and the subdomains have approximately the same size. The cost of performing strong RRQR of $\tilde{\boldsymbol{A}}$ is $O(npk^2)$. We also remark that the strong RRQR of each subdomain plus the interfaces in step 2 can be performed in parallel. Moreover, strong RRQR factorizations in step 2 can exploit more efficient parallel computing by using a 2D tournament pivoting strategy, see [10]. In the next section we give an analysis of one step of performing the sparse QR with tournament pivoting and nested dissection to approximate the smallest singular values.

4.4.1 Analyzing one step of sparse QR with tournament pivoting and nested dissection to approximate the smallest singular values of A

In this section we perform strong RRQR factorization following Algorithm 8 and Remark 4.2.2 to analyze one step of tournament pivoting strategy with nested dissection partitioning on $\boldsymbol{A}^T\boldsymbol{A}$. For simplicity, we consider again a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ that is partitioned by using nested dissection on $\boldsymbol{A}^T\boldsymbol{A}$ as,

$$\boldsymbol{A} = \begin{bmatrix} A_{11} & & A_{13} \\ & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix},$$

where $A_{ii} \in \mathbb{R}^{n_i\times n_i}$, $i = 1:3$, and $n_1 + n_2 + n_3 = n$. Following Algorithm 8 and Remark 4.2.2, we first perform strong RRQR factorization for the columns corresponding to the first subdomain and the interface,

$$B_1\Pi_1 = \begin{bmatrix} A_{11} & A_{13} \\ & A_{23} \\ A_{31} & A_{33} \end{bmatrix} \Pi_1 = \begin{bmatrix} A_{11}^1 & A_{12}^1 & A_{13}^1 \\ & A_{22}^1 & A_{23}^1 \\ A_{31}^1 & A_{32}^1 & A_{33}^1 \end{bmatrix} = Q_1R_1 = Q_1 \begin{bmatrix} R_{11}^1 & R_{12}^1 \\ & R_{22}^1 \end{bmatrix} = \begin{bmatrix} Q_1^1 & Q_2^1 & Q_3^1 \end{bmatrix} \begin{bmatrix} \tilde{R}_{11}^1 & \tilde{R}_{12}^1 & \tilde{R}_{13}^1 \\ & \tilde{R}_{22}^1 & \tilde{R}_{23}^1 \\ & & \tilde{R}_{33}^1 \end{bmatrix}, \tag{4.7}$$

where the matrix sizes are the following: $A_{11}^1 \in \mathbb{R}^{n_1 \times n_1'}, A_{12}^1 \in \mathbb{R}^{n_1 \times (n_1+n_3-n_1'-k)}, A_{13}^1 \in \mathbb{R}^{n_1 \times k}$ with $n_1 - k \leq n_1' \leq n_1$, $A_{22}^1 \in \mathbb{R}^{n_2 \times (n_1+n_3-n_1'-k)}, A_{23}^1 \in \mathbb{R}^{n_2 \times k}$, $A_{31}^1 \in \mathbb{R}^{n_3 \times n_1'}, A_{32}^1 \in \mathbb{R}^{n_3 \times (n_1+n_3-n_1'-k)}, A_{33}^1 \in \mathbb{R}^{n_3 \times k}$, $Q_1 \in \mathbb{R}^{n \times n}, R_{11}^1 \in \mathbb{R}^{(n-k) \times (n-k)}, R_{12}^1 \in \mathbb{R}^{(n-k) \times k}, R_{22}^1 \in \mathbb{R}^{k \times k}$, and where Π_1 is the permutation matrix from the strong RRQR factorization (4.7) such that for given $F_1 > 1$ and $1 \leq j \leq n_1 + n_3 - k$,

$$\gamma_j^2({R_{11}^1}^{-1} R_{12}^1) + \gamma_j^2(R_{22}^1)/\sigma_{\min}^2(R_{11}^1) \leq F_1^2. \tag{4.8}$$

By identifying $R_{11}^1 \equiv \begin{bmatrix} \tilde{R}_{11}^1 & \tilde{R}_{12}^1 \\ & \tilde{R}_{22}^1 \end{bmatrix}, R_{12}^1 \equiv \begin{bmatrix} \tilde{R}_{13}^1 \\ \tilde{R}_{23}^1 \end{bmatrix}, R_{22}^1 \equiv \tilde{R}_{33}^1$, (4.8) becomes

$$\gamma_j^2(\tilde{R}_{11}^{1\,-1}\tilde{R}_{13}^1 - \tilde{R}_{11}^{1\,-1}\tilde{R}_{12}^1\tilde{R}_{22}^{1\,-1}\tilde{R}_{23}^1) + \gamma_j^2(\tilde{R}_{22}^{1\,-1}\tilde{R}_{23}^1) + \sigma_{\max}^2({R_{11}^1}^{-1})\gamma_j^2(\tilde{R}_{11}^1) \leq F_1^2. \tag{4.9}$$

Similarly, we next perform strong RRQR factorization for the columns corresponding to the second subdomain and the interface,

$$B_2\Pi_2 = \begin{bmatrix} & A_{13} \\ A_{22} & A_{23} \\ A_{32} & A_{33} \end{bmatrix} \Pi_2 = \begin{bmatrix} & A_{12}^2 & A_{13}^2 \\ A_{21}^2 & A_{22}^2 & A_{23}^2 \\ A_{31}^2 & A_{32}^2 & A_{33}^2 \end{bmatrix} = Q_2R_2 = Q_2 \begin{bmatrix} R_{11}^2 & R_{12}^2 \\ & R_{22}^2 \end{bmatrix} = \begin{bmatrix} Q_1^2 & Q_2^2 & Q_3^2 \end{bmatrix} \begin{bmatrix} \tilde{R}_{11}^2 & \tilde{R}_{12}^2 & \tilde{R}_{13}^2 \\ & \tilde{R}_{22}^2 & \tilde{R}_{23}^2 \\ & & \tilde{R}_{33}^2 \end{bmatrix}, \tag{4.10}$$

where the matrix sizes are the following: $A_{21}^2 \in \mathbb{R}^{n_2 \times n_2'}, A_{22}^2 \in \mathbb{R}^{n_2 \times (n_2+n_3-n_2'-k)}, A_{23}^1 \in \mathbb{R}^{n_2 \times k}$ with $n_2 - k \leq n_2' \leq n_2$, $A_{12}^2 \in \mathbb{R}^{n_1 \times (n_2+n_3-n_2'-k)}, A_{13}^2 \in \mathbb{R}^{n_1 \times k}$, $A_{31}^2 \in \mathbb{R}^{n_3 \times n_2'}, A_{32}^2 \in \mathbb{R}^{n_3 \times (n_2+n_3-n_2'-k)}, A_{33}^2 \in \mathbb{R}^{n_3 \times k}$, $Q_2 \in \mathbb{R}^{n \times n}, R_{11}^2 \in \mathbb{R}^{(n-k) \times (n-k)}, R_{12}^2 \in \mathbb{R}^{(n-k) \times k}, R_{22}^2 \in \mathbb{R}^{k \times k}$, and where Π_2 is the permutation matrix from the strong RRQR factorization (4.10) such that for given $F_2 > 1$ and $1 \leq j \leq n_2 + n_3 - k$,

$$\gamma_j^2({R_{11}^2}^{-1} R_{12}^2) + \gamma_j^2(R_{22}^2)/\sigma_{\min}^2(R_{11}^2) \leq F_2^2. \tag{4.11}$$

By identifying $R_{11}^2 \equiv \begin{bmatrix} \tilde{R}_{11}^2 & \tilde{R}_{12}^2 \\ & \tilde{R}_{22}^2 \end{bmatrix}, R_{12}^2 \equiv \begin{bmatrix} \tilde{R}_{13}^2 \\ \tilde{R}_{23}^2 \end{bmatrix}, R_{22}^2 \equiv \tilde{R}_{33}^2$, (4.11) becomes

$$\gamma_j^2(\tilde{R}_{11}^{2\,-1}\tilde{R}_{13}^2 - \tilde{R}_{11}^{2\,-1}\tilde{R}_{12}^2\tilde{R}_{22}^{2\,-1}\tilde{R}_{23}^2) + \gamma_j^2(\tilde{R}_{22}^{2\,-1}\tilde{R}_{23}^2) + \sigma_{\max}^2({R_{11}^2}^{-1})\gamma_j^2(\tilde{R}_{11}^2) \leq F_2^2. \tag{4.12}$$

We then form a new matrix from the last k selected columns of $B_1\Pi_1$ and the last k selected columns of $B_2\Pi_2$ and perform its strong RRQR as,

$$B_3\Pi_3 = \begin{bmatrix} A_{13}^1 & A_{13}^2 \\ A_{23}^1 & A_{23}^2 \\ A_{33}^1 & A_{33}^2 \end{bmatrix} \Pi_3 = \begin{bmatrix} A_{11}^3 & A_{12}^3 \\ A_{21}^3 & A_{22}^3 \\ A_{31}^3 & A_{32}^3 \end{bmatrix} = Q_3R_3 = \begin{bmatrix} Q_1^3 & Q_2^3 \end{bmatrix} \begin{bmatrix} R_{11}^3 & R_{12}^3 \\ & R_{22}^3 \end{bmatrix} = \begin{bmatrix} Q_1^3 & Q_2^3 \end{bmatrix} \begin{bmatrix} \tilde{R}_{11}^3 & \tilde{R}_{12}^3 \\ & \tilde{R}_{22}^3 \\ & \tilde{R}_{32}^3 \end{bmatrix}, \tag{4.13}$$

where $Q_3 \in \mathbb{R}^{n \times 2k}, R_{11}^3 \in \mathbb{R}^{k \times k}, R_{12}^3 \in \mathbb{R}^{k \times k}, R_{22}^3 \in \mathbb{R}^{k \times k}$ and where Π_3 is the permutation matrix from the strong RRQR factorization (4.13) such that for given $F_3 > 1$ and $1 \leq j \leq k$,

$$\gamma_j^2({R_{11}^3}^{-1} R_{12}^3) + \gamma_j^2(R_{22}^3)/\sigma_{\min}^2(R_{11}^3) \leq F_3^2. \tag{4.14}$$

The final set of k selected columns from $B_3\Pi_3$ is permuted to the end and we perform the final permutation of $\boldsymbol{A}$ and its QR factorization as,

$$\boldsymbol{A}\Pi = \begin{bmatrix} A_{11}^1 & & A_{13}^4 & A_{11}^3 & A_{12}^3 \\ & A_{21}^2 & A_{23}^4 & A_{21}^3 & A_{22}^3 \\ A_{31}^1 & A_{31}^2 & A_{33}^4 & A_{31}^3 & A_{32}^3 \end{bmatrix} = Q \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix} = Q_1^1 Q_1^2 \bar{Q}_3 Q_2^3 \begin{bmatrix} \bar{R}_{11} & & \bar{R}_{13} & \bar{R}_{14} \\ & \bar{R}_{22} & \bar{R}_{23} & \bar{R}_{24} \\ & & \bar{R}_{33} & \bar{R}_{34} \\ & & & \bar{R}_{44} \end{bmatrix}, \tag{4.15}$$

where $\begin{bmatrix} A_{13}^4 \\ A_{23}^4 \\ A_{33}^4 \end{bmatrix} \equiv \begin{bmatrix} A_{12}^1 & A_{12}^2 \\ A_{22}^1 & A_{22}^2 \\ A_{32}^1 & A_{32}^2 \end{bmatrix}$, $\bar{Q}_3 \equiv \begin{bmatrix} Q_2^1 & Q_2^2 & Q_1^3 \end{bmatrix}$, $R_{22} \equiv \bar{R}_{44} \in \mathbb{R}^{k\times k}$.

To be able to show that this strategy allows to approximate the smallest singular values of $\boldsymbol{A}$, we would need to prove that

$$\gamma_j^2({R_{11}}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq F^2, \tag{4.16}$$

where $F > 1$ and $1 \leq j \leq n-k$, or by identifying $R_{11} \equiv \begin{bmatrix} \bar{R}_{11} & & \bar{R}_{13} \\ & \bar{R}_{22} & \bar{R}_{23} \\ & & \bar{R}_{33} \end{bmatrix}$, $R_{12} \equiv \begin{bmatrix} \bar{R}_{14} \\ \bar{R}_{24} \\ \bar{R}_{34} \end{bmatrix}$ and $R_{22} \equiv \bar{R}_{44}$, (4.16) is equivalent to

$$\gamma_j^2(\bar{R}_{11}^{-1}\bar{R}_{14} - \bar{R}_{11}^{-1}\bar{R}_{13}\bar{R}_{33}^{-1}\bar{R}_{34}) + \gamma_j^2(\bar{R}_{22}^{-1}\bar{R}_{24} - \bar{R}_{22}^{-1}\bar{R}_{23}\bar{R}_{33}^{-1}\bar{R}_{34}) + \gamma_j^2(\bar{R}_{33}^{-1}\bar{R}_{34}) + \sigma_{\max}^2(R_{11}^{-1})\gamma_j^2(\bar{R}_{44}) \leq F^2. \tag{4.17}$$

We were not able to prove this inequality for the moment. However given the sparsity of A and the nested dissection ordering, it is possible that this inequality is satisfied for an F which depends on n and k.

4.5 Deflation preconditioner

In this section we introduce a preconditioner that allows to deflate the smallest singular values. As described in the previous section, by applying Algorithm 8, the matrix $\boldsymbol{A}$ is first partitioned by using nested dissection on $\boldsymbol{A}^T\boldsymbol{A}$ and then sparse QR with tournament pivoting and nested dissection is used to determine a permutation that allows to approximate the k smallest singular values of A. We denote by P_{ND} the nested dissection permutation matrix from $\boldsymbol{A}^T\boldsymbol{A}$, and Π the permutation matrix from sparse QR factorization with tournament pivoting strategy. For any $f > 1$, consider that the QR factorization computes a decomposition

$$P_{ND}^T \boldsymbol{A} P_{ND} \Pi = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}, \tag{4.18}$$

where Π is such that for any $1 \leq i \leq n-k,\ 1 \leq j \leq k$

$$(R_{11}^{-1}R_{12})_{ij}^2 + \gamma_j^2(R_{22})/\omega_i^2(R_{11}) \leq f^2, \tag{4.19}$$

or we can rewrite it as the relaxed form as in (4.5) by summing over i

$$\gamma_j^2(R_{11}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq (n-k)f^2, \tag{4.20}$$

where $\gamma_j(R_{22})$ denotes the 2-norm of the jth column of R_{22}, $1/\omega_i(R_{11})$ denotes the 2-norm of the ith row of R_{11}^{-1} and $(R_{11}^{-1}R_{12})_{i,j}$ is the element at row ith and column jth of the matrix $R_{11}^{-1}R_{12}$. The singular values of R_{11} in (4.18) can be considered as good approximations of the largest singular values of $\boldsymbol{A}$ and the singular values of R_{22} can be considered as good approximations of the smallest singular values of $\boldsymbol{A}$, see [19]. Our goal is to define a deflation preconditioner which allows to replace the smallest singular values close to zero by much larger ones, so that GMRES is not affected by those small singular values and converges faster. In particular, we build a deflation preconditioner from Q_2, R_{22} in (4.18) as,

$$M^{-1} = (I_n - Q_2Q_2^T) + Q_2R_{22}^{-1}Q_2^T. \tag{4.21}$$

Let $\{\sigma_1(\boldsymbol{A}), \ldots, \sigma_n(\boldsymbol{A})\}$ denote the singular values of $\boldsymbol{A}$ such that $\sigma_1(\boldsymbol{A}) \geq \cdots \geq \sigma_n(\boldsymbol{A})$. The following Theorem 4.5.1 shows that the preconditioner M^{-1} defined in (4.21) can deflate the smallest singular values of $P_{ND}^T\boldsymbol{A}P_{ND}\Pi$.

Theorem 4.5.1. *Given $f > 1$, by taking $\tilde{f} = \max_{1\leq j\leq k}\{f, \frac{f}{\gamma_j(R_{22})}\}$, where $\gamma_j(R_{22})$ denotes the 2-norm of the jth column of R_{22}, the singular values of the matrix $M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}\Pi$, where M^{-1} is from* (4.21) *and $P_{ND}^T\boldsymbol{A}P_{ND}\Pi$ is from the strong RRQR factorization* (4.18) *in which P_{ND} denotes the permutation from nested dissection and Π denotes the permutation of strong RRQR, are bounded by*

$$\begin{cases} \sigma_i(M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}) & \leq \sigma_i(R_{11})\sqrt{1+\tilde{f}^2k(n-k)}, \quad 1 \leq i \leq n-k, \\ \sigma_{n-k+j}(M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}) & \geq 1/\sqrt{1+\tilde{f}^2k(n-k)}, \quad 1 \leq j \leq k. \end{cases} \tag{4.22}$$

Proof. The strong RRQR factorization of $\boldsymbol{A}$ with permutation matrices P_{ND} and Π in (4.18) can be rewritten as

$$P_{ND}^T\boldsymbol{A}P_{ND}\Pi = Q_1\begin{bmatrix} R_{11} & R_{12}\end{bmatrix} + Q_2\begin{bmatrix} 0 & R_{22}\end{bmatrix}, \tag{4.23}$$

where Π is such that for any $1 \leq i \leq n-k,\ 1 \leq j \leq k$,

$$(R_{11}^{-1}R_{12})_{ij}^2 + \gamma_j^2(R_{22})/\omega_i^2(R_{11}) \leq f^2, \tag{4.24}$$

which is equivalent to

$$\max_{1\leq i\leq n-k, 1\leq j\leq k} \sqrt{(R_{11}^{-1}R_{12})_{ij}^2 + (\gamma_j(R_{22})/\omega_i(R_{11}))^2} \leq f. \tag{4.25}$$

By applying the deflation preconditioner M^{-1} defined in (4.21) we obtain

$$\begin{aligned} M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}\Pi &= (I_n - Q_2Q_2^T)\boldsymbol{A}\Pi + Q_2R_{22}^{-1}Q_2^T\boldsymbol{A}\Pi = Q_1\begin{bmatrix} R_{11} & R_{12}\end{bmatrix} + Q_2R_{22}^{-1}\begin{bmatrix} 0 & R_{22}\end{bmatrix} \\ &= Q_1\begin{bmatrix} R_{11} & R_{12}\end{bmatrix} + Q_2\begin{bmatrix} 0 & I_k\end{bmatrix} = \begin{bmatrix} Q_1 & Q_2\end{bmatrix}\begin{bmatrix} R_{11} & R_{12} \\ & I_k\end{bmatrix}. \end{aligned}$$

Now that R_{22} is replaced by identity, we define $R \equiv \begin{bmatrix} R_{11} & R_{12} \\ & I_k \end{bmatrix}$ and we need to find an $\tilde{f}$ such that

$$\rho(R,k) = \max_{1\le i\le n-k, 1\le j\le k} \sqrt{(R_{11}^{-1}R_{12})_{ij}^2 + (1/\omega_i(R_{11}))^2} \le \tilde{f}.$$

The condition $\max_{1\le i\le n-k,1\le j\le k} \sqrt{(R_{11}^{-1}R_{12})_{ij}^2 + (\gamma_j(R_{22})/\omega_i(R_{11}))^2} \le f$ is required in [48, Theorem 3.2] to obtain the bound on strong RRQR factorization. We apply [48, Theorem 3.2] to obtain the results on the bounds of the singular values which correspond to equations (8) and (9) in [48, Theorem 3.2],

$$\sigma_i(A_k) \ge \frac{\sigma_i(\boldsymbol{M})}{\sqrt{1+f^2k(n-k)}}, \quad 1 \le i \le k,$$

and

$$\sigma_j(C_k) \le \sigma_{j+k}(\boldsymbol{M})\sqrt{1+f^2k(n-k)}, \quad 1 \le j \le n-k.$$

By identifying $\boldsymbol{M} \equiv M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}$, $A_k \equiv R_{11}$, $B_k \equiv R_{12}$ and $C_k \equiv I_k$, we derive the matrix $R = \begin{bmatrix} R_{11} & R_{12} \\ & I_k \end{bmatrix}$ in our context which satisfies the condition $\rho(R,k) \le \tilde{f}$, with $\tilde{f} = \max_{1\le j\le k}\{f, \frac{f}{\gamma_j(R_{22})}\}$, then the bounds are obtained directly from [48, Theorem 3.2] as,

$$\sigma_i(R_{11}) \ge \frac{\sigma_i(M^{-1}P_{ND}^T\boldsymbol{A}P_{ND})}{\sqrt{1+\tilde{f}^2k(n-k)}}, \quad 1 \le i \le n-k, \tag{4.26}$$

and

$$\sigma_j(I_k) \le \sigma_{n-k+j}(M^{-1}P_{ND}^T\boldsymbol{A}P_{ND})\sqrt{1+\tilde{f}^2k(n-k)}, \quad 1 \le j \le k. \tag{4.27}$$

Hence we can deduce from (4.26),(4.27) the bound for the $n-k$ largest singular values of $M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}$ as

$$\sigma_i(M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}) \le \sigma_i(R_{11})\sqrt{1+\tilde{f}^2k(n-k)}, \quad 1 \le i \le n-k, \tag{4.28}$$

and the bound for the k smallest singular values of $M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}$ as

$$\sigma_{n-k+j}(M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}) \ge \frac{1}{\sqrt{1+\tilde{f}^2k(n-k)}}, \quad 1 \le j \le k. \tag{4.29}$$

□

We can see from 4.29 that the smallest singular values of $M^{-1}P_{ND}^T\boldsymbol{A}P_{ND}$ are larger than $1/\sqrt{1+\tilde{f}^2k(n-k)}$. This can be considered as a very good bound for the deflation in practical problems since it is much larger than the smallest singular values of $\boldsymbol{A}$.

Lemma 4.5.1. *The bound in* (4.29) *can be improved by a factor of* $\sigma_1(\boldsymbol{A})$*, where* $\sigma_1(\boldsymbol{A})$ *is the largest singular value of* $\boldsymbol{A}$*. In this case, the deflation preconditioner in* (4.21) *is formulated as,*

$$M^{-1} = (I_n - Q_2 Q_2^T) + Q_2 \sigma_1(\boldsymbol{A}) R_{22}^{-1} Q_2^T, \tag{4.30}$$

and the bound for the k *smallest singular values of* $M^{-1} P_{ND}^T \boldsymbol{A} P_{ND}$ *becomes,*

$$\sigma_{n-k+j}(M^{-1} P_{ND}^T \boldsymbol{A} P_{ND}) \geq \frac{\sigma_1(\boldsymbol{A})}{\sqrt{1 + \tilde{f}^2 k(n-k)}}, \quad 1 \leq j \leq k. \tag{4.31}$$

Proof. The proof is similar to the proof in Theorem 4.5.1. □

Lemma 4.5.1 is useful in the case when the largest singular value of $\boldsymbol{A}$ is known or is well approximated and it is is very large. In that case it could be used to increase the lower bound of the smallest singular values as in equation (4.31). However we do not explore this option further in this work.

In the following discussion we show how to apply the deflation preconditioner M^{-1} defined in (4.21) to the original system (4.32). Let $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$, we want to solve the linear system

$$\boldsymbol{A} x = b. \tag{4.32}$$

As a consequence of Theorem 4.5.1, the following Lemma 4.5.2 shows how the preconditioner M^{-1} can deflate the smallest singular values of $\boldsymbol{A}$.

Lemma 4.5.2. *The singular values of the matrix* $\tilde{M}^{-1}\boldsymbol{A}$ *are bounded by*

$$\begin{cases} \sigma_i(\tilde{M}^{-1}\boldsymbol{A}) & \leq & \sigma_i(R_{11})\sqrt{1 + \tilde{f}^2 k(n-k)}, & 1 \leq i \leq n-k, \\ \sigma_{n-k+j}(\tilde{M}^{-1}\boldsymbol{A}) & \geq & 1/\sqrt{1 + \tilde{f}^2 k(n-k)}, & 1 \leq j \leq k, \end{cases} \tag{4.33}$$

where $\tilde{M}^{-1} = M^{-1} P_{ND}^T$ *and* $\tilde{f}$ *defined in Theorem 4.5.1.*

Proof. Given that P_{ND} and Π are orthonormal permutation matrices, it is obvious that $\sigma_i(P_{ND}^T \boldsymbol{A} P_{ND} \Pi) = \sigma_i(\boldsymbol{A}),\ 1 \leq i \leq n$. From Theorem 4.5.1, we have

$$M^{-1} P_{ND}^T \boldsymbol{A} P_{ND} \Pi \;=\; \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & I_k \end{bmatrix}, \tag{4.34}$$

or it can be rewritten as,

$$\tilde{M}^{-1}\boldsymbol{A} \;=\; \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & I_k \end{bmatrix} \Pi^T P_{ND}^T. \tag{4.35}$$

The rest of the proof is similar to the proof in Theorem 4.5.1. □

The original system (4.32) can now be preconditioned by multiplying both sides by $\tilde{M}^{-1}$ from Remark 4.5.2, which yields

$$\tilde{M}^{-1}\boldsymbol{A}x = \tilde{M}^{-1}b. \tag{4.36}$$

We can see from Remark 4.5.2 that $\tilde{M}^{-1}$ also deflates the smallest singular values of $\boldsymbol{A}$. This can help Krylov subspace methods like GMRES avoid suffering from the bad convergence behavior due to the impact of those smallest singular values in the spectrum.

Remark 4.5.2. *The deflation preconditioner $\tilde{M}^{-1}$ in* (4.36) *is obtained from $M^{-1}P_{ND}^T$ which requires Q_2 and R_{22} in* (4.18)*. Tournament pivoting allows to efficiently compute a permutation that allows to separate the largest singular values from the small ones that are captured by R_{22}. However once this permutation is known, in our current work we still require to perform the QR factorization of a permuted matrix $P_{ND}^T A P_{ND}\Pi$. The overall cost for obtaining Q_2 and R_{22} to build the deflation preconditioner $\tilde{M}^{-1}$ is thus expensive. In our future work, we plan to combine this with domain decomposition approaches in which a coarse space would be constructed from contributions from each subdomain. This would allow to require computing the QR factorization of each subdomain, and this could be performed in parallel.*

4.6 Deflation and block Jacobi combination

In this section we combine the deflation preconditioner with the block Jacobi preconditioner. In the context of preconditioning, together with SSOR, incomplete Cholesky factorization, incomplete LU factorization, multigrid preconditioning, block Jacobi preconditioner is one of the most used preconditioners thanks to its simplicity in implementation and its effectiveness in parallel computing. For an overview as well as some of its practical applications, see [2, 47, 57, 89]. We briefly recall that block Jacobi preconditioner is well known as the simplest preconditioner which exploits parallel computing and communication avoiding. Relying on the multiplication $y = P^{-1}\boldsymbol{A}x$, where

$$M_{BJ} = \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_p \end{bmatrix} = \begin{bmatrix} L_1U_1 & & & \\ & L_2U_2 & & \\ & & \ddots & \\ & & & L_pU_p \end{bmatrix}, \tag{4.37}$$

each block-diagonal of $\boldsymbol{A}$ can be factorized using LU factorization as, $A_i = L_iU_i, i = 1 : p$. Each A_i can be assigned to corresponding processor ith and the corresponding unknowns can be solved in parallel through the following expression,

$$M_{BJ}^{-1} = \sum_{i=1}^{p} R_i^T (R_i \boldsymbol{A} R_i^T)^{-1} R_i, \tag{4.38}$$

where $R_i \in \mathbb{R}^{n\times N}$ denotes the restriction matrix corresponding to the unknowns in block-diagonal ith and $R_i\boldsymbol{A}R_i^T \in \mathbb{R}^{n\times n}$ stands for the diagonal block A_i of $\boldsymbol{A}$ which corresponds to

the unknowns on the processor ith. It can be seen from (4.38) that the unknowns from each diagonal block ith, $R_i \boldsymbol{A} R_i^T$, can be solved independently in parallel and then be prolongated to the global unknowns through the operator $R_i^T (R_i \boldsymbol{A} R_i^T)^{-1} R_i$.

In particular in this work, the matrix $\boldsymbol{A}$ is first partitioned into diagonal blocks by performing the k-way partitioning using METIS library, see [63]. We briefly recall k-way partition from [63], which relies on the recursive bisection. First the bisection is applied to the original graph of the matrix to obtain a 2-way partitioning. Then, it is recursively applied to each of the resulting partition. The k partitions of the original graph are obtained after $\log k$ steps. To that goal, the restriction matrix R_i in (4.38) is chosen to be the k-way partitioning permutation matrix corresponding to unknowns on the diagonal block A_i.

In the following discussion we present the combination between the deflation preconditioner $\tilde{M}^{-1}$ from (4.36) and the block Jacobi preconditioner M_{BJ}^{-1} from (4.38). Consider again the linear system (4.32), we first apply the block Jacobi preconditioner M_{BJ}^{-1} from (4.38) and then the deflation preconditioner $\tilde{M}^{-1}$ from (4.36), which yields

$$\tilde{M}^{-1} M_{BJ}^{-1} \boldsymbol{A} x = \tilde{M}^{-1} M_{BJ}^{-1} b. \tag{4.39}$$

The combination results in a mixed preconditioner which gives faster convergence for GMRES since the smallest singular values are deflated and the condition number becomes smaller. In the next section we present the deflation of singular vectors based on strong RRQR factorization of a general invertible matrix.

4.7 Deflation of singular vectors based on strong RRQR factorization

Apart from the deflation preconditioner, in this section we present the theory of deflation of singular vectors for a general invertible matrix which is motivated by the work in [1, 99]. By applying Algorithm 8, we first obtained the matrix $\boldsymbol{A}$ with the last k columns corresponding to the k smallest singular values, then the deflation is described in the following Theorem 4.7.1,

Theorem 4.7.1. *Let x_* be the exact solution of $\boldsymbol{A}x = b$. Consider the QR factorization of $\boldsymbol{A}$ with the last k columns corresponding to the k smallest singular values,*

$$\boldsymbol{A}\Pi = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}, \tag{4.40}$$

where Π is such that for all $(i, j) \in [1, n-k] \times [1, k]$,

$$\gamma_j^2(R_{11}^{-1} R_{12}) + \gamma_j^2(R_{22}) / \sigma_{\min}^2(R_{11}) \leq (n-k) f^2. \tag{4.41}$$

If $||R_{22}||_2$ is small, then

$$\tilde{V}_2 = \Pi \begin{bmatrix} -R_{11}^{-1} R_{12} \\ I_k \end{bmatrix} \tag{4.42}$$

is an approximate right null space of $\boldsymbol{A}$ *where* $\tilde{V}_2 \in \mathbb{R}^{n\times k}, \Pi \in \mathbb{R}^{n\times n}, R_{11} \in \mathbb{R}^{(n-k)\times(n-k)}, R_{12} \in \mathbb{R}^{(n-k)\times k}, R_{22} \in \mathbb{R}^{k\times k}, Q_1 \in \mathbb{R}^{n\times(n-k)}, Q_2 \in \mathbb{R}^{n\times k}$ *and* I_n, I_k *denote the identity matrices of order* n, k *respectively.*

Consider $\tilde{x}$ *an approximate solution of the following linear system of equations*

$$(I_n - Q_2Q_2^T)\boldsymbol{A}x = (I_n - Q_2Q_2^T)b, \tag{4.43}$$

such that $||\hat{x} - \tilde{x}||_2 \leq \epsilon$*, where* $\hat{x}$ *is an exact solution of* (4.43) *and* $\epsilon > 0$*. Then, the following holds*

$$||x_* - (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T)\tilde{x} - \tilde{V}_2R_{22}^{-1}Q_2^Tb||_2 \leq \epsilon. \tag{4.44}$$

Proof. We have

$$\begin{aligned} \boldsymbol{A}\tilde{V}_2 &= Q_2R_{22} \\ \tilde{V}_2 &= \boldsymbol{A}^{-1}Q_2R_{22} \\ \tilde{V}_2R_{22}^{-1} &= \boldsymbol{A}^{-1}Q_2. \end{aligned} \tag{4.45}$$

We have

$$\begin{aligned} \boldsymbol{A}\Pi &= Q_1 \begin{bmatrix} R_{11} & R_{12} \end{bmatrix} + Q_2 \begin{bmatrix} 0 & R_{22} \end{bmatrix} \\ R_{22}^{-1}Q_2^T\boldsymbol{A}\Pi &= R_{22}^{-1}Q_2^T(Q_1 \begin{bmatrix} R_{11} & R_{12} \end{bmatrix} + Q_2 \begin{bmatrix} 0 & R_{22} \end{bmatrix}) \\ R_{22}^{-1}Q_2^T\boldsymbol{A}\Pi &= \begin{bmatrix} 0 & I_k \end{bmatrix} \\ R_{22}^{-1}Q_2^T\boldsymbol{A} &= \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T \\ R_{22}^{-1}Q_2^T &= \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T\boldsymbol{A}^{-1}. \end{aligned} \tag{4.46}$$

We remark that x_* is a solution of (4.43) and the set of solutions of (4.43) can be written as $S = \{x = x_* + \tilde{V}_2u,\ u \in \mathbb{R}^k\}$. Indeed, let x be a solution of (4.43), we have

$$\begin{aligned} \boldsymbol{A}x &= (I_n - Q_2Q_2^T)b + Q_2Q_2^T\boldsymbol{A}x \\ \boldsymbol{A}x &= \boldsymbol{A}x_* - Q_2Q_2^T(b - \boldsymbol{A}x), \end{aligned} \tag{4.47}$$

multiplying both sides of (4.47) by $\boldsymbol{A}^{-1}$ and using (4.45), (4.46) lead to

$$\begin{aligned} x &= x_* + \boldsymbol{A}^{-1}Q_2Q_2^T(\boldsymbol{A}x - b) \\ &= x_* + \tilde{V}_2R_{22}^{-1}Q_2^T(\boldsymbol{A}x - b) \\ &= x_* + \tilde{V}_2R_{22}^{-1}Q_2^T\boldsymbol{A}x - \tilde{V}_2R_{22}^{-1}Q_2^Tb \\ &= x_* + \tilde{V}_2R_{22}^{-1}Q_2^T\boldsymbol{A}\Pi\Pi^Tx - \tilde{V}_2R_{22}^{-1}Q_2^Tb \\ &= x_* + \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^Tx - \tilde{V}_2R_{22}^{-1}Q_2^Tb. \end{aligned} \tag{4.48}$$

On the opposite direction, suppose that $x = x_* + \tilde{V}_2u,\ u \in \mathbb{R}^k$, we prove that x is a solution of (4.43).

$$\begin{aligned} (I_n - Q_2Q_2^T)\boldsymbol{A}x &= (I_n - Q_2Q_2^T)b \\ (I_n - Q_2Q_2^T)\boldsymbol{A}(x_* + \tilde{V}_2u) &= (I_n - Q_2Q_2^T)b \\ (I_n - Q_2Q_2^T)\boldsymbol{A}x_* + (I_n - Q_2Q_2^T)\boldsymbol{A}\tilde{V}_2u &= (I_n - Q_2Q_2^T)b \\ (I_n - Q_2Q_2^T)\boldsymbol{A}x_* + (I_n - Q_2Q_2^T)Q_2R_{22}u &= (I_n - Q_2Q_2^T)b. \end{aligned} \tag{4.49}$$

On the other hand, thanks to (4.46) we have

$$\begin{aligned} \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T x_* &= \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T \mathbf{A}^{-1} b \\ &= \tilde{V}_2 R_{22}^{-1} Q_2^T b. \end{aligned} \tag{4.50}$$

Hence, we can write

$$\begin{aligned} x_* &= (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T) x_* + \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T x_* \\ &= (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T) \hat{x} + \tilde{V}_2 R_{22}^{-1} Q_2^T b, \end{aligned} \tag{4.51}$$

thus

$$x_* - (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T) \tilde{x} - \tilde{V}_2 R_{22}^{-1} Q_2^T b = (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T)(\hat{x} - \tilde{x}). \tag{4.52}$$

Finally, we obtain

$$\begin{aligned} ||x_* - (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T) \tilde{x} - \tilde{V}_2 R_{22}^{-1} Q_2^T b||_2 &\leq ||(I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T)(\hat{x} - \tilde{x})||_2 \\ &\leq ||\hat{x} - \tilde{x}||_2 \\ &\leq \epsilon. \end{aligned} \tag{4.53}$$

□

So the approximate solution to $\boldsymbol{A}x = b$ can be recovered as

$$x = (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T) \tilde{x} + \tilde{V}_2 R_{22}^{-1} Q_2^T b, \tag{4.54}$$

and $\tilde{x}$ is the approximate solution to (4.43) given by the Krylov method.

4.8 Numerical results

In this section we conduct numerical experiments to illustrate the proposed deflation technique on the convergence of various linear systems arising from PDEs discretizations such as fluid mechanics, oil reservoir modeling, finite element modeling. We remark that for the nested dissection partition on $\boldsymbol{A}^T\boldsymbol{A}$, we set the number of subdomain to 8 for all test cases. Table 4.1 gives information about the matrices used in our tests. For all test cases, the right-hand side b is chosen randomly with the normalization $b = \frac{b}{||b||_2}$.

4.8.1 Strong RRQR deflated GMRES and SVD deflated GMRES comparison

The first test matrix in this section is the e20r0100 matrix arsing from the 2D fluid flow modeling in a driven cavity with the Reynold number $Re = 100$, it is non-symmetric and indefinite and not easy to solve using iterative methods like preconditioned Krylov subspace methods since it is difficult to find an effective preconditioner.

In Figure 4.1 we show the impact of deflating the smallest singular values based on the two different methods, the strong RRQR deflated GMRES and the SVD deflated GMRES. In each method, 50 vectors are computed approximately to a tolerance of 10^{-6}. The deflation

Number	Name	Size	Description
1	e20r0100	4241 × 4241	2D fluid flow in a driven cavity
2	e20r5000	4241 × 4241	2D fluid flow in a driven cavity
3	e30r5000	9661 × 9661	2D fluid flow in a driven cavity
4	e40r5000	17281 × 17281	2D fluid flow in a driven cavity
5	fidap012	3973 × 3973	Finite element modeling
6	bcsstk24	3562 × 3562	Dynamic analyses in structural engineering
7	bcsstk35	30237 × 30237	Stiffness matrix, automobile seat frame and body attachment
8	crystk03	24696 × 24696	Stiffness matrix, FEM crystal free vibration
9	saylr4	3564 × 3564	Oil reservoir modeling
10	sherman3	5005 × 5005	Oil reservoir modeling
11	lshp3025	3025 × 3025	Finite element model problem
12	lshp3466	3466 × 3466	Finite element model problem
13	msc01440	1440 × 1440	Structural Problem

Table 4.1: Test matrices.

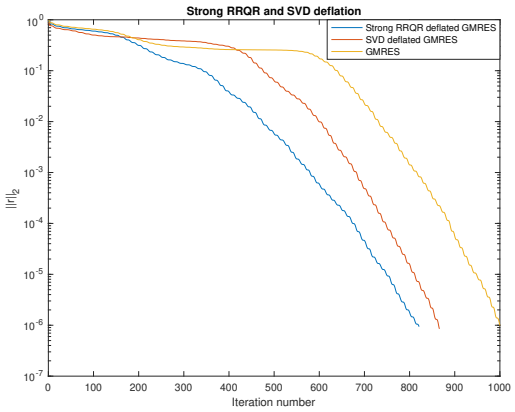

Figure 4.1: Convergence history of singular vectors based on strong RRQR and SVD deflated GMRES. Results for non-deflated GMRES is also plotted, for the matrix e20r0100 in MATLAB 2017, $Re = 100$. 50 singular vectors are approximated with a tolerance of 10^{-6}.

of the singular vectors based on SVD following [1], recalled in section 1.7 as it can be seen in the preconditioned linear system

$$(I - U_2 U_2^T)\boldsymbol{A}x = (I - U_2 U_2^T)b, \tag{4.55}$$

where $U_2 = \boldsymbol{A}V_2\Sigma_2^{-1}$ denotes the approximated left singular vectors. The solution is approximated by $x = (I - V_2 V_2^T)\bar{x} + V_2 x_2$, where $\bar{x}$ denotes the solution of (4.55) obtained by GMRES and $x_2 = \Sigma_2^{-1} U_2^T b$. The deflation of the singular vectors based on the strong RRQR factorization as introduced in section 4.7 that we can rewrite,

$$(I_n - Q_2 Q_2^T)\boldsymbol{A}x = (I_n - Q_2 Q_2^T)b, \tag{4.56}$$

where $Q_2 = \boldsymbol{A}\tilde{V}_2 R_{22}^{-1}$ is the approximated singular vectors. The solution is recovered as $x = (I_n - \tilde{V}_2 \begin{bmatrix} 0 & I_k \end{bmatrix} \Pi^T)\tilde{x} + \tilde{V}_2 \tilde{x}_2$, where $\tilde{x}$ denotes the solution of (4.56) obtained by GMRES and $\tilde{x}_2 = R_{22}^{-1} Q_2^T b$. We observe that the convergence rate for both deflation techniques are approximately the same, however the strong RRQR variant is slightly faster. In particular in the first iterations, GMRES and the SVD variant stagnate while the strong RRQR variant has a sharper convergence curve.

The second test matrix we consider in this section is the e20r5000 matrix from the same problem with the first one but with larger Reynold number, $Re = 5000$. We observe from Figure 4.2 that both methods suffer from a long stagnation and converge to the desired tolerance of 10^{-6} after more than 3500 iterations with a slightly faster rate for the SVD deflated GMRES curve.

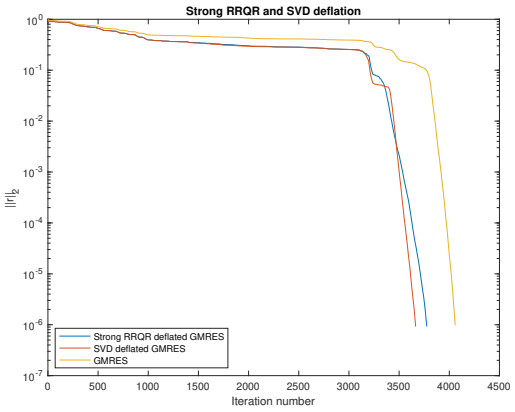

Figure 4.2: Convergence history of singular vectors based on strong RRQR and SVD deflated GMRES. Results for non-deflated GMRES is also plotted, for the matrix e20r5000 in MATLAB 2017, $Re = 5000$. 50 singular vectors are approximated with a tolerance of 10^{-6}.

In the next test cases we consider the fidap012 matrix arising from the finite element modeling generated by the FIDAP package and the bcsstk24 matrix from dynamic analyses in structural engineering. We observe in Figure 4.3 that GMRES almost stagnates while the two deflation technique curves are quite sharp with a small advantage for the SVD deflated GMRES curve.

For the final test cases in this section, we consider the bcsstk35 matrix arising from the automobile seat frame and body attachment problem and the crystk03 matrix from the finite element method of crystal free vibration problem. It can be seen from Figure 4.4 that GMRES totally stagnates, on the contrary, the two deflation technique curves converge quite fast with a faster rate for the SVD deflated GMRES curve.

The deflation technique based on sparse QR with tournament pivoting and nested dissection deflates the approximations of the smallest singular values of $\boldsymbol{A}$, hence it is not as precise as the deflation based on SVD. However, the trade-off is considerable in the sense that we can exploit parallel computing with tournament pivoting strategy and reduce the computational cost. In particular for dense matrices, the cost for the strong

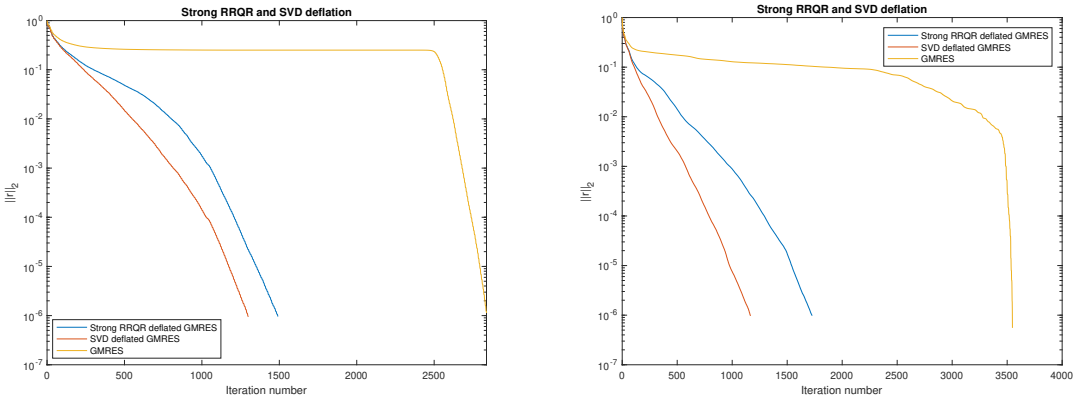

Figure 4.3: Convergence history of singular vectors based on strong RRQR and SVD deflated GMRES. Results for the matrix fidap012 (left) and bcsstk24 (right) in MATLAB 2017. 50 singular vectors are approximated with a tolerance of 10^{-6}.

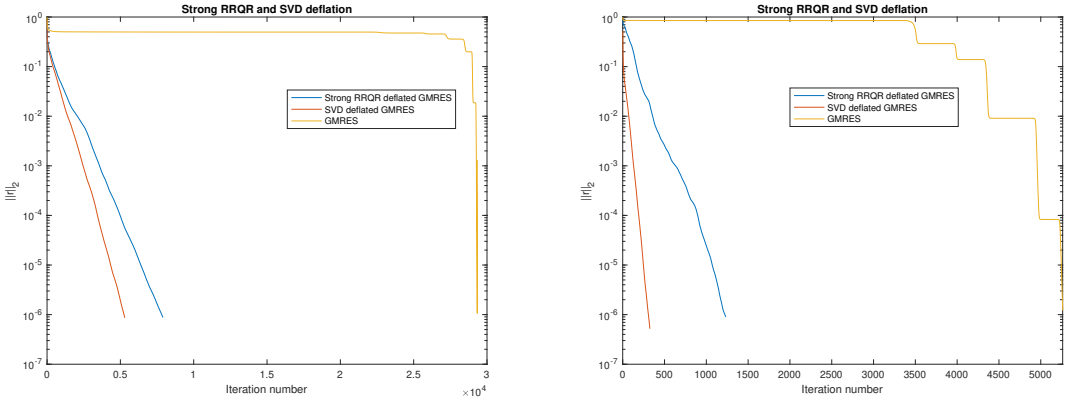

Figure 4.4: Convergence history of singular vectors based on strong RRQR and SVD deflated GMRES. Results for the matrix bcsstk35 (left) and crystk03 (right) in MATLAB 2017. 50 singular vectors are approximated with a tolerance of 10^{-6}.

RRQR factorization $\boldsymbol{A}\Pi = QR$ is $\mathcal{O}(n^3)$ flops in the worst case, as stated in [48], while the overall cost of SVD is $\mathcal{O}(13n^3)$ flops, as stated in [100], for a square dense matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$.

4.8.2 Deflation of smallest singular values approximation by sparse QR with tournament pivoting and nested dissection

In section 4.7 and 4.8.1, we present the deflation of singular vectors based on strong RRQR factorization. By applying Algorithm 8, the matrix $\boldsymbol{A}$ is first partitioned by using nested dissection on $\boldsymbol{A}^T\boldsymbol{A}$ then sparse QR with tournament pivoting are performed to obtain a matrix with the last k columns corresponding to the k smallest singular values. From the QR factorization of that permuted matrix, $Q_2 = \boldsymbol{A}\tilde{V}_2 R_{22}^{-1}$, the approximated singular

vectors, and $\tilde{V}_2 = \Pi \begin{bmatrix} -R_{11}^{-1}R_{12} \\ I_k \end{bmatrix}$, the approximate right null space are used for the deflation of singular vectors as it can be seen in equation (4.56). This allows us to compare deflated GMRES based on strong RRQR with tournament pivoting and deflated GMRES based on SVD which was described in [1]. The deflation preconditioner from section 4.5, 4.8.2 and 4.8.3 is built from Q_2 and R_{22} of the QR factorization of the matrix $\boldsymbol{A}$ with the last k columns corresponding to the k smallest singular values, as $M^{-1} = (I_n - Q_2Q_2^T) + Q_2R_{22}^{-1}Q_2^T$. This deflation preconditioner can be combined with other preconditioners such as block Jacobi.

More specifically, in this section we show in the numerical experiments the approximation of the smallest singular values of $\boldsymbol{A}$ using sparse QR with tournament pivoting strategy and nested dissection on $\boldsymbol{A}^T\boldsymbol{A}$ and then the deflation results. We are interested in the cases where there is a large gap among the smallest singular values of $\boldsymbol{A}$. For that reason, we consider the two test matrices crystk03 which is the crystal free vibration stiffness matrix from finite element method and bcsstk35 which is the stiffness matrix from automobile seat frame and body attachment. We remark that if the ratio between the maximum and the minimum elements of K is less than some threshold, for example we take 0.1, then the matrix is scaled by $\frac{a_{ij}}{\sqrt{K_iK_j}}$, where K is a vector containing the maximum element on each row and a_{ij} is the element at row ith and column jth of $\boldsymbol{A}$, $1 \leq i, j \leq n$.

We show in Figure 4.5 the approximation of smallest singular values of $\boldsymbol{A}$ and the impact of the deflation preconditioner for the two test cases, the crystk03 matrix arising from finite element method for crystal free vibration and the bcsstk35 matrix from the automobile seat frame and body attachment problem. We observe that the smallest singular of R_{22} approximate quite well the smallest singular values of $\boldsymbol{A}$, in particular the error is of order one for crystk03 and less than one for bcsstk35. The smallest singular values of $\boldsymbol{A}$ are replaced by larger ones in both test cases, especially for the last 6 smallest singular values, under the impact of the deflation preconditioner $\tilde{M}$.

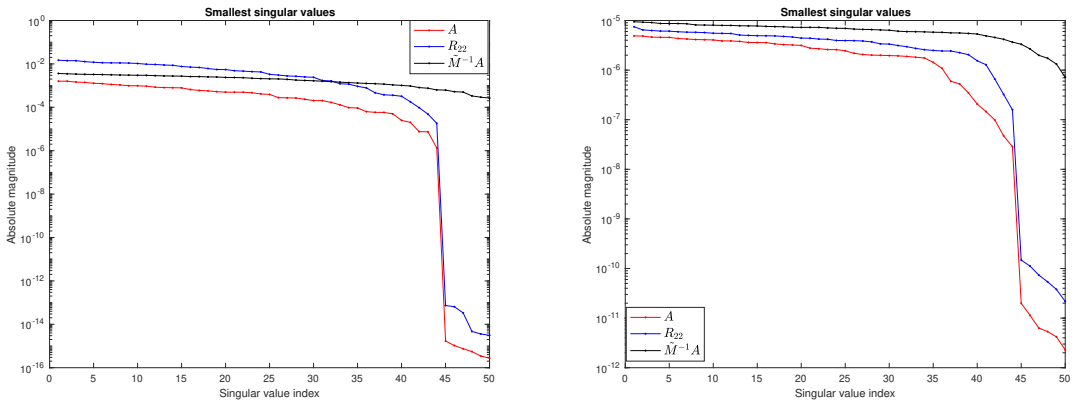

Figure 4.5: 50 smallest singular values of $\boldsymbol{A}$, R_{22} and $\tilde{M}^{-1}\boldsymbol{A}$. Results for the matrix crystk03 (left) and bcsstk35 (right) in MATLAB 2017.

Figure 4.6 shows the numerical results for the matrix sherman3 arising from the oil resevoir modeling and the matrix msc01440 from the structural problem. We can see that the smallest singular values of $\boldsymbol{A}$ are well approximated by the smallest ones of R_{22} and they are also replaced by larger ones when the deflation preconditioner $\tilde{M}$ is applied.

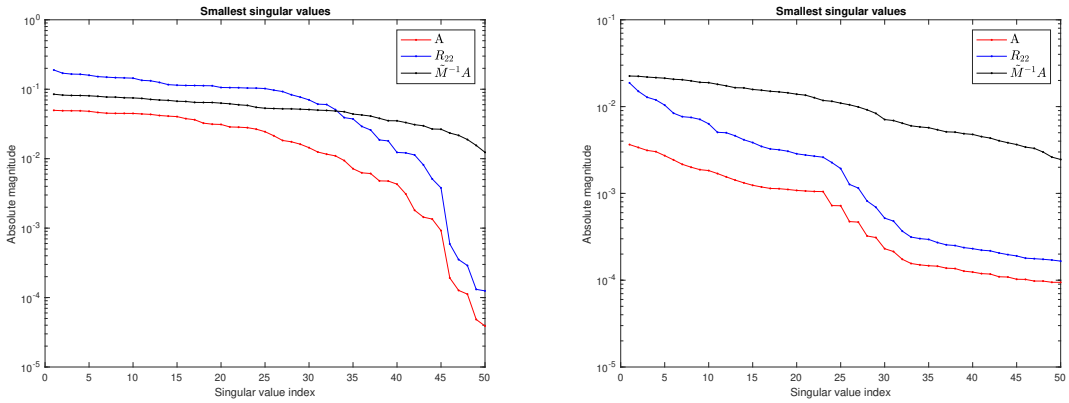

Figure 4.6: 50 smallest singular values of $\boldsymbol{A}$, R_{22} and $\tilde{M}^{-1}\boldsymbol{A}$. Results for the matrix sherman3 (left) and msc01440 (right) in MATLAB 2017.

In Figure 4.7 we show the numerical results for the matrix e20r5000 and e40r5000 from the 2D fluid flow in a driven cavity with the Reynold number $Re = 5000$. As the previous test cases, the smallest singular values of R_{22} can be considered as good approximations for the smallest ones of $\boldsymbol{A}$. They also become larger under the impact of the deflation preconditioner $\tilde{M}$.

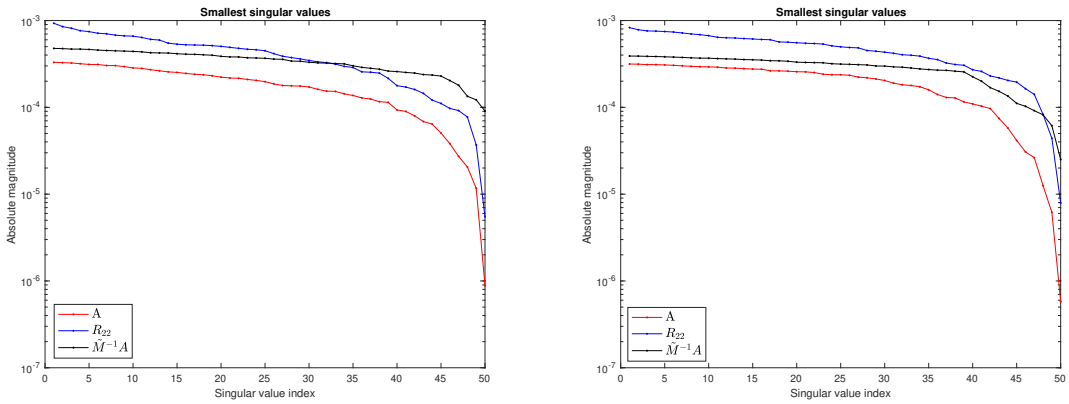

Figure 4.7: 50 smallest singular values of $\boldsymbol{A}$, R_{22} and $\tilde{M}^{-1}\boldsymbol{A}$. Results for the matrix e20r5000 (left) and e40r5000 (right) in MATLAB 2017.

4.8.3 Comparison between deflation, block Jacobi and mixed preconditioners when used with GMRES

In this section we give the comparison results between deflation preconditioner, block Jacobi preconditioner and their combination when used with GMRES. To be more specific, we consider the following linear system and preconditioned systems,

$$\begin{aligned} \boldsymbol{A}x &= b, && (4.57)\\ \tilde{M}^{-1}\boldsymbol{A}x &= \tilde{M}^{-1}b, && (4.58)\\ M_{BJ}^{-1}\boldsymbol{A}x &= M_{BJ}^{-1}b, && (4.59)\\ \tilde{M}^{-1}M_{BJ}^{-1}\boldsymbol{A}x &= \tilde{M}^{-1}M_{BJ}^{-1}b, && (4.60) \end{aligned}$$

where $\boldsymbol{A}$ is the test matrix and the right hand side b is a random orthonormal vector. We remark that the linear system (4.57) corresponds to the non-preconditioned case, the preconditioned system (4.58) corresponds to the deflation preconditioner case, (4.59) corresponds to the block Jacobi preconditioner case, and (4.60) corresponds to the mixed preconditioner case. We also remark that the number of diagonal blocks in the block Jacobi preconditioner is set to 8 for all test cases. With $tol = 10^{-6}$, the maximum number of iteration $maxit = length(b)$, we apply GMRES for the 4 systems (4.57) and the results are given in Table 4.2.

We observe from Table 4.2 that the mixed preconditioner always results in a much smaller number of iterations compared to non-preconditioned GMRES. Specifically for crystk03, it converges after 6 iterations to the error of 10^{-6} while non-preconditioned GMRES does not converge after 5266 iterations, deflation and block Jacobi preconditioners seem to be stagnated. For bcsstk35, the mixed preconditioned converges to the error of 10^{-5} after only 48 iterations while for the non-preconditioned GMRES it takes 29333 iterations to converge to the error of 10^{-3}. For e40r5000, it takes 128 iterations for the mixed preconditioner compared to 13793 for the non-preconditioned GMRES to converge to the error of 10^{-7}.

GMRES with deflation preconditioner converges faster than the non-preconditioned GMRES in all cases. However for the test matrices from 2D fluid flow in a driven cavity the difference is small. For example for e20r5000, it takes 3430 iterations compared to 4047 iterations of the non-preconditioned GMRES. For e30r5000, it takes 7449 iterations compared to 8139 iterations and for e40r5000, it takes 12491 compared to 13793 of the non-preconditioned GMRES. The difference becomes significant in the cases of mixed preconditioner GMRES, with the help of block Jacobi preconditioner.

4.9 Conclusion and perspectives

In this work we study a new deflation technique based on the sparse QR with tournament pivoting and nested dissection, which allows to reduce the computational costs as well as exploit parallel computing. The smallest singular values of $\boldsymbol{A}$ are shown to be well approximated by the smallest singular values of R_{22} in the QR factorization following our strategy. The deflation technique based on QR factorization is not as good as the deflation

technique based on SVD in terms of accuracy. However it significantly reduces the computational costs and allows us to effectively exploit parallelism. Especially the combination between the deflation technique based on QR with the block Jacobi preconditioner results in a very small number of iterations compare to the non-preconditioned case for GMRES for our test cases. Parallel computing experiments of the method and the combination with block Jacobi preconditioner as well as the computational costs will be studied in the future work.

Test matrix	Type of preconditioner	Relative residual	Number of iteration
saylr4 3564×3564	Non-preconditioned	9.8817e-07	1600
	Deflation preconditioner	9.5106e-07	132
	Block Jacobi preconditioner	8.5713e-07	312
	Mixed preconditioner	8.7417e-07	51
sherman3 5005×5005	Non-preconditioned	8.1486e-07	433
	Deflation preconditioner	8.3483e-07	74
	Block Jacobi preconditioner	9.2290e-07	42
	Mixed preconditioner	8.7324e-07	18
lshp3466 3466×3466	Non-preconditioned	9.9674e-07	3095
	Deflation preconditioner	9.9289e-07	1376
	Block Jacobi preconditioner	7.0979e-07	293
	Mixed preconditioner	9.9508e-07	228
lshp3025 3025×3025	Non-preconditioned	6.7470e-07	2700
	Deflation preconditioner	9.8547e-07	1199
	Block Jacobi preconditioner	9.6807e-07	307
	Mixed preconditioner	6.2170e-07	245
crystk03 24696×24696	Non-preconditioned	5.5625e+00	5266
	Deflation preconditioner	2.1876e-01	286
	Block Jacobi preconditioner	1.0644e-01	509
	Mixed preconditioner	9.3349e-06	6
msc01440 1440×1440	Non-preconditioned	9.6790e-07	809
	Deflation preconditioner	9.9560e-07	168
	Block Jacobi preconditioner	7.7009e-07	189
	Mixed preconditioner	9.4425e-07	58
e20r5000 4241×4241	Non-preconditioned	9.8670e-07	4047
	Deflation preconditioner	9.7713e-07	3430
	Block Jacobi preconditioner	9.9383e-07	168
	Mixed preconditioner	9.7225e-07	171
e30r5000 9661×9661	Non-preconditioned	9.6887e-07	8139
	Deflation preconditioner	9.9700e-07	7449
	Block Jacobi preconditioner	7.9167e-07	160
	Mixed preconditioner	9.2968e-07	143
e40r5000 17281×17281	Non-preconditioned	9.5896e-07	13793
	Deflation preconditioner	9.8871e-07	12491
	Block Jacobi preconditioner	9.5714e-07	154
	Mixed preconditioner	8.5884e-07	128
bcsstk35 30237×30237	Non-preconditioned	1.3054e-03	29333
	Deflation preconditioner	3.4782e-05	77
	Block Jacobi preconditioner	2.0646e-05	1662
	Mixed preconditioner	5.9531e-05	48

Table 4.2: GMRES convergence comparison.

Conclusion and Perspectives

In this thesis, we presented a new preconditioner so-called SC two-level additive Schwarz in time preconditioner. Connection between this preconditioner, MGRIT with F-relaxation and parareal was made. By studying SC two-level additive Schwarz preconditioner and its variants, we proposed a variant so-called SCS^2 that can effectively exploit parallel computing for the fine propagator. Numerical experiments and convergence analysis were given for SC two-level additive Schwarz preconditioner and its variants. Furthermore, one can accelerate the parareal algorithm via SC two-level additive Schwarz in time preconditioner by using some Krylov methods, namely GMRES. We found that for the advection-reaction-diffusion equation, in the case where the advection and reaction terms are large compared to the diffusion term, parareal with GMRES acceleration shows its advantage while the plain parareal stagnates. The convergence analysis of parareal with GMRES acceleration will be studied further in our future work. Also in this work we applied parareal for solving oscillatory singularly perturbed ODEs which are characteristics of a six-dimensional Vlasov equation. By using a limit model based on the two-scale asymptotic expansion for the coarse solver, we obtained rapid convergence in simulations of charged particles in a Penning trap, isotope separation by ion cyclotron resonance or in a strong variable magnetic field. Convergence results obtained from various numerical experiments show that the coupling strategy is efficient with a uniform rate. The same coupling strategy for solving the Vlasov-Poisson equation together with the convergence analysis are left for future work. Finally in this work, we derived a new deflation preconditioner for the smallest singular values of the coefficient matrix $\boldsymbol{A}$, based on sparse QR, especially strong RRQR factorization with tournament pivoting strategy and nested dissection partition on $\boldsymbol{A}$. This coupling strategy approximates well the smallest singular values of $\boldsymbol{A}$ and replaces them with larger ones. Furthermore, it also allows us to take advantages of parallel computing thank to the tournament pivoting strategy. Combination between the deflation preconditioner and block Jacobi preconditioner gives promising results for GMRES's convergence. In the future work, we aim at parallelizing the method as well as the combination with block Jacobi preconditioner.

Bibliography

[1] H. Al Daas, L. Grigori, P. Hénon, and P. Ricoux. *Recycling Krylov subspaces and reducing deflation subspaces for solving sequence of linear systems*. Research Report RR-9206. Inria Paris, Oct. 2018. URL: https://hal.inria.fr/hal-01886546.

[2] H. Anzt, J. Dongarra, G. Flegar, N. Higham, and E. S. Quintana-Orti. "Adaptive precision in block-Jacobi preconditioning for iterative sparse linear system solvers: Adaptive precision in block-Jacobi preconditioning for iterative solvers". In: *Concurrency and Computation: Practice and Experience* 31 (Mar. 2018), e4460. DOI: 10.1002/cpe.4460.

[3] G. Ariel, S. J. Kim, and R. Tsai. "Parareal multiscale methods for highly oscillatory dynamical systems". In: *SIAM Journal on Scientific Computing* 38.6 (2016), A3540–A3564.

[4] E.H. Ayachour. "A fast implementation for GMRES method". In: *Journal of Computational and Applied Mathematics* 159.2 (2003), pages 269–283. ISSN: 0377-0427. DOI: https://doi.org/10.1016/S0377-0427(03)00534-X. URL: https://www.sciencedirect.com/science/article/pii/S037704270300534X.

[5] L. Baffico, S. Bernard, Y. Maday, G. Turinici, and G. Zérah. "Parallel-in-time molecular-dynamics simulations". In: *Phys. Rev. E* 66 (5 2002), page 057701. DOI: 10.1103/PhysRevE.66.057701. URL: http://link.aps.org/doi/10.1103/PhysRevE.66.057701.

[6] G. Bal. "On the Convergence and the Stability of the Parareal Algorithm to Solve Partial Differential Equations". In: *Domain Decomposition Methods in Science and Engineering*. Edited by T. J. Barth et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pages 425–432. ISBN: 978-3-540-26825-3.

[7] G. Bal and Y. Maday. "A "Parareal" time discretization for non-linear PDE's with application to the pricing of an American Put". In: *Recent Developments in Domain Decomposition Methods*. Edited by L. Pavarino and A. Toselli. Volume 23. Lecture Notes in Computational Science and Engineering. Springer Berlin, 2002, pages 189–202. DOI: 10.1007/978-3-642-56118-4_12. URL: http://dx.doi.org/10.1007/978-3-642-56118-4_12.

[8] A. M. Baudron, J. J. Lautard, Y. Maday, and O. Mula. "The parareal in time algorithm applied to the kinetic neutron diffusion equation". In: *Domain Decomposition Methods in Science and Engineering XXI*. Lecture Notes in Computational Science and Engineering. Springer International Publishing, 2014, pages 437–445. DOI: 10.1007/978-3-319-05789-7_41. URL: http://dx.doi.org/10.1007/978-3-319-05789-7_41.

[9] A. M. Baudron, J. J. Lautard, Y. Maday, M. K. Riahi, and J. Salomon. "Parareal in time 3D numerical solver for the LWR Benchmark neutron diffusion transient model". In: *Journal of Computational Physics* 279.0 (2014), pages 67–79. DOI: 10.1016/j.jcp.2014.08.037. URL: http://dx.doi.org/10.1016/j.jcp.2014.08.037.

[10] M. Beaupère and L. Grigori. "Communication avoiding low rank approximation based on QR with tournament pivoting". working paper or preprint. Jan. 2021. URL: https://hal.inria.fr/hal-02947991.

[11] C.H. Bischof and G.M. Shroff. "On updating signal subspaces". In: *IEEE Transactions on Signal Processing* 40.1 (1992), pages 96–105. DOI: 10.1109/78.157185.

[12] W. Briggs, V. Henson, and S. McCormick. *A Multigrid Tutorial, 2nd Edition*. Jan. 2000. ISBN: 978-0-89871-462-3.

[13] P. Businger and G. H. Golub. "Linear least squares solutions by householder transformations". In: *Numerische Mathematik* 7.3 (1965), pages 269–276. DOI: 10.1007/BF01436084. URL: https://doi.org/10.1007/BF01436084.

[14] T. F. Chan and P. C. Hansen. "Computing Truncated Singular Value Decomposition Least Squares Solutions by Rank Revealing QR-Factorizations". In: *SIAM Journal on Scientific and Statistical Computing* 11.3 (1990), pages 519–530. DOI: 10.1137/0911029. eprint: https://doi.org/10.1137/0911029. URL: https://doi.org/10.1137/0911029.

[15] T. F. Chan and P. C. Hansen. "Some Applications of the Rank Revealing QR Factorization". In: *SIAM Journal on Scientific and Statistical Computing* 13.3 (1992), pages 727–741. DOI: 10.1137/0913043. eprint: https://doi.org/10.1137/0913043. URL: https://doi.org/10.1137/0913043.

[16] T. F. Chan and T. P. Mathew. "Domain decomposition algorithms". In: *Acta Numerica* 3 (1994), pages 61–143. DOI: 10.1017/S0962492900002427.

[17] X. Dai, C. Le Bris, F. Legoll, and Y. Maday. "Symmetric parareal algorithms for Hamiltonian systems". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 47 (03 Apr. 2013), pages 717–742. DOI: 10.1051/m2an/2012046. URL: http://dx.doi.org/10.1051/m2an/2012046.

[18] E. de Sturler. "Nested Krylov methods based on GCR". In: *Journal of Computational and Applied Mathematics* 67.1 (1996), pages 15–41. ISSN: 0377-0427. DOI: https://doi.org/10.1016/0377-0427(94)00123-5. URL: http://www.sciencedirect.com/science/article/pii/0377042794001235.

[19] J. Demmel, L. Grigori, M. Gu, and H. Xiang. "Communication Avoiding Rank Revealing QR Factorization with Column Pivoting". In: *SIAM Journal on Matrix Analysis and Applications* 36 (Jan. 2015), pages 55–89. DOI: 10.1137/13092157X.

[20] V. Dobrev, T. Kolev, N. Petersson, and J. Schroder. "Two-Level Convergence Theory for Parallel Time Integration with Multigrid". In: *SIAM Journal on Scientific Computing* 39 (Oct. 2017), S501–S527. DOI: 10.1137/16M1074096.

[21] V. Dolean, P. Jolivet, and F. Nataf. *An Introduction to Domain Decomposition Methods*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2015. DOI: 10.1137/1.9781611974065. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611974065. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611974065.

[22] N. Duan, S. Simunovic, A. Dimitrovski, and K. Sun. "Improving the Convergence Rate of Parareal-in-time Power System Simulation using the Krylov Subspace". In: *2018 IEEE Power Energy Society General Meeting (PESGM)*. 2018, pages 1–5. DOI: 10.1109/PESGM.2018.8586354. URL: https://dx.doi.org/10.1109/PESGM.2018.8586354.

[23] A. Eghbal, A. G. Gerber, and E. Aubanel. "Acceleration of Unsteady Hydrodynamic Simulations Using the Parareal Algorithm". In: *Journal of Computational Science* 19 (2016), pages 57–76. DOI: 10.1016/j.jocs.2016.12.006. URL: http://dx.doi.org/10.1016/j.jocs.2016.12.006.

[24] J. Erhel, K. Burrage, and B. Pohl. "Restarted GMRES preconditioned by deflation". In: *Journal of Computational and Applied Mathematics* 69.2 (1996), pages 303–318. ISSN: 0377-0427. DOI: https://doi.org/10.1016/0377-0427(95)00047-X. URL: http://www.sciencedirect.com/science/article/pii/037704279500047X.

[25] R. D. Falgout, S. Friedhoff, T. V. Kolev, S. P. MacLachlan, and J. B. Schroder. "Parallel time integration with multigrid". In: *SIAM Journal on Scientific Computing* 36 (6 2014), pages C635–C661. DOI: 10.1137/130944230. URL: http://dx.doi.org/10.1137/130944230.

[26] C. Farhat and M. Chandesris. "Time-decomposed parallel time-integrators: theory and feasibility studies for fluid, structure, and fluid-structure applications". In: *International Journal for Numerical Methods in Engineering* 58.9 (2003), pages 1397–1434. DOI: 10.1002/nme.860. URL: http://dx.doi.org/10.1002/nme.860.

[27] P. F. Fischer, F. Hecht, and Y. Maday. "A parareal in time semi-implicit approximation of the Navier-Stokes equations". In: *Domain decomposition methods in science and engineering*. Springer, 2005, pages 433–440.

[28] E. Frénod. "Application of the averaging method to the gyrokinetic plasma". In: *Asymptotic Analysis* 46 (2006), pages 1–28.

[29] E. Frénod and F. Watbled. "The Vlasov equation with strong magnetic field and oscillating electric field as a model for isotop resonant separation". In: *Electron. J. Diff. Eqns.* 2002.6 (2002), pages 1–20.

[30] S. Friedhoff, R. D. Falgout, T. V. Kolev, S. P. MacLachlan, and J. B. Schroder. "A Multigrid-in-Time Algorithm for Solving Evolution Equations in Parallel". In: *Presented at: Sixteenth Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO, United States, Mar 17 - Mar 22, 2013*. 2013. URL: http://www.osti.gov/scitech/servlets/purl/1073108.

[31] M. Gander. "Time Parallel Time Integration". 2018. URL: https://www.unige.ch/~gander/poly.pdf.

[32] M. Gander, Y. Jiang, and B. Song. "A Superlinear Convergence Estimate for the Parareal Schwarz Waveform Relaxation Algorithm". In: *SIAM Journal on Scientific Computing* 41.2 (2019), A1148–A1169. DOI: 10.1137/18M1177226. URL: https://doi.org/10.1137/18M1177226.

[33] M. J. Gander and E. Hairer. "Nonlinear Convergence Analysis for the Parareal Algorithm". In: *Domain Decomposition Methods in Science and Engineering XVII*. Edited by U. Langer, M. Discacciati, D. E. Keyes, O. B. Widlund, and W. Zulehner. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pages 45–56. ISBN: 978-3-540-75199-1.

[34] M. J. Gander and E. Hairer. "Analysis for parareal algorithms applied to Hamiltonian differential equations". In: *Journal of Computational and Applied Mathematics* 259, Part A.0 (2014). Proceedings of the Sixteenth International Congress on Computational and Applied Mathematics (ICCAM-2012), Ghent, Belgium, 9-13 July, 2012, pages 2–13. DOI: 10.1016/j.cam.2013.01.011. URL: http://dx.doi.org/10.1016/j.cam.2013.01.011.

[35] M. J. Gander, Y. L. Jiang, and R. J. Li. "Parareal Schwarz Waveform Relaxation Methods". In: *Domain Decomposition Methods in Science and Engineering XX*. Edited by R. Bank, M. Holst, O. Widlund, and J. Xu. Volume 91. Lecture Notes in Computational Science and Engineering. Springer Berlin Heidelberg, 2013, pages 451–458. DOI: 10.1007/978-3-642-35275-1_53. URL: http://dx.doi.org/10.1007/978-3-642-35275-1_53.

[36] M. J. Gander, F. Kwok, and H. Zhang. "Multigrid interpretations of the parareal algorithm leading to an overlapping variant and MGRIT". In: *Computing and Visualization in Science* (2018). DOI: 10.1007/s00791-018-0297-y. URL: https://doi.org/10.1007/s00791-018-0297-y.

[37] M. J. Gander and S. Vandewalle. "Analysis of the Parareal Time-Parallel Time-Integration Method". In: *SIAM Journal on Scientific Computing* 29.2 (2007), pages 556–578. DOI: 10.1137/05064607X. URL: http://dx.doi.org/10.1137/05064607X.

[38] Gander, M. and Petcu, M. "Analysis of a Krylov subspace enhanced parareal algorithm for linear problems". In: *ESAIM: Proc.* 25 (2008), pages 114–129. DOI: 10.1051/proc:082508. URL: https://doi.org/10.1051/proc:082508.

[39] J. Geiser and S. Güttel. "Coupling methods for heat transfer and heat flow: Operator splitting and the parareal algorithm". In: *Journal of Mathematical Analysis and Applications* 388.2 (2012), pages 873–887. DOI: 10.1016/j.jmaa.2011.10.030. URL: http://dx.doi.org/10.1016/j.jmaa.2011.10.030.

[40] A. George. "Nested Dissection of a Regular Finite Element Mesh". In: *SIAM Journal on Numerical Analysis* 10.2 (1973), pages 345–363. DOI: 10.1137/0710032. eprint: https://doi.org/10.1137/0710032. URL: https://doi.org/10.1137/0710032.

[41] J. R. Gilbert and R. E. Tarjan. "The analysis of a nested dissection algorithm". In: *Numerische Mathematik* 50.4 (1986), pages 377–404. DOI: 10.1007/BF01396660. URL: https://doi.org/10.1007/BF01396660.

[42] G. Golub. "Numerical methods for solving linear least squares problems". In: *Numerische Mathematik* 7.3 (1965), pages 206–216. DOI: 10.1007/BF01436075. URL: https://doi.org/10.1007/BF01436075.

[43] G. Golub, V. Klema, and G. W. Stewart. *Rank Degeneracy and Least Squares Problems*. Technical report. 1976.

[44] G. H. Golub and C. F. Van Loan. *Matrix Computations*. 2nd ed. The Johns Hopkins University Press, Baltimore, MD, 1989.

[45] G. H. Golub and V. Pereyra. "The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Variables Separate". In: *SIAM Journal on Numerical Analysis* 10.2 (1973), pages 413–432. DOI: 10.1137/0710036. eprint: https://doi.org/10.1137/0710036. URL: https://doi.org/10.1137/0710036.

[46] L. Grigori, J. W. Demmel, and H. Xiang. "CALU: A Communication Optimal LU Factorization Algorithm". In: *SIAM Journal on Matrix Analysis and Applications* 32.4 (2011), pages 1317–1350. DOI: 10.1137/100788926.

[47] L. Grigori and S. Moufawad. "Communication Avoiding ILU0 Preconditioner". In: *SIAM Journal on Scientific Computing* 37.2 (2015), pages C217–C246. DOI: 10.1137/130930376. eprint: https://doi.org/10.1137/130930376. URL: https://doi.org/10.1137/130930376.

[48] M. Gu and S. C. Eisenstat. "Efficient Algorithms for Computing a Strong Rank-Revealing QR Factorization". In: *SIAM J. Sci. Comput.* 17.4 (July 1996), pages 848–869. ISSN: 1064-8275. DOI: 10.1137/0917055. URL: https://doi.org/10.1137/0917055.

[49] R. Guetat. "Coupling Parareal with Non-Overlapping Domain Decomposition Method". hal-01312528. 2016. URL: https://hal.archives-ouvertes.fr/hal-01312528/.

[50] G. Gurrala, A. Dimitrovski, S. Pannala, S. Simunovic, and M. Starke. "Parareal in Time for Fast Power System Dynamic Simulations". In: *IEEE Transactions on Power Systems* 31.3 (2016), pages 1820–1830. DOI: 10.1109/TPWRS.2015.2434833. URL: http://dx.doi.org/10.1109/TPWRS.2015.2434833.

[51] G. Gurrala, A. Dimitrovski, P. Sreekanth, S. Simunovic, and M. Starke. "Parareal in Time for Dynamic Simulations of Power Systems". In: *Proceedings of the International Conference on Power Systems Transients (IPST2015) in Cavtat, Croatia June 15-18, 2015*. 2015. URL: http://www.ipstconf.org/papers/Proc_IPST2015/15IPST073.pdf.

[52] W. Hackbusch. *Multi-Grid Methods and Applications (2nd printing)*. Jan. 2003. ISBN: 3-540-12761-5.

[53] T. Haut and B. Wingate. "An asymptotic parallel-in-time method for highly oscillatory PDEs". In: *SIAM Journal on Scientific Computing* 36.2 (2014), A693–A713.

[54] R. D. Hazeltine and J. D. Meiss. *Plasma confinement*. Courier Corporation, 2003.

[55] L. P. He and M. He. "Parareal in Time Simulation Of Morphological Transformation in Cubic Alloys with Spatially Dependent Composition". In: *Communications in Computational Physics* 11 (5 2012), pages 1697–1717. DOI: 10.4208/cicp.110310.090911a. URL: http://dx.doi.org/10.4208/cicp.110310.090911a.

[56] Y. He, Y. Sun, J. Liu, and H. Qin. "Volume-preserving algorithms for charged particle dynamics". In: *Journal of Computational Physics* 281 (2015), pages 135–147.

[57] M. Hegland and P. Saylor. "Block Jacobi Preconditioning of the Conjugate Gradient Method on a Vector Processor". In: *International Journal of Computer Mathematics* 44 (Jan. 1992), pages 71–89. DOI: 10.1080/00207169208804096.

[58] A. Hessenthaler, D. Nordsletten, O. Röhrle, J. B. Schroder, and R. D. Falgout. "Convergence of the multigrid reduction in time algorithm for the linear elasticity equations". In: *Numerical Linear Algebra with Applications* 25.3 (2018), e2155. DOI: 10.1002/nla.2155.

[59] C. Hirsch. *Numerical computation of internal and external flows: The fundamentals of computational fluid dynamics*. Elsevier, 2007.

[60] In: *Domain Decomposition Methods in Science and Engineering XXI*. Edited by Erhel J., Gander M. J., Halpern L., Pichot G., Sassi T., and Widlund O. Springer, Cham, 2014. ISBN: 978-3-319-05789-7.

[61] S. M. Kaber and Y. Maday. "Parareal in time approximation of the Korteveg-deVries-Burgers' equations". In: *PAMM* 7 (1 2007), pages 1026403–1026404. DOI: {10.1002/pamm.200700574}. URL: %7Bhttp://dx.doi.org/10.1002/pamm.200700574%7D.

[62] V. E. Kane, R. C. Ward, and G. J. Davis. "Assessment of Linear Dependencies in Multivariate Data". In: *SIAM Journal on Scientific and Statistical Computing* 6.4 (1985), pages 1022–1032. DOI: 10.1137/0906070. eprint: https://doi.org/10.1137/0906070. URL: https://doi.org/10.1137/0906070.

[63] G. Karypis and V. Kumar. "Multilevel k-way Partitioning Scheme for Irregular Graphs". In: *Journal of Parallel and Distributed Computing* 48.1 (1998), pages 96–129. ISSN: 0743-7315. DOI: https://doi.org/10.1006/jpdc.1997.1404. URL: https://www.sciencedirect.com/science/article/pii/S0743731597914040.

[64] A. Kreienbuehl, A. Naegel, D. Ruprecht, R. Speck, G. Wittum, and R. Krause. "Numerical simulation of skin transport using Parareal". In: *Computing and Visualization in Science* 17 (2 2015), pages 99–108. DOI: 10.1007/s00791-015-0246-y. URL: http://dx.doi.org/10.1007/s00791-015-0246-y.

[65] M. Kretzschmar. "Particle motion in a Penning trap". In: *European Journal of Physics* 12.5 (1991), page 240.

[66] C. L. Lawson and Richard J. Hanson. *Solving Least Squares Problems*. Society for Industrial and Applied Mathematics, 1995. DOI: 10.1137/1.9781611971217. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611971217. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611971217.

[67] F. Legoll, T. Lelievre, and G. Samaey. “A micro-macro parareal algorithm: application to singularly perturbed ordinary differential equations”. In: *SIAM Journal on Scientific Computing* 35.4 (2013), A1951–A1986.

[68] J.-L. Lions, Y. Maday, and G. Turinici. “A "parareal" in time discretization of PDE’s”. In: *Comptes Rendus de l’Académie des Sciences - Series I - Mathematics* 332 (2001), pages 661–668. DOI: 10.1016/S0764-4442(00)01793-6. URL: http://dx.doi.org/10.1016/S0764-4442(00)01793-6.

[69] R. J. Lipton, D. J. Rose, and R. E. Tarjan. “Generalized Nested Dissection”. In: *SIAM Journal on Numerical Analysis* 16.2 (1979), pages 346–358. DOI: 10.1137/0716027. eprint: https://doi.org/10.1137/0716027. URL: https://doi.org/10.1137/0716027.

[70] T. Loderer, V. Heuveline, and R. Lohner. “The parareal algorithm as a new approach for numerical integration of ODEs in real-time simulations in automotive industry”. In: *PAMM* 14.1 (2014), pages 1027–1030. DOI: 10.1002/pamm.201410489. URL: http://dx.doi.org/10.1002/pamm.201410489.

[71] Y. Maday. “Parareal in time algorithm for kinetic systems based on model reduction”. In: *High-dimensional partial differential equations in science and engineering, CRM Proc. Lecture Notes* 41 (2007), pages 183–194.

[72] F. Magoulès, G. Gbikpi-Benissan, and Q. Zou. “Asynchronous Iterations of Parareal Algorithm for Option Pricing Models”. In: *Mathematics* 6.4 (2018). DOI: 10.3390/math6040045. URL: https://doi.org/10.3390/math6040045.

[73] M. Matinfar, H. Zareamoghaddam, M. Eslami, and M. Saeidy. “GMRES implementations and residual smoothing techniques for solving ill-posed linear systems”. In: *Computers and Mathematics with Applications* 63.1 (2012), pages 1–13. ISSN: 0898-1221. DOI: https://doi.org/10.1016/j.camwa.2011.09.022. URL: https://www.sciencedirect.com/science/article/pii/S0898122111007905.

[74] M. Minion. “A Hybrid Parareal Spectral Deferred Corrections Method”. In: *Communications in Applied Mathematics and Computational Science* 5 (Dec. 2010). DOI: 10.2140/camcos.2010.5.265.

[75] M. L. Minion. “A Hybrid Parareal Spectral Deferred Corrections Method”. In: *Communications in Applied Mathematics and Computational Science* 5.2 (2010), pages 265–301. DOI: 10.2140/camcos.2010.5.265. URL: http://dx.doi.org/10.2140/camcos.2010.5.265.

[76] M. L. Minion and S. A. Williams. “Parareal and spectral deferred corrections”. In: *AIP Conference Proceedings*. Volume 1048. 2008, page 388. DOI: 10.1063/1.2990941. URL: http://dx.doi.org/10.1063/1.2990941.

[77] R.C Mittal and A.H Al-Kurdi. “An efficient method for constructing an ILU preconditioner for solving large sparse nonsymmetric linear systems by the GMRES method”. In: *Computers and Mathematics with Applications* 45.10 (2003), pages 1757–1772. ISSN: 0898-1221. DOI: https://doi.org/10.1016/S0898-1221(03)00154-8. URL: https://www.sciencedirect.com/science/article/pii/S0898122103001548.

[78] R. Morgan. "GMRES with deflation restarting". In: *Siam Journal on Scientific Computing* 24 (Aug. 2002). DOI: 10.1137/S1064827599364659.

[79] O. Nevanlinna. "Linear acceleration of Picard-Lindelöf iteration". In: *Numerische Mathematik* 57 (Dec. 1990), pages 147–156. DOI: 10.1007/BF01386404.

[80] Bruce W. P. and James P. H. "Application of preconditioned GMRES to the numerical solution of the neutron transport equation". In: *Annals of Nuclear Energy* 29.2 (2002), pages 109–136. ISSN: 0306-4549. DOI: https://doi.org/10.1016/S0306-4549(01)00034-2. URL: https://www.sciencedirect.com/science/article/pii/S0306454901000342.

[81] G. Pagès, O. Pironneau, and G. Sall. "The Parareal Algorithm for American Options". In: *SIAM Journal on Financial Mathematics* 9.3 (2018), pages 966–993. DOI: 10.1137/17M1138832. URL: https://doi.org/10.1137/17M1138832.

[82] M. L. Parks, E. de Sturler, G. Mackey, D. D. Johnson, and S. Maiti. "Recycling Krylov Subspaces for Sequences of Linear Systems". In: *SIAM Journal on Scientific Computing* 28.5 (2006), pages 1651–1674. DOI: 10.1137/040607277. eprint: https://doi.org/10.1137/040607277. URL: https://doi.org/10.1137/040607277.

[83] A. Quarteroni and A. Valli. "Domain Decomposition Methods for Partial Differential Equations". In: (Jan. 1999).

[84] In: *Domain Decomposition Methods in Science and Engineering XX*. Edited by E. Bank R., Holst M., Widlund O. B., and Xu J. Springer-Verlag Berlin Heidelberg, 2013. ISBN: 978-3-642-35275-1.

[85] J. M. Reynolds-Barredo, D. E. Newman, R. S. Sánchez, and L. A. Berry. "Modelling parareal convergence in 2D drift wave plasma turbulence". In: *High Performance Computing and Simulation (HPCS), 2012 International Conference on*. 2012, pages 726–727. DOI: 10.1109/HPCSim.2012.6267004. URL: http://dx.doi.org/10.1109/HPCSim.2012.6267004.

[86] D. Ruprecht. "Convergence of Parareal with spatial coarsening". In: *PAMM* 14.1 (2014), pages 1031–1034. DOI: 10.1002/pamm.201410490. URL: http://dx.doi.org/10.1002/pamm.201410490.

[87] D. Ruprecht. "Wave propagation characteristics of Parareal". In: *Computing and Visualization in Science* 19.1 (2018), pages 1–17. DOI: 10.1007/s00791-018-0296-z. URL: https://doi.org/10.1007/s00791-018-0296-z.

[88] Ulrich T.; Cornelius W. O.; Anton S. *Multigrid*. 2000. ISBN: 9780080479569.

[89] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Second. Society for Industrial and Applied Mathematics, 2003. DOI: 10.1137/1.9780898718003. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9780898718003. URL: https://epubs.siam.org/doi/abs/10.1137/1.9780898718003.

[90] Y. Saad and M. H. Schultz. "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems". In: *SIAM Journal on Scientific and Statistical Computing* 7.3 (1986), pages 856–869. DOI: 10.1137/0907058.

[91] Youcef Saad and Martin H. Schultz. "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems". In: *SIAM Journal on Scientific and Statistical Computing* 7.3 (1986), pages 856–869. DOI: 10.1137/0907058. eprint: https://doi.org/10.1137/0907058. URL: https://doi.org/10.1137/0907058.

[92] D. Samaddar, D. P. Coster, X. Bonnin, C. Bergmeister, E. Havlícková, L. A. Berry, W. R. Elwasif, and D. B. Batchelor. "Poster: Greater than 10x Acceleration of fusion plasma edge simulations using the Parareal algorithm". In: *Proceedings of the 2014 Conference on High Performance Computing Networking, Storage and Analysis Companion*. SC '14 Companion. New Orleans, Louisiana, USA, 2014. URL: http://sc14.supercomputing.org/sites/all/themes/sc14/files/archive/tech_poster/poster_files/post163s2-file3.pdf.

[93] D. Samaddar, D. E. Newman, and R. S. Sánchez. "Parallelization in time of numerical simulations of fully-developed plasma turbulence using the parareal algorithm". In: *Journal of Computational Physics* 229 (18 2010), pages 6558–6573. DOI: 10.1016/j.jcp.2010.05.012. URL: http://dx.doi.org/10.1016/j.jcp.2010.05.012.

[94] T. Sekine, T. Tsuji, T. Oyama, F. Magoulès, and K. Uchida. "Speedup of parallel computing by parareal method in transient stability analysis of Japanese power system". In: *2016 IEEE Innovative Smart Grid Technologies - Asia (ISGT-Asia)*. 2016, pages 1177–1182. DOI: 10.1109/ISGT-Asia.2016.7796552. URL: http://dx.doi.org/10.1109/ISGT-Asia.2016.7796552.

[95] T. M. Shah. In: *Analysis of the multigrid method*. Ph.D. Thesis Oxford Univ. (England), 1989.

[96] V. Simoncini. "On the Convergence of Restarted Krylov Subspace Methods". In: *SIAM Journal on Matrix Analysis and Applications* 22.2 (2000), pages 430–452. DOI: 10.1137/S0895479898348507. eprint: https://doi.org/10.1137/S0895479898348507. URL: https://doi.org/10.1137/S0895479898348507.

[97] G. A. Staff and E. M. Rønquist. "Stability of the Parareal Algorithm". In: *Domain Decomposition Methods in Science and Engineering*. Edited by T. J. Barth et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pages 449–456. ISBN: 978-3-540-26825-3.

[98] J. Steiner, D. Ruprecht, R. Speck, and R. Krause. "Convergence of Parareal for the Navier-Stokes equations depending on the Reynolds number". In: *Numerical Mathematics and Advanced Applications - ENUMATH 2013*. Edited by Assyr Abdulle, Simone Deparis, Daniel Kressner, Fabio Nobile, and Marco Picasso. Volume 103. Lecture Notes in Computational Science and Engineering. Springer International Publishing, 2015, pages 195–202. DOI: 10.1007/978-3-319-10705-9_19. URL: http://dx.doi.org/10.1007/978-3-319-10705-9_19.

[99] J. M. Tang, R. Nabben, C. Vuik, and Y. A. Erlangga. "Comparison of Two-Level Preconditioners Derived from Deflation, Domain Decomposition and Multigrid Methods". In: *Journal of Scientific Computing* 39.3 (2009), pages 340–370. ISSN: 1573-7691. DOI: 10.1007/s10915-009-9272-6. URL: https://doi.org/10.1007/s10915-009-9272-6.

[100] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997. ISBN: 0898713617.

[101] Shaidurov V.V. *Multigrid Methods for Finite Elements*. Jan. 1995. ISBN: 978-94-015-8527-9.

[102] C. Vuik, R.R.P. van Nooyen, and P. Wesseling. "Parallelism in ILU-preconditioned GMRES". In: *Parallel Computing* 24.14 (1998), pages 1927–1946. ISSN: 0167-8191. DOI: https://doi.org/10.1016/S0167-8191(98)00084-2. URL: https://www.sciencedirect.com/science/article/pii/S0167819198000842.

[103] P. Wesseling. *An Introduction to Multigrid Methods*. An Introduction to Multigrid Methods. R.T. Edwards, 2004. ISBN: 9781930217089. URL: https://books.google.fr/books?id=RoRGAAAAYAAJ.

[104] S. Wold, A. Ruhe, H. Wold, and W. J. Dunn III. "The Collinearity Problem in Linear Regression. The Partial Least Squares (PLS) Approach to Generalized Inverses". In: *SIAM J. Sci. Stat. Comput.* 5.3 (Sept. 1984), pages 735–743. ISSN: 0196-5204. DOI: 10.1137/0905052. URL: https://doi.org/10.1137/0905052.

[105] In: *Domain Decomposition Methods in Science and Engineering XIX*. Edited by Huang Y., Kornhuber R., Widlund O., and Xu J. Springer-Verlag Berlin Heidelberg, 2011. ISBN: 978-3-642-11304-8.