Parallel-in-iteration optimization using multigrid-reduction-in-time

by

Guilherme Henrique Macieira de Araujo

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Applied Mathematics

Waterloo, Ontario, Canada, 2026

© Guilherme Henrique Macieira de Araujo 2026

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Felix Kwok Associate Professor, Dept. of Mathematics and Statistics Université Laval
Supervisor:	Hans de Sterck Professor, Dept. of Applied Mathematics University of Waterloo
Internal Members:	Giang Tran Associate Professor, Dept. of Applied Mathematics University of Waterloo
	Roberto Guglielmi Associate Professor, Dept. of Applied Mathematics University of Waterloo
Internal-External Member:	Walaa Moursi Associate Professor, Dept. of Combinatorics and Optimization, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Guilherme Henrique Macieira de Araujo was the sole author of Chapters 1 to 4 and 6, which were written under the supervision of Dr. Hans De Sterck and were not intended for publication. Guilherme Henrique Macieira de Araujo's contributions to Chapters 5 and 7 are detailed as follows:

Chapter 5: The research results presented in this chapter are based on a collaboration between Guilherme Henrique Macieira de Araujo, Dr. Hans De Sterck and Dr. Oliver Krzysik. It has been published in [3]. Guilherme Henrique Macieira de Araujo worked on the derivations and performed the numerical experiments. Editing, verification and formulation were provided by Dr. Hans De Sterck and Dr. Oliver Krzysik. The manuscript [3] was written and revised by Guilherme Henrique Macieira de Araujo for publication at Electronic Transactions on Numerical Analysis; we note that Chapter 5 consists of an extended version of that manuscript, with the added content not intended for publication.

Chapter 7: The research project presented in this chapter is the result of a collaboration between Guilherme Henrique Macieira de Araujo, Dr. Hans De Sterck and Dr. Oliver Krzysik. Guilherme Henrique Macieira de Araujo worked on the derivations and performed the numerical experiments. Preliminary editing and verification have been provided by Dr. Hans De Sterck and Dr. Oliver Krzysik. This project is currently being worked on for publication and will be made available as a standalone manuscript at a future date.

Abstract

Standard gradient-based iteration algorithms for optimization, such as gradient descent and its various proximal-based extensions to nonsmooth problems, are known to converge slowly for ill-conditioned problems, sometimes requiring many tens of thousands of iterations in practice. Since these iterations are computed sequentially, they may present a computational bottleneck in large-scale parallel simulations. In this thesis, we present a "parallel-in-iteration" framework that allows one to parallelize across these iterations using multiple processors with the objective of reducing the wall-clock time needed to solve the underlying optimization problem.

Our methodology is based on re-purposing parallel time integration algorithms for time-dependent differential equations, namely the parallel-in-time method of multigrid reduction-in-time (MGRIT), motivated by the fact that optimization algorithms often have interpretations as discretizations of time-dependent differential equations. We numerically demonstrate the efficacy of this parallel-in-iteration approach on two different model problems, including a standard convex quadratic problem and the nonsmooth elastic obstacle problem in one and two spatial dimensions. For our model problems, we observe fast MGRIT convergence analogous to its prototypical performance on partial differential equations of diffusion type. We present theoretical results to connect the convergence of MGRIT to the convergence of the underlying optimization algorithm; theoretically predicted parallel speedup results are also provided.

Having established our parallel-in-iteration framework, we develop an extension of this framework based on approximately computing multiple trajectories rather than computing the trajectory yielded by the underlying optimization method with high accuracy. These multiple trajectories are constructed by reinitializing our MGRIT parallel-in-iteration solve with updated initial guesses, assumed to be closer to the solution than then previous ones. Therefore, we may intepret our multiple-trajectory method as a restarting optimization algorithm. By truncating the time domain of the original MGRIT solve to a small fixed fraction of the original number of time points, our multiple-trajectories approach allows for a multiplicative increase in the estimated parallel speedup while simultaneously proportionally reducing the number of processors required for full parallelization. We present theoretical results on the asymptotic equivalence between standard and restarting MGRIT schemes, which support the strategy of restarting from an earlier time point employed by the truncated variant of the restarting scheme.

Acknowledgements

I would like to first thank my supervisor, Prof. Hans De Sterck, for the privilege of spending 4 years under his tutelage. From our first conversation before I received an invitation letter to the writing of this thesis, Hans has been nothing short of a role model in both professional and personal spheres. I extend this gratitude to Dr. Oliver Krzysik for his continued selfless support and guidance that went beyond the expectations of our professional relationship, acting as a *de facto* co-supervisor on many occasions. Now that my PhD journey is coming to end, I hope to be able to call Hans and Oliver friends for life, regardless of whether we collaborate again or cross paths professionally in any regard.

I would like to thank the remaining members of the thesis defence committee for allotting some of their precious time to carefully read, review and improve this thesis. To Associate Profs. Giang Tran and Walaa Moursi, I am grateful for the lessons learned as a student in your courses, for the recommended reading for my comprehensive exam and for how well you have treated me in all of our interactions. To Associate Profs. Roberto Guglielmi and Felix Kwok, I am thankful for having taken on such a time consuming responsibility on my behalf.

I extend my gratitude to all my instructors, colleagues and everyone else in the Department of Applied Mathematics for all I have learned in and outside classroom environments during my PhD program, for supporting me and creating a welcoming environment regardless of my position as a student, peer or teacher. I am grateful for the University of Waterloo for the continued financial assistance and, most importantly, for the opportunity to be here in the first place.

I would like to thank my MATH 235 Fall 2025 students for all I learned while being their instructor. The responsibility of teaching gave me a newfound sense of motivation and confidence, and I appreciate their patience and continued support during my early days as an instructor.

I would like to thank the professors and students from the Masters of Mathematics for Teachers courses, from whom I learned much during my time as a TA for the MMT courses.

On a personal note, I would like to thank my family, beginning with my parents Gutemberg and Francisca, for all the love, support and sacrifice throughout my entire life, but especially these past 4 years in Canada. The path I have chosen may have taken me far away, but I feel we are closer than ever. I can never repay all you have done for me, and all I can hope for is to make you proud.

I would like to thank my wife Manoela for the endless love, support, companionship and inspiration. I will never take for granted the leap of faith you took in coming here with me and for us braving the unknown together. You have made me into the man I am today and I would have achieved nothing in life had you not been by my side. Last, but not least, I would like to thank to my dog Cecil, my loving companion after long days away from home, and during many hours spent working from home. To have lived through the wonderful experiences we have shared these past 4 years is a dream come true unto itself.

Finally, I would like to thank all of my friends, family, colleagues, neighbours, friendly strangers, all who have played even the smallest part in my PhD studies and the writing of this thesis. If you were not mentioned here by name, if we have shared a single smile or laugh together, I am thankful to you too.

Dedication

I dedicate this thesis to my parents Gutemberg and Francisca, for their endless sacrifice, unwavering support and unconditional love, without whom this would not have been possible. All the joys and success I have achieved so far and will achieve someday stem directly from the blessing of being your child.

To my wife Manoela and dog Cecil, who make every day a happy one even in the hardest of times. I love you more than anything and aspire to become someone worthy of the luck I had in finding you. You are my source of inspiration and motivation, and who gave me the strength required to write this thesis.

To all my teachers, present and past, for sharing your knowledge and experience with kindness and generosity.

Table of Contents

Examining Committee Membership ii

Author's Declaration iii

Statement of Contributions iv

Abstract v

Acknowledgements vi

Dedication viii

List of Figures xv

List of Tables xxiii

List of Abbreviations xxvi

1 Introduction 1

2 Multigrid methods 5

2.1 Multigrid notation 5

2.1.1 Continuous and discrete boundary value problems 6

2.1.2 Laplacian matrices 8

2.1.3 Matrix ordering . . . 10
2.2 Poisson elliptic boundary value problems . . . 11
2.2.1 1D elliptic boundary value problem . . . 11
2.2.2 2D elliptic boundary value elliptic problem . . . 13
2.3 Iterative Methods . . . 14
2.3.1 Relaxation schemes . . . 16
2.3.2 Error smoothing (relaxation) . . . 18
2.3.3 The coarse grid approximation principle . . . 23
2.4 The multigrid method cycles . . . 24
2.4.1 Two-grid correction scheme . . . 24
2.4.2 V-cycle . . . 27
2.4.3 Full multigrid cycle . . . 29
2.5 Multigrid for nonlinear methods . . . 30
2.5.1 Newton's method . . . 30
2.5.2 Nonlinear Gauss-Seidel relaxation scheme . . . 32
2.5.3 Nonlinear iterative methods and Newton-multigrid . . . 33
2.5.4 The Full Approximation Scheme (FAS) . . . 34
2.6 Multigrid for time-dependent problems . . . 35
2.6.1 Heat equation . . . 36
2.6.2 Wave equation . . . 38

3 Convex optimization algorithms 39
3.1 Preliminaries . . . 39
3.1.1 Affine and convex sets . . . 39
3.1.2 Proper, lower semicontinuous and convex functions . . . 41
3.1.3 Subgradients and derivatives . . . 42
3.1.4 Lipschitz smoothness and strong convexity . . . 45
3.1.5 Strong convexity . . . 46

3.1.6 Minimizers . . . 47
3.1.7 Nonexpansive mappings and monotone operator theory . . . 49
3.1.8 Proximal operators . . . 51
3.2 Optimization algorithms for minimizing smooth functions . . . 54
3.2.1 Gradient descent (GD) . . . 54
3.2.2 Proximal point (PP) . . . 59
3.3 Splitting Algorithms for Minimizing Nonsmooth Sums of Functions . . . 61
3.3.1 Proximal gradient descent (PG) . . . 62
3.3.2 Alternating proximals (AP) . . . 65

4 Parallel-in-time multigrid methods **67**
4.1 Multigrid Reduction-In-Time (MGRIT) . . . 67
4.1.1 The MGRIT F- and C-relaxation schemes . . . 69
4.1.2 The MGRIT coarse-grid correction problem . . . 71
4.2 MGRIT from an algorithmic perspective . . . 74
4.2.1 Linear MGRIT algorithm . . . 74
4.2.2 FAS-MGRIT algorithm . . . 76
4.3 MGRIT from a multigrid operator perspective . . . 77
4.3.1 The MGRIT matrices . . . 81
4.4 MGRIT convergence theory under a diagonalizability assumption . . . 85
4.4.1 The MGRIT error propagation operators in block-matrix form . . . 85
4.4.2 Error propagation of one MGRIT iteration . . . 88
4.4.3 MGRIT convergence for diagonalizable operators . . . 90

5 Parallel-in-iteration optimization **93**
5.1 Other parallel optimization strategies . . . 94
5.1.1 Parallel-in-space optimization . . . 94
5.1.2 Parallel-in-time optimization . . . 96

5.2 Optimization Model Problems . . . 97
5.2.1 Model Problem 1 (MP1) - Quadratic minimization . . . 98
5.2.2 Model Problem 2 (MP2) - Elastic obstacle problem (EOP) . . . 98
5.3 Optimization algorithms . . . 102
5.3.1 Optimization algorithms for MP1 . . . 103
5.3.2 Optimization algorithms for MP2 . . . 103
5.4 Connections between optimization algorithms and differential equations . . . 104
5.4.1 Optimization methods as ODE discretizations . . . 104
5.4.2 Links between solving ODEs using MGRIT and parallel-in-iteration optimization . . . 106
5.5 MGRIT for parallel-in-iteration optimization . . . 108
5.5.1 Choice of Ψ for MP1 using gradient descent . . . 110
5.5.2 Choice of Ψ for MP2 using proximal gradient descent . . . 120
5.6 MGRIT convergence in the (generalized) gradient . . . 126
5.7 Numerical results . . . 130
5.7.1 MGRIT for gradient descent applied to MP1 . . . 132
5.7.2 MGRIT for proximal gradient applied to MP2 . . . 135
5.8 MGRIT parallel speedup estimates . . . 139
5.8.1 Estimating 2-level MGRIT wall-clock time speedup . . . 143
5.8.2 Estimating 3-level MGRIT wall-clock time speedup . . . 144
5.8.3 MGRIT parallel speedup estimates for MP1 and MP2 . . . 146
5.9 Conclusions . . . 149

6 Parallel-in-iteration optimization for momentum-accelerated algorithms 151
6.1 Momentum-accelerated algorithms . . . 152
6.1.1 Nesterov accelerated gradient descent (NAGD) . . . 152
6.1.2 Nesterov-accelerated proximal point algorithm (NAPP) . . . 153
6.1.3 FISTA . . . 153

6.1.4 Nesterov-accelerated alternating proximals (NAAP) 154
6.2 Connections between momentum-accelerated optimization algorithms and differential equations 154
6.2.1 Nesterov accelerated gradient descent (NAGD) and the limit of an ODE 155
6.2.2 Nesterov-accelerated proximal point (NAPP) and the limit of an ODE 157
6.2.3 FISTA and NAAP as ODEs 158
6.3 Momentum-accelerated parallel-in-iteration optimization using MGRIT . . 158
6.3.1 Choice of Ψ for MP1 using NAGD 158
6.4 Issues with parallel-in-iteration (P-in-I) solves of momentum-accelerated algorithms 161
6.4.1 Numerical MGRIT results for NAGD applied to MP1 162
6.4.2 Failure of the momentum-accelerated MGRIT operators 164
6.4.3 Parallel-in-iteration GD vs. sequential NAGD 167
6.5 Conclusions 168

7 Multi-trajectory parallel-in-iteration optimization 169
7.1 Limitations of MGRIT as a parallel-in-iteration solver 169
7.1.1 The multiple trajectories idea for MGRIT with an insufficient number of time points 171
7.2 MGRIT for multi-trajectory parallel-in-iteration optimization 174
7.2.1 Defining the multiple trajectories 174
7.2.2 The restarting MGRIT algorithm for optimization problems (R-MGRIT) 176
7.3 Asymptotic final-point analysis of MGRIT and R-MGRIT 181
7.3.1 MGRIT error propagation after k iterations 182
7.3.2 Characterizing (R-)MGRIT final-point approximate solutions 184
7.3.3 Error operators under a simultaneous diagonalizability assumption . 189
7.3.4 Asymptotic final-point (R-)MGRIT approximate solutions 192
7.3.5 Finite-time bounds for (R)-MGRIT 193

7.3.6 On a truncated R-MGRIT strategy . . . 197
7.4 Numerical results for MGRIT vs. R-MGRIT . . . 198
7.4.1 Numerical experiments for gradient descent applied to MP1 . . . 199
7.4.2 Numerical experiments for proximal gradient applied to MP2 . . . 202
7.5 R-MGRIT parallel speedup estimates . . . 206
7.5.1 2-level γ-truncated R-MGRIT parallel speedup . . . 207
7.5.2 3-level γ-truncated R-MGRIT parallel speedup . . . 208
7.5.3 R-MGRIT parallel speedup estimates for MP1 and MP2 . . . 210
7.6 On a greedy restart strategy . . . 212
7.6.1 Analysis of the MP1 optimal greedy restart index . . . 213
7.6.2 Estimating the MP1 optimal greedy restart index . . . 218
7.6.3 Numerical experiments for R-MGRIT vs. GR-MGRIT . . . 220
7.7 Conclusions . . . 223

8 Final remarks **226**
8.1 Other optimization algorithms for consideration . . . 227
8.1.1 Gradient descent-related methods with variable stepsizes . . . 227
8.1.2 Proximal-gradient related methods . . . 229
8.2 Other optimization problems . . . 231
8.2.1 Statistics and machine learning problems . . . 232
8.2.2 The convex feasibility problem (CFP) . . . 234
8.2.3 Minimizing sums of functions with linear constraints . . . 235
8.3 Conclusion . . . 236

References **237**

APPENDIX **247**

A Schur complement **248**

List of Figures

2.1 One-dimensional grid Ω_h on the interval $\Omega = (0,1)$ with constant grid spacing $h = 1/(N+1)$. The jth grid point is $x_j = jh$ for $0 \leq j \leq N+1$. 11

2.2 Two-dimensional grid Ω_h on the unit square $\Omega = (0,1)^2$. 13

2.3 Illustrations of the GS-RB update for 1 (a) and 2 (b) dimensions. 19

2.4 Illustration of 4-level V-cycle (algorithm 3) and W-cycle schemes, where h denotes the fine grid and $8h$ the coarsest grid. 28

2.5 Illustration of a 4-level full multigrid V-cycle scheme, where h denotes the fine grid and $8h$ the coarsest grid. 29

4.1 Fine- and coarse-grid discretization meshes with coarsening factor m. F-points (in black) are points t_i for $i = 1, \ldots, N_t$ such that $i \neq jm$ for $j = 1, \ldots, N_T = \frac{N_t}{m}$, and C-points (in blue) are points $T_j = t_{jm}$ for $j = 1, \ldots, N_T = \frac{N_t}{m}$. 69

4.2 Illustration of F- and C-relaxation schemes. Here, the blue squares represent C-points and the white circles represent F-points. The Φ symbol followed by an arrow denotes a forward time-step from one time point to the following one using the operator Φ. 70

5.1 **(a)** and **(b)**: obstacle ϕ and solution $\widehat{u}$ of Model Problem 2 (MP2)-1D (5.23). **(c)** and **(d)**: obstacle ϕ and solution $\widehat{u}$ of MP2-2D (5.23), computed numerically using the Proximal Gradient Descent (PG) algorithm (5.32). . 101

5.2 **(a)**: Convergence of Model Problem 1 (MP1), (5.15), with a standard sequential solve using Gradient Descent (GD) method (5.30); $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. **(b)**: Convergence of MP1, (5.15), with Multigrid Reduction-in-Time (MGRIT) using GD operator (5.30) on the fine grid and Proximal Point (PP) operator (5.31) on the coarse grid; $\mathbf{r}^{(k)}$ denotes residual (4.4) at iteration k, $\nabla f(\mathbf{v}_{N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point N_t and MGRIT iteration k, $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual. 133

5.3 **(a)**: 2-norm of $\nabla f\left(\mathbf{v}_{cm}^{(k)}\right)$, where ∇f is the gradient of f, and $\mathbf{v}_{cm}^{(k)}$ is the approximate solution at coarse time point cm, $c = 0, \ldots, N_T$, and iteration k. **(b)**: 2-norm of $\mathbf{r}_{cm}^{(k)}$, where $\mathbf{r}_{cm}^{k}$ is the residual (4.4) at coarse time point jm, $c = 0, \ldots, N_T$, and iteration k. 134

5.4 **(a)**: Convergence of MP2-1D, (5.23), with a standard sequential solve using PG method (5.32); $G_{sf}(\mathbf{u}_q)$ denotes the generalized gradient (5.145) of the qth sequential iterate. **(b)**: Convergence of MP2-1D, (5.23), with MGRIT using PG operator (5.32) on the fine grid and Alternating Proximals (AP) operator (5.35) on the coarse grid; $\mathbf{r}^{(k)}$ denotes residual (4.4) at iteration k, $G_{sF}(\mathbf{v}_{N_t}^{(k)})$ denotes generalized gradient (5.145) of the approximate solution at the final time point N_t and MGRIT iteration k, $G_{sF}(\mathbf{u}_0)$ denotes the generalized gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual. 136

5.5 **(a)**: 2-norm of $G_{sF}\left(\mathbf{v}_{cm}^{(k)}\right)$, where G_{sF} is the generalized gradient of F (5.145), and $\mathbf{v}_{cm}^{(k)}$ is the approximate solution at coarse time point cm, $c = 0, 1, \ldots, N_T$ and iteration k. **(b)** 2-norm of $\mathbf{r}_{cm}^{(k)}$, where $\mathbf{r}_{cm}^{(k)}$ is the residual (4.4) at coarse time point cm, $c = 0, 1, \ldots, N_T$, and iteration k. 137

5.6 **Left axis:** 2-norm of $(\mathbf{r}^{(k)})_i$, where $(\mathbf{r}^{(k)})_i$ is the ith spatial component of residual (4.4) at iteration k. **Right axis:** $\widehat{u}$ is the exact solution and ϕ is the obstacle of the 1D elastic obstacle problem (EOP), as in figure 5.1. The i points form a 256-point discretization of the interior of the domain $\Omega = [0, 3\pi]$ of both $\widehat{u}$ and ϕ. 138

5.7 **(a)**: Convergence of MP2-2D, (5.23), with a standard sequential solve using PG method (5.32); $G_{sf}(\mathbf{u}_q)$ denotes generalized gradient (5.145) of the qth sequential iterate. **(b)**: Convergence of MP2-2D, (5.23), with MGRIT using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $\mathbf{r}^{(k)}$ denotes the residual (4.4) at iteration k, $G_{sF}(\mathbf{u}_{N_t}^k)$ denotes the generalized gradient (5.145) of the approximate solution at the final time point N_t and MGRIT iteration k, $G_{sF}(\mathbf{u}_0)$ denotes the generalized gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual. 139

5.8 **(a)**: 2-norm of $G_{sF}\left(\mathbf{v}_{cm}^{(k)}\right)$, where G_{sF} is the generalized gradient of F, (5.145), and $\mathbf{v}_{cm}^{(k)}$ is the approximate solution vector at coarse time point cm, $c = 0, 1, \ldots, N_T$, and iteration k. **(b)**: 2-norm of $\mathbf{r}_{cm}^{(k)}$, where $\mathbf{r}_{cm}^{(k)}$ is residual (4.4) at coarse time point cm, $c = 0, 1, \ldots, N_T$, and iteration k. . . 140

5.9 2-norm of $(\mathbf{r}^{(10)})_{(i,j)}$, where $(\mathbf{r}^{(10)})_{(i,j)}$ is the (i,j)th spatial component of residual (4.4) for the 10th and final MGRIT iteration. 141

5.10 Contour plots of the approximate solution $\widehat{\mathbf{u}}$ and obstacle ϕ for the 2D EOP, as in figure 5.1, and the residual of the 10th and final MGRIT iteration in figure 5.9. 141

6.1 **(a)**: Convergence of MP1, (5.15), with a standard sequential solve using Nesterov's Accelerated Gradient Descent (NAGD) method (6.1) with fixed $\beta = 0.9262$; $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. **(b)**: Convergence of MP1, (5.15), with MGRIT using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\mathbf{r}^{(k)}$ denotes residual (4.4) at iteration k, $\nabla f(\mathbf{v}_{N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point N_t and MGRIT iteration k, $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual. 163

6.2 **(a)**: Convergence of MP1, (5.15), with a standard sequential solve using NAGD method (6.1) with fixed $\beta = 0.1$; $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. **(b)**: Convergence of MP1, (5.15), with MGRIT using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\mathbf{r}^{(k)}$ denotes residual (4.4) at iteration k, $\nabla f(\mathbf{v}_{N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point N_t and MGRIT iteration k, $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual. 163

6.3 Error of the $\Psi \approx \Phi^m$ approximation for the fixed-β NAGD (6.1) fine-grid operator Φ (6.34), and the fixed-β Nesterov-accelerated Proximal Point (NAPP) (6.5) coarse-grid operator Ψ (6.48), for varying levels of β, as measured in terms of the absolute difference of their smallest (a) and largest (b) eigenvalues. Here, $\beta^* = 0.9262$ is the optimal fixed β (6.2). 165

7.1 Convergence of MP1, (5.15), with a standard sequential solve using GD method (5.30); $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. Each trajectory, the first associated with (7.1) and the second with (7.3), corresponds to $\widetilde{N}_t = 20,000$ GD iterations, with the 10^{-8} gradient reduction threshold being crossed exactly at $q = 30,299$. 172

7.2 **Left:** 2-norm of $\nabla f\left(\mathbf{v}_{jm}^{(k)}\right)$, where $\mathbf{v}_{jm}^{(k)}$ is the approximate solution at coarse time point jm, $j = 0, \ldots, N_T$, and k is the MGRIT iteration index for the (7.1) system. **Right:** 2-norm of $\nabla f\left(\widetilde{\mathbf{v}}_{jm}^{(k)}\right)$, $\mathbf{v}_{jm}^{(k)}$ is the approximate solution at coarse time point jm, $j = 0, \ldots, N_T$, and k is the MGRIT iteration index for the (7.3) system. Here, $k = 6$ MGRIT iterations is sufficient to reach the desired 10^{-8} accuracy. 173

7.3 **Left:** 2-norm of $\nabla f\left(\mathbf{v}_{jm}^{(k)}\right)$, where $\mathbf{v}_{jm}^{(k)}$ is the approximate solution at coarse time point jm, $j = 0, \ldots, N_T$, and k is the MGRIT iteration index for the (7.1) system. **Right:** 2-norm of $\nabla f\left(\widetilde{\mathbf{v}}_{jm}^{(k)}\right)$, $\mathbf{v}_{jm}^{(k)}$ is the approximate solution at coarse time point jm, $j = 0, \ldots, N_T$, and k is the MGRIT iteration index for the (7.3) system. Here, $k = 6$ MGRIT iterations is sufficient to reach the desired 10^{-8} accuracy. 178

7.4 **(a)**: Convergence of MP1, (5.15), with a standard sequential solve using GD method (3.49); $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. **(b)**: Convergence of MP1, (5.15), with MGRIT using GD (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\nabla f(\mathbf{v}_{N_t}^{(k)})$ and $\nabla f(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denote the gradient of the approximate solution at the final time point N_t and iteration k of MGRIT and Restarted MGRIT (R-MGRIT), respectively, $\nabla f(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point γN_t and iteration k of γ-truncated R-MGRIT for $\gamma = 0.2$, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 200

7.5 2-norm of the gradient of the MP1, (5.15), approximate solutions at coarse time index j and iteration k of MGRIT, R-MGRIT and γ-truncated R-MGRIT (with $\gamma = 0.2$), from left to right, divided by the 2-norm of the gradient of the initial guess $\mathbf{u}_0$; $\nabla f(\mathbf{v}_{jm}^{(k)})$ and $\nabla f(\widetilde{\mathbf{v}}_{jm}^{(k)})$ denote the gradient of the approximate solution at the C-point jm for $j = 0, 1, \ldots, N_T$ (restricted to $j = 0, 1, \ldots, \overline{N_T}$ for truncated R-MGRIT) and iteration k of MGRIT and R-MGRIT, respectively, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 201

7.6 **(a)**: Convergence of MP2-1D, (5.23), with a standard sequential solve using PG method (5.32); $G_{sf}(\mathbf{u}_q)$ denotes the generalized gradient (5.145) of the qth sequential iterate. **(b)**: Convergence of MP2-1D, (5.23), with MGRIT using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $G_{sF}(\mathbf{v}_{N_t}^{(k)})$ and $G_{sF}(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denote the gradient of the approximate solution at the final time point N_t and iteration k of MGRIT and R-MGRIT, respectively, $G_{sF}(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point γN_t and iteration k of γ-truncated R-MGRIT for $\gamma = 0.2$, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 203

7.7 2-norm of the generalized gradient G_{sF} (5.145) of the MP2-1D, (5.23), approximate solutions at coarse time index j and iteration k of MGRIT, R-MGRIT and γ-truncated R-MGRIT (with $\gamma = 0.2$), from left to right, respectively, divided by the 2-norm of the gradient of the initial guess $\mathbf{u}_0$; $G_{sF}(\mathbf{v}_{jm}^{(k)})$ and $G_{sF}(\widetilde{\mathbf{v}}_{jm}^{(k)})$ denote the generalized gradient of the approximate solution at the C-point jm for $j = 0, 1, \ldots, N_T$ (restricted to $j = 0, 1, \ldots, \overline{N_T}$ for truncated R-MGRIT) and iteration k of MGRIT and R-MGRIT, respectively, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 204

7.8 **(a)**: Convergence of MP2-2D, (5.23), with a standard sequential solve using PG method (3.84); $G_{sf}(\mathbf{u}_q)$ denotes the generalized gradient (5.145) of the qth sequential iterate. **(b)**: Convergence of MP2-2D, (5.23), with MGRIT using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $G_{sF}(\mathbf{v}_{N_t}^{(k)})$ and $G_{sF}(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denote the gradient of the approximate solution at the final time point N_t and iteration k of MGRIT and R-MGRIT, respectively, $G_{sF}(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point γN_t and iteration k of γ-truncated R-MGRIT for $\gamma = 0.2$, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 205

7.9 2-norm of the generalized gradient G_{sF} (5.145) of the MP2-2D, (5.23), approximate solutions at coarse time index j and iteration k of MGRIT, R-MGRIT and γ-truncated R-MGRIT (with $\gamma = 0.2$), from left to right, respectively, divided by the 2-norm of the gradient of the initial guess $\mathbf{u}_0$; $G_{sF}(\mathbf{v}_{jm}^{(k)})$ and $G_{sF}(\widetilde{\mathbf{v}}_{jm}^{(k)})$ denote the generalized gradient of the approximate solution at the C-point jm for $j = 0, 1, \ldots, N_T$ (restricted to $j = 0, 1, \ldots, \overline{N_T}$ for truncated R-MGRIT) and iteration k of MGRIT and R-MGRIT, respectively, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 206

7.10 **Left axis**: modal scalar gradients $g_{j,n}$ (7.115) for modes $n = 1, \ldots, 10$ and C-point index $j = 0, \ldots, 100$. **Right axis**: squared gradient norm $\|\nabla f(\widetilde{\mathbf{v}}_{jm})\|^2$ (7.114), where $\nabla f(\widetilde{\mathbf{v}}_{jm})$ denotes the gradient of the C-point approximate solution at C-point index $j = 0, \ldots, 100$. 217

7.11 **Left axis**: absolute modal scalar gradients $|g_{j,n}|$ (7.115) for modes $n = 1, \ldots, 10$ and C-point index $j = 0, \ldots, 100$. **Right axis**: squared gradient norm $\|\nabla f(\widetilde{\mathbf{v}}_{jm})\|^2$ (7.114), where $\nabla f(\widetilde{\mathbf{v}}_{jm})$ denotes the gradient of the C-point approximate solution at C-point index $j = 0, \ldots, 100$. 218

7.12 Convergence of MP1, (5.15), with (G)R-MGRIT using GD (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\nabla f(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point N_t and iteration k of R-MGRIT and $\nabla f(\widehat{\mathbf{v}}_{t_k^*}^{(k)})$ denotes the gradient of the approximate solution at the optimal C-point t_k^* (7.111) and iteration k of Greedy Restarted MGRIT (GR-MGRIT), respectively, with $\nabla f(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ and $\nabla f(\widehat{\mathbf{v}}_{t_{k,\gamma}^*}^{(k)})$ denoting their analogues for γ-truncated R-MGRIT and GR-MGRIT, respectively, with $\gamma = 0.2$; $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 220

7.13 2-norm of the gradient of the MP1, (5.15), approximate solutions at coarse time index j and iteration k of R-MGRIT, GR-MGRIT, γ-truncated R-MGRIT and γ-truncated GR-MGRIT (both with $\gamma = 0.2$), from left to right, divided by the 2-norm of the gradient of the initial guess $\mathbf{u}_0$; $\nabla f(\widetilde{\mathbf{v}}_{jm}^{(k)})$ and $\nabla f(\widehat{\mathbf{v}}_{jm}^{(k)})$ denote the gradient of the approximate solution at the C-point jm for $j = 0, 1, \ldots, N_T$ (restricted to $j = 0, 1, \ldots, \overline{N_T}$ for truncated (G)R-MGRIT) and iteration k of R-MGRIT and GR-MGRIT, respectively, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 221

7.14 Convergence of MP2-1D, (5.23), with (G)R-MGRIT using PG (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $G_{sF}(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denotes the generalized gradient (5.145) of the approximate solution at the final time point N_t and iteration k of R-MGRIT and $G_{sF}(\widehat{\mathbf{v}}_{t_k^*}^{(k)})$ denotes the generalized gradient of the approximate solution at the optimal C-point t_k^* (7.111) and iteration k of GR-MGRIT, respectively, with $G_{sF}(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ and $G_{sF}(\widehat{\mathbf{v}}_{t_{k,\gamma}^*}^{(k)})$ denoting their analogues for γ-truncated R-MGRIT and GR-MGRIT, respectively, with $\gamma = 0.2$; $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 222

7.15 2-norm of the generalized gradient (5.145) of the MP2-1D, (5.23), approximate solutions at coarse time index j and iteration k of R-MGRIT, GR-MGRIT, γ-truncated R-MGRIT and γ-truncated GR-MGRIT (both with $\gamma = 0.2$), from left to right, divided by the 2-norm of the generalized gradient of the initial guess $\mathbf{u}_0$; $G_{sF}(\widetilde{\mathbf{v}}_{jm-1}^{(k)})$ and $G_{sF}(\widehat{\mathbf{v}}_{jm-1}^{(k)})$ denote the generalized gradient of the approximate solution at the F-point $jm-1$ for $j = 1, \ldots, N_T$ (restricted to $j = 1, \ldots, \overline{N_T}$ for truncated (G)R-MGRIT) and iteration k of R-MGRIT and GR-MGRIT, respectively, and $G_{sF}(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 223

7.16 Convergence of MP2-1D, (5.23), with (G)R-MGRIT using PG (5.32) on the fine grid and AP operator (5.35) on the coarse grid, with an additional post F-relaxation; $G_{sF}(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denotes the generalized gradient (5.145) of the approximate solution at the final time point N_t and iteration k of R-MGRIT and $G_{sF}(\widehat{\mathbf{v}}_{t_k^*}^{(k)})$ denotes the gradient of the approximate solution at the optimal C-point t_k^* (7.111) and iteration k of GR-MGRIT, respectively, with $G_{sF}(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ and $G_{sF}(\widehat{\mathbf{v}}_{t_{k,\gamma}^*}^{(k)})$ denoting their analogues for γ-truncated R-MGRIT and GR-MGRIT, respectively, with $\gamma = 0.2$; $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 224

7.17 2-norm of the generalized gradient (5.145) of the MP2-1D, (5.23), approximate solutions after post-relaxation at F-points of index $jm-1$ and iteration k of R-MGRIT, GR-MGRIT, γ-truncated R-MGRIT and γ-truncated GR-MGRIT (both with $\gamma = 0.2$), from left to right, divided by the 2-norm of the generalized gradient of the initial guess $\mathbf{u}_0$; $G_{sF}(\widetilde{\mathbf{v}}_{jm-1}^{(k)})$ and $G_{sF}(\widehat{\mathbf{v}}_{jm-1}^{(k)})$ denote the generalized gradient of the approximate solution at the F-point $jm-1$ for $j = 1, \ldots, N_T$ (restricted to $j = 1, \ldots, \overline{N_T}$ for truncated (G)R-MGRIT) and iteration k of R-MGRIT and GR-MGRIT, respectively, and $G_{sF}(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$. 225

8.1 Comparison of PG (3.84), AP (3.91), Douglas-Rachford (DR) (8.7) and Peaceman-Rachford (PR) (8.8) trajectories for minimizing $F(\mathbf{x})$ from (8.9), starting from the same initial guess. Here, x and y denote the coordinates of the $\mathbf{x} \in \mathbb{R}^2$ vectors generated by each optimization method. 231

List of Tables

5.1 Multigrid Reduction-in-Time (MGRIT) convergence results for different values of N, m and $\widehat{\ell}$, using Gradient Descent (GD) operator (5.30) on the fine grid and Proximal Point (PP) operator (5.31) on the coarse grid. 135

5.2 Model Problem 2 (MP2)-1D convergence results for different values of n, m and $\widehat{\ell}$, using Proximal Gradient Descent (PG) operator (5.32) on the fine grid and Alternating Proximals (AP) operator (5.35) on the coarse grid. . . 142

5.3 MP2-2D convergence results for different values of n, m and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. 142

5.4 Speedup estimates for Model Problem 1 (MP1), (5.15), different values of n and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 2.9$ for $n = 40$, $\alpha = 3.1$ for $n = 60$, and $\alpha = 3.5$ for $n = 80$. N_p denotes the number of processors required to achieve optimal speedup. 147

5.5 Speedup estimates for MP1, (5.15), for different values of n and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on a hypothetical lower bound for $\frac{t_c}{t_f}$ given by $\widehat{\alpha} = 1$. N_p denotes the number of processors required to achieve optimal speedup. 147

5.6 Speedup estimates MP2-1D, (5.23), for different values of n and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 2.4$ for $N = 256$, and $\alpha = 3.9$ for $N = 512$. N_p denotes the number of processors required to achieve optimal speedup. 147

5.7 Speedup estimates for MP2-1D, (5.23), for different values of n and $\widehat{\ell}$, using the PG operator (5.32) on the fine grid and the AP operator (5.35) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on an hypothetical lower bound for $\frac{t_c}{t_f}$ given by $\widehat{\alpha} = 1$. N_p denotes the number of processors required to achieve optimal speedup. 148

5.8 Speedup estimates for MP2-2D, (5.23), for different values of n and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 13$ for $n^2 = 64^2$, and $\alpha = 7$ for $n^2 = 128^2$. N_p denotes the number of processors required to achieve optimal speedup. 148

5.9 Speedup estimates for MP2-2D, (5.23), for different values of n and $\widehat{\ell}$, using the PG operator (5.32) on the fine grid and the AP operator (5.35) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on an hypothetical lower bound for $\frac{t_c}{t_f}$ given by $\widehat{\alpha} = 1$. N_p denotes the number of processors required to achieve optimal speedup. 148

7.1 MP1 (5.15) MGRIT convergence results for with different values of m and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid. In parentheses, results for Restarted MGRIT (R-MGRIT). Here, ℓ denotes the maximum number of levels for the given coarsening factor. For both $n = 60$ and $n = 80$, we have $\ell = 8$ for $m = 4$, $\ell = 4$ for $m = 16$, $\ell = 3$ for $m = 64$ and $\ell = 2$ for $m = 256$ and $m = 1024$. 202

7.2 MP2-1D MGRIT convergence results for different values of n, m and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. In parentheses, results for R-MGRIT; in square brackets, results for R-MGRIT with truncation factor $\gamma = 0.2$. For $m = 4$, we have $\ell = 8$ for $n = 256$ and $\ell = 10$ for $n = 512$. For both $n = 256$ and $n = 512$, we have $\ell = 5$ for $m = 16$, $\ell = 4$ for $m = 64$, $\ell = 3$ for $m = 256$ and $\ell = 2$ for $m = 1024$. 205

7.3 MP2-2D MGRIT convergence results for different values of n, m and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. In parentheses, results for R-MGRIT; in square brackets, results for R-MGRIT with truncation factor $\gamma = 0.2$. For both $n^2 = 64^2$ and $n^2 = 128^2$, we have $\ell = 7$ for $m = 4$, $\ell = 4$ for $m = 16$, $\ell = 3$ for $m = 64$ and $\ell = 2$ for $m = 256$. 207

7.4 Speedup estimates for MP1, (5.15), different values of n and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 3.1$ for $N = 60$, and $\alpha = 3.5$ for $N = 80$. N_p denotes the number of processors required to achieve optimal speedup. 211

7.5 Speedup estimates MP2-1D, (5.23), for different values of n and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 2.4$ for $N = 256$, and $\alpha = 3.9$ for $N = 512$. N_p denotes the number of processors required to achieve optimal speedup. 211

7.6 Speedup estimates for MP2-2D, (5.23), for different values of n and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 13$ for $N^2 = 64^2$, and $\alpha = 7$ for $N^2 = 128^2$. N_p denotes the number of processors required to achieve optimal speedup. 212

List of Abbreviations

ω-JAC ω-(damped) Jacobi 16, 17, 20–22

ADMM Alternating Direction Method of Multipliers 235, 236

AP Alternating Proximals xvi, xvii, xix, xxi–xxv, 62, 65, 66, 102, 104, 106, 120, 121, 123, 135, 136, 139, 142, 147–149, 151, 154, 199, 202, 203, 205, 207, 211, 212, 222, 224, 227, 229–231, 234

CBVP Continuous Boundary Value Problem 6, 7, 11, 30, 35, 36

CFL Courant-Friedrichs-Lewy 37, 38, 54, 56, 59, 113, 153, 234

CFP Convex Feasibility Problem 234, 235

CG Conjugate Gradient 228

CGC coarse-grid correction 26

DBVP Discrete Boundary Value Problem 7, 10–12, 14, 16, 17, 20–24, 26, 27, 29, 30, 35, 100

DR Douglas-Rachford xxii, 229–231, 234, 236

EOP elastic obstacle problem xvi, xvii, 98, 130, 135, 136, 138, 141, 202

FAS Full Approximation Scheme 5, 34, 35, 71, 73, 74, 76, 109, 130, 177

FISTA Fast Iterative Shrinkage-Thresholding Algorithm 2, 151, 153, 154, 158, 162, 168

FMG full multigrid 29, 35, 177

FNE firmly nonexpansive 49–52

FPGM Fast Proximal Gradient Method 154

GD Gradient Descent xvi–xviii, xx, xxiii–xxv, 54, 56–66, 93, 102, 103, 105–108, 110, 111, 113, 123, 125–129, 132, 133, 135, 147, 151–154, 157–159, 161–163, 166–168, 172–175, 180, 181, 198–202, 211, 220, 226–228, 234

GR-MGRIT Greedy Restarted MGRIT xx–xxii, 212, 213, 218–225

GS Gauss-Seidel 16–18, 22, 30, 32

HB Heavy Ball 152, 153, 156, 159

IBVP Initial Boundary Value Problem 36–38, 68, 100, 106, 108, 169, 180

IDC Integral Deferred Correction 164

ISTA Iterative Shrinkage-Thresholding Algorithm 153

LASSO Least Absolute Shrinkage and Selection Operator 233

LR Logistic Regression 233, 234

lsc lower semicontinuous 41, 42, 48, 51–53, 60, 61, 63

MAP Method of Alternating Projections 234

MGR Multigrid Reduction 3, 80, 81

MGRIT Multigrid Reduction-in-Time xvi–xx, xxiii, xxiv, 3, 4, 67–77, 80–82, 85–90, 92, 94, 97, 106, 108–111, 113, 114, 116, 118, 120, 123, 126, 128–133, 135, 136, 138, 139, 141, 143, 144, 146, 147, 149, 150, 161–164, 166–187, 192–210, 217–220, 223, 226, 228, 231

MM Majorization-Minimization 55, 56

MP1 Model Problem 1 xvi–xx, xxiii–xxv, 98, 100–103, 106–108, 110, 115, 120, 125–127, 129, 130, 133, 138, 139, 146, 147, 158, 159, 161–163, 167, 172, 174, 175, 180, 181, 189, 198–202, 210, 211, 213, 214, 220, 221, 226, 233, 234

MP2 Model Problem 2 xv–xvii, xix–xxv, 100–104, 110, 120, 123, 126, 127, 129, 130, 135, 136, 138, 139, 142, 146–149, 158, 162, 175, 198, 199, 202–207, 210–212, 220, 222–226, 230–233, 235

MT Multi-trajectory 173, 176, 180, 226

NAAP Nesterov-accelerated Alternating Proximals 154, 158

NAGD Nesterov's Accelerated Gradient Descent xvii, xviii, 151–154, 156–160, 162–168, 227, 228

NAPP Nesterov-accelerated Proximal Point xviii, 153, 154, 157–160, 162, 164, 165

NE nonexpansive 45, 49, 50

NR Newton-Raphson 30–33, 97, 228

ODE Ordinary Differential Equation 3, 6, 11, 67, 68, 94, 104–106, 108, 149, 152, 154–158, 162, 227, 230

OLS Ordinary Least Squares 232, 233

P-in-I parallel-in-iteration xiii, 39, 93, 104, 108, 151, 153, 158, 161, 164, 167–170, 173, 175, 176, 179, 180, 226–228, 231, 233, 235

P-in-T parallel-in-time 67, 93, 96, 97, 100, 108, 164, 176, 180

PD positive definite 100, 113

PDE Partial Differential Equation 3, 6, 13, 94, 96, 97, 105, 106, 110, 162–164, 168

PDEs Partial Differential Equations 6

PFASST Parallel Full Approximation Scheme in Space-Time 67, 94

PG Proximal Gradient Descent xv–xvii, xix, xxi–xxv, 59, 61–66, 93–95, 101–104, 106, 110, 120, 121, 123, 126, 127, 129, 135, 136, 139, 142, 147, 148, 151, 153, 154, 175, 198, 199, 202, 203, 205, 207, 211, 212, 222, 224, 226, 227, 229–231, 233, 234

PL Polyak-Lojasiewicz 59

PP Proximal Point xvi–xviii, xx, xxiii–xxv, 54, 59–66, 102, 103, 105, 106, 111, 113, 123, 125, 133, 135, 147, 151, 153, 154, 157–159, 161, 163, 199, 200, 202, 211, 220, 227, 228, 230, 234

PR Peaceman-Rachford xxii, 229–231, 234

R-MGRIT Restarted MGRIT xviii–xxii, xxiv, 176–179, 181, 182, 184–187, 192–210, 212–214, 218–225

RK Runge-Kutta 36

SD Steepest Descent 227, 228

SPD symmetric positive definite 10, 98, 100–102, 111, 113, 226, 230, 231, 233

SWR Schwarz Waveform Relaxation 164

Chapter 1

Introduction

Consider the unconstrained optimization problem,

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad f(\mathbf{u}), \tag{1.1}$$

with $f : \mathbb{R}^N \to \mathbb{R}$ a convex and L-smooth function, i.e., a convex function with L-Lipschitz gradient:

$$\|\nabla f(\mathbf{u}) - \nabla f(\mathbf{v})\| \leqslant L\|\mathbf{u} - \mathbf{v}\| \ \forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^N. \tag{1.2}$$

Such optimization problems are ubiquitous throughout the sciences, with many theoretical properties established for them and applications in fields such as machine learning [91, 92], signal reconstruction [10, 46, 16], game theory [43], econometrics [2] and mathematical physics [81, 93]. Various iterative gradient-based optimization algorithms for solving unconstrained minimization problems, such as gradient descent and Nesterov's accelerated gradient descent [70] take the form of 1- or 2-step updates. Generally, the convergence rates of these algorithms for convex functions can be slow. For example, gradient descent has a convergence rate of $\mathcal{O}(1/k)$ for problem (1.1), where k denotes the number of iterations; while Nesterov's improved rate of $\mathcal{O}(1/k^2)$ was shown in [71] to be optimal among all methods that only have information about the gradient of f for its updates. In practical terms, this means that many iterations are typically required to solve the problem; for example, tens of thousands of iterations is not uncommon [16, 74].

Related to (1.1) is the unconstrained problem of minimizing a sum of two functions

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad F(\mathbf{u}), \text{ where } F(\mathbf{u}) := f(\mathbf{u}) + g(\mathbf{u}), \tag{1.3}$$

where $f : \mathbb{R}^N \to \mathbb{R}$ is convex and L-smooth, and $g : \mathbb{R}^N \to (-\infty, +\infty]$ is continuous, convex and possibly nondifferentiable. The minimization problem (1.3) is typically solved by splitting algorithms [63], in which the differentiable f and possibly nondifferentiable g are handled differently. Many classical splitting algorithms also take the form of 1- or 2-step updates. Namely, the proximal gradient and Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) algorithms [10] are generalizations of gradient descent and Nesterov's accelerated gradient, respectively, for solving (1.3), maintaining the convergence rates of $\mathcal{O}(1/k)$ and $\mathcal{O}(1/k^2)$ as their counterparts do for (1.1). The convergence rates of these optimization algorithms are also slow, in practice requiring many iterations to get close to the true optimum [10, 16]. If gradient evaluations take some nontrivial amount of compute time, we may have to wait for a long wall-clock time to get an accurate approximation to the optimum because the algorithms are sequential in k.

One can write 1-step algorithms in the form $\mathbf{u}_{k+1} = \Phi(\mathbf{u}_k) + \mathbf{w}_{k+1}$, where $\Phi(\cdot)$ is a linear or nonlinear operator and $\mathbf{w}_{k+1}$ is a generic term that does not depend on $\mathbf{u}_k$, for example, to encode the initial condition or a forcing term. For example, we can write the gradient descent update for (1.1) given by

$$\mathbf{u}_{k+1} = \mathbf{u}_k - s\nabla f(\mathbf{u}_k), \quad k = 0, 1, \dots \tag{1.4}$$

as

$$\mathbf{u}_{k+1} = (I - s\nabla f)(\mathbf{u}_k). \tag{1.5}$$

Suppose we need K iterations to approximate the true optimum within the desired accuracy. Then, taking $\Phi := (I - s\nabla f)$, computing the K gradient descent iterations in parallel is equivalent to solving the below "all-at-once" system in parallel

$$\mathcal{A}(\mathbf{u}) = \begin{bmatrix} I & & & \\ -\Phi(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_K \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_K \end{bmatrix} = \mathbf{w}, \tag{1.6}$$

where for this particular case $\mathbf{w}_0$ is the initial condition and $\mathbf{w}_1 = \cdots = \mathbf{w}_K = \mathbf{0}$. We note that some 2-step methods, such as Nesterov's accelerated gradient descent [70] for solving (1.1), can be rewritten as 1-step algorithms of the form $[\mathbf{u}_{k+1} \quad \mathbf{v}_{k+1}]^T = \Phi([\mathbf{u}_k \quad \mathbf{v}_k]^T)$ for block vector inputs $[\mathbf{u}_k \quad \mathbf{v}_k]^T$, such that an analogous all-at-once system to (1.6) holds in such cases too.

The system (1.6) is exactly of the kind which parallel-in-time algorithms solve for time-dependent evolution equations [34, 73]. For example, the parareal method [62] is

a 2-level parallel-in-time method [41] which solves systems of differential equations by employing sequential coarse-grid solves to speed up solving the original fine-grid problem in parallel. However, parareal is limited in concurrency, as the coarse-grid solve is still sequential. Thus, the Multigrid Reduction-in-Time (MGRIT) [27] was developed as a multilevel generalization of parareal based on Multigrid Reduction (MGR) techniques [80].

In the linear case where $\Phi(\mathbf{u}_k)$ is computed by multiplying $\mathbf{u}_k$ by the matrix Φ, MGRIT solves the system (1.6) via a coarse-grid system derived from an approximation to the Schur complement of (1.6) in terms of the coarse-level variables, and has a factor of $m \in \mathbb{N}$ fewer variables on each successively coarser grid. Here, m is called the coarsening factor. The biggest challenge in applying MGRIT is developing a suitable coarse-grid operator $\Psi \approx \Phi^m$, where Φ^m is the operator appearing in the Schur complement. That is, Ψ should approximate m successive applications with Φ, but it should be computationally much less expensive than applying Φ m times; this cost reduction combined with a quickly converging method is what allows for speedup on a parallel machine. It has been well-documented that parallel-in-time methods such as parareal and MGRIT often work very well for diffusion-dominated (parabolic-type) Partial Differential Equation (PDE) problems [27, 24, 55]. Advection-dominated (hyperbolic-type) problems are more challenging, but careful construction of coarse-grid operators may facilitate fast convergence [55, 20, 23]. This provides motivation for the use of parallel-in-time methods for optimization problems such as (1.1) and (1.3).

For sufficiently small step lengths s as in (1.4), we can sometimes interpret an optimization algorithm as a discretization of a certain Ordinary Differential Equation (ODE), in which k acts as a "discrete time index," and the step length s is related to the time-scale of the ODE. For example as $s \to 0$, gradient descent can be thought of as an explicit Euler discretization of the gradient flow ODE,

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u} = -\nabla f(\mathbf{u}(t)), \quad t > 0, \tag{1.7}$$

and as $s \to 0$, Nesterov's accelerated gradient descent can be thought of as an explicit discretization [90] of the ODE,

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{u} + \frac{3}{t}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u} + \nabla f(\mathbf{u}(t)) = 0, \quad t > 0. \tag{1.8}$$

With these connections between solving optimization problems in parallel and solving systems of differential equations using parallel-in-time methods, we present a framework for implementing MGRIT for solving optimization problems in the classes (1.1) and (1.3)

in parallel. We describe two model problems belonging to the former and latter classes, and discuss the choices of fine- and coarse-grid operators for solving these problems with MGRIT. These operators are based on standard choices of algorithms for solving these problems sequentially, such that the MGRIT implementation can be interpreted as a sped-up version of these sequential algorithms.

The rest of this thesis is organized as follows. Chapter 2 presents an introduction to multigrid methods. Chapter 3 presents many key definition and facts regarding the optimization methods we consider in later chapters. Chapter 4 introduces the parallel-in-time method of choice for this thesis, MGRIT, with definitions of linear and nonlinear variants and a summary of the MGRIT error propagation and convergence analysis. Chapter 5 presents our parallel-in-iteration framework for optimization problems first presented in [3], featuring numerical experiments and theoretical parallel speedup estimates. Chapter 6 discusses the applying our parallel-in-iteration framework to momentum-accelerated algorithms, with exploratory numerical experiments and several remarks noting the issues and limitations of our framework with this intent. Chapter 7 provides an extension of our framework by considering applying our parallel-in-iteration to multiple trajectories, which provides a substantial improvement to the results found in chapter 5. Chapter 8 presents final remarks, mostly concerning possibilities for future work and extensions of the parallel-in-iteration framework developed in this thesis. We note that chapters 2 to 4 may be considered optional reading for readers familiar with their contents, while chapters 5 to 8 feature original work by the author of this thesis.

Chapter 2

Multigrid methods

In this chapter, we provide a systematic introduction to multigrid methods, with our main references being [94, 12]. Multigrid methods can fundamentally be viewed from two distinct but complementary perspectives: as a numerical solver for continuous boundary value problems or as an iterative linear solver for discrete algebraic systems [94]. We will discuss both interpretations to introduce the two basic principles behind the multigrid approach: error smoothing (or relaxation) and coarse-grid correction [94]. Building on this foundation, we will outline the basic multigrid algorithmic structures, known as multigrid cycles, and explore their extension to nonlinear problems via the Full Approximation Scheme (FAS) [94, 12]. If the reader is familiar with the fundamental aspects of multigrid methods, they may skip this chapter at their discretion.

2.1 Multigrid notation

We begin by introducing our multigrid notation, which mostly follows that of [94]. Here, we borrow the terminology of discrete differential operators, by which we call vectors *grid functions* and matrices *grid operators*, in order to emphasize the correspondence between discrete and continuous problems and between the discrete formulations on different grids of the multigrid structure.

2.1.1 Continuous and discrete boundary value problems

The problems of our interest follow from classical Ordinary Differential Equation (ODE) and Partial Differential Equation (PDE) boundary value problems described in terms of differential and boundary operators. We define the general d-dimensional continuous differential equation problem below.

Definition 1 (Continuous Boundary Value Problem (CBVP)). *Let $\boldsymbol{x} = (x_1, \cdots, x_d) \in \mathbb{R}^d$ be a d-dimensional vector. Let $\Omega \subset \mathbb{R}^d$ be a given open bounded domain with boundary $\Gamma \coloneqq \partial\Omega$. We denote scalar linear boundary value problems by*

$$\begin{aligned} L^{\Omega}u(\boldsymbol{x}) &= f^{\Omega}(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \\ L^{\Gamma}u(\boldsymbol{x}) &= f^{\Gamma}(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma, \end{aligned} \tag{2.1}$$

where

- *L^{Ω} is a linear differential operator on Ω,*
- *L^{Γ} represents one or several linear boundary operators,*
- *f^{Ω} denotes a given function on Ω,*
- *f^{Γ} denotes one or several functions on Γ, and*
- *$u = u(\boldsymbol{x})$ denotes a solution.*

For brevity, we simply write $Lu = f$ instead of (2.1).

The linear differential operators in (2.1) are defined by the ODE (for $d = 1$, as in (2.19)) or PDE (for $d = 2$, as in (2.27)) associated with the given CBVP. For Partial Differential Equations (PDEs), it is important to recall their classification into parabolic, elliptic and hyperbolic types, as well what we mean by linear in this differential context.

Definition 2 (Classification of PDEs). *Let $Lu = f$ be a CBVP in some domain $\Omega \in \mathbb{R}^2$, where*

$$Lu = a_{11}u_{xx} + a_{12}u_{xy} + a_{22}u_{yy} + a_1u_x + a_2u_y + a_0u, \tag{2.2}$$

with coefficients a_{ij}, a_i, a_0, $i, j \in \{1, 2\}$. We say that Lu is linear if its coefficients only depend on x and y, and say that Lu is quasilinear if the coefficients depend on x, y, u, u_x, u_y. Furthermore, we say that Lu is elliptic if $4a_{11}a_{22} > a_{12}^2$, hyperbolic if $4a_{11}a_{22} < a_{12}^2$ and parabolic if $4a_{11}a_{22} = a_{12}^2$.

While many CBVPs may be solved analytically, in practice these continuous problems are discretized so that they may be solved with computational numerical methods. We call this discretization of a CBVP a Discrete Boundary Value Problem (DBVP), which we define below. If the DBVP accurately approximates the CBVP, then the numerical solution of the DBVP is an accurate approximation (possibly exact) of the solution of the CBVP.

Definition 3 (DBVP)**.** *Define the infinite grid*

$$\mathbf{G}_h \coloneqq \{(x,y) : \ x = x_i = ih_x, \ y = y_j = jh_y; \ i,j \in \mathbb{Z}\}, \tag{2.3}$$

where $\boldsymbol{h} = (h_x, h_y)$ *is a vector of fixed mesh sizes. In particular, we consider square grids, such that we let* $h \coloneqq h_x = h_y$*. Then, we define* $\Omega_h = \Omega \cap \mathbf{G}_h$ *and the discrete boundary* Γ_h *is the set of points where the continuous boundary* Γ *intersects the lines of the grid.*

Let $Lu = f$ *be a CBVP as defined in definition 1. Then, its discrete analog for* $d = 2$ *is given by*

$$\begin{aligned} L_h^\Omega \mathbf{u}_h(x,y) &= \mathbf{f}_h^\Omega(x,y), \quad (x,y) \in \Omega_h, \\ L_h^\Gamma \mathbf{u}_h(x,y) &= \mathbf{f}_h^\Gamma(x,y), \quad (x,y) \in \Gamma_h, \end{aligned} \tag{2.4}$$

where

- *the discrete solution* $\mathbf{u}_h$ *is a grid function defined on the grid* $\Omega_h \cup \Gamma_h$*;*
- $\mathbf{f}_h^\Omega$*,* $\mathbf{f}_h^\Gamma$ *are discrete analogs of* f^Ω *and* f^Γ*, respectively, obtained by restricting them to the grid* $\Omega_h \cup \Gamma_h$ *(i.e.,* $\mathbf{f}_h^\Omega = f^\Omega \,|_{\Omega_h \cup \Gamma_h}$ *and* $\mathbf{f}_h^\Gamma = f^\Gamma \,|_{\Omega_h \cup \Gamma_h}$*);*
- L_h^Ω *is a discrete difference operator;*
- L_h^Γ *is a discrete boundary operator;*
- L_h^Ω *and* L_h^Γ *are grid operators (i.e., mappings between spaces of grid functions).*

When working with second-order finite-difference equations with Dirichlet boundary conditions, we further assume that discrete boundary equations are eliminated from the DBVP, which means $\mathbf{u}_h = \mathbf{f}_h^\Gamma, (x,y) \in \Gamma_h$. With this assumption, we can more compactly write the DBVP as $L_h \mathbf{u}_h = \mathbf{f}_h$, where $\mathbf{u}_h$ and $\mathbf{f}_h$ are grid functions on Ω_h and $L_h : \mathcal{G}(\Omega_h) \to \mathcal{G}(\Omega_h)$ is a linear operator, where $\mathcal{G}(\Omega_h)$ denotes the linear space of grid functions on Ω_h. We further explain and exemplify what we mean by elimination of boundary conditions in the discussion of the model problems presented in section 2.2.

2.1.2 Laplacian matrices

In this thesis, we consider continuous problems on d-dimensional domains $\Omega \subset \mathbb{R}^d$, $d \in \{1, 2\}$. Note that for $d \in \{1, 2\}$, we replace the $\mathbf{x} = (x_1, \ldots, x_d)$ vector notation presented in definition 1 by a simpler Cartesian coordinate notation: we write $\mathbf{x} = x$ when $d = 1$ and $\mathbf{x} = (x, y)$ when $d = 2$. The continuous domain Ω is discretized with a mesh Ω_h consisting of $N + 2$ total points, and N interior points, in each direction, equally separated by a distance of $h = 1/(N + 1)$.

Of particular relevance to our model problems are finite-difference discretizations of (negative) Laplacian operators, which we simply refer to as Laplacian matrices. We begin by defining the 1D Laplacian below.

Definition 4 (1D Laplacian)**.** *The 1D Laplacian A_1 is defined as*

$$A_1 = -\frac{1}{h^2}\begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & \ddots & \ddots & \ddots \\ & & 1 & -2 \end{bmatrix} \in \mathbb{R}^{N\times N}, \tag{2.5}$$

where $h = 1/(N + 1)$.

It can be shown that the 1D Laplacian A_1 in (2.5) has N distinct eigenvalues of the form

$$\lambda_k = \frac{2}{h^2}\left[1 - \cos(k\pi h)\right] = \frac{4}{h^2}\sin^2\left(\frac{k\pi h}{2}\right), \quad k = 1, 2, \ldots, N, \tag{2.6}$$

with corresponding eigenvectors of the form

$$\mathbf{w}_k = \sin\left(k\pi x\right), \quad x \in \Omega_h. \tag{2.7}$$

We note that the kth eigenvector $\mathbf{w}_k$ can be interpreted as the mode corresponding to the kth frequency of a Fourier expansion [12]. It also follows that the 2-norm of A_1 is given by

$$\|A_1\|_2 = \frac{4}{h^2}\sin^2\left(\frac{N\pi}{2(N+1)}\right) \approx \frac{4}{h^2} \quad \text{for } h \ll 1. \tag{2.8}$$

Similarly, we define the 2D Laplacian below.

Definition 5 (2D Laplacian)**.** *The 2D Laplacian A_2 is defined as*

$$A_2 = -\frac{1}{h^2}\begin{bmatrix} -T & I & & \\ I & -T & I & \\ & \ddots & \ddots & \ddots \\ & & I & -T \end{bmatrix} \in \mathbb{R}^{N^2\times N^2}, \tag{2.9}$$

where $h = 1/(N+1)$, $I \in \mathbb{R}^{N\times N}$ is the identity matrix and

$$T = \begin{bmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & \ddots & \ddots & \ddots \\ & & -1 & 4 \end{bmatrix} \in \mathbb{R}^{N\times N}. \tag{2.10}$$

It can be shown that the N^2 eigenvalues of the 2D Laplacian are given by

$$\begin{aligned} \lambda_{k,\ell} &= \frac{2}{h^2}\left[2 - \cos(k\pi h) - \cos(\ell\pi h)\right] \\ &= \frac{4}{h^2}\left[\sin^2\left(\frac{k\pi h}{2}\right) + \sin^2\left(\frac{\ell\pi h}{2}\right)\right], \quad k,\ell = 1,\ldots,N, \end{aligned} \tag{2.11}$$

which are the N^2 possible sums of the N eigenvalues of the 1D Laplacian matrices in the x and y directions. The eigenvalues above correspond to Fourier mode eigenvectors of the form

$$\mathbf{w}_{k,\ell} = \sin(k\pi x)\sin(\ell\pi y), \quad (x,y) \in \Omega_h. \tag{2.12}$$

It follows that the 2-norm of A_2 is

$$\|A_2\|_2 = \max_{k,\ell}|\lambda_{k,\ell}| = |\lambda_{N,N}| = \frac{8}{h^2}\sin^2\left(\frac{N\pi}{2(N+1)}\right) \approx \frac{8}{h^2} \quad \text{for } h \ll 1. \tag{2.13}$$

In order to simplify the definition of discrete operators L_h on a Cartesian grid, we occasionally appeal to the stencil notation of discrete operators [12, 94], which simplifies the representation of local grid operations. For example, the 1D Laplacian(2.5), can be represented by the three-point stencil

$$(A_1)_h = -\frac{1}{h^2}\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}_h, \tag{2.14}$$

while the 2D Laplacian, (2.9), has a corresponding five-point stencil of the form

$$(A_2)_h = \frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}_h ; \tag{2.15}$$

note that we typically omit the h subscript on both the stencil and the operator. The stencils above apply to all operations in the interior of Ω_h, but may need to be modified near the boundary Γ_h. However, this modification is straightforward in square grids, as the "out-of-bounds" coefficients simply become zero.

2.1.3 Matrix ordering

The DBVP discrete operators L_h from definition 3 are often represented by sparse symmetric positive definite (SPD) matrices A_h such as the 1D and 2D Laplacian, given by (2.5) and (2.9), respectively. By reordering the grid points, we change the position of the nonzero entries of A_h. In a lexicographic ordering, grid points are ordered column- or row-wise, starting with points at the left lower corner. For example, we can order the points of a 4×4 grid as

13	14	15	16
9	10	11	12
5	6	7	8
1	2	3	4

(2.16)

For the red-black ordering, first all unknowns at odd (red) grid points are considered, then the unknowns at the even (**black**) points. For the same 4×4 grid above, we get

15	7	**16**	**8**
5	13	**6**	**14**
11	3	**12**	**4**
1	9	**2**	**10**

. (2.17)

Note that the corresponding red-black ordered matrix A_h is a block matrix with blocks A_{rr}, representing the connections of the red grid points to red grid points, A_{rb} the connections of the red points to the black points, A_{br} the connections of black points to red points, and A_{bb} the connections of black points to black points:

$$A_h = \begin{bmatrix} A_{rr} & A_{rb} \\ A_{br} & A_{bb} \end{bmatrix}. \tag{2.18}$$

2.2 Poisson elliptic boundary value problems

In this section, we present two classical boundary value problems associated with 1D and 2D Poisson equations, which serve as model problems for this chapter. With these problems, we discuss how the DBVP is obtained from a CBVP, highlighting how the Laplacian operators introduced in subsection 2.1.2 bridge the continuous and differential equation problems. We note that these are purely spatial boundary value problems. We present them here for two reasons: *1.* they are the type of problem multigrid methods were originally designed for solving and *2.* allow for a simpler introduction to multigrid methods than time-dependent problems, which are we explore later in this thesis. We generally discuss time-dependent parabolic and hyperbolic boundary value problems in section 2.6; starting in chapter 4, we focus on one specific multigrid method for time-dependent problems.

2.2.1 1D elliptic boundary value problem

As an example, we first consider a prototypical elliptic CBVP, the 1D Poisson ODE with zero Dirichlet boundary conditions in the unit interval $\Omega = (0,1)$, with boundary $\Gamma := \partial\Omega = \{0,1\}$:

$$\begin{aligned} -\frac{d^2}{dx^2}u(x) &= f(x), \quad x \in \Omega, \\ u(x) &= 0, \quad x \in \Gamma. \end{aligned} \tag{2.19}$$

While this problem may possibly be solved analytically, in order to solve it numerically we must work with a corresponding DBVP. Thus, we consider discretizing the domain $\Omega = (0,1)$ with a mesh consisting of $N+2$ total points, and N interior points, equally separated by a distance of $h = 1/(N+1)$. We denote this discretized grid by Ω_h, and illustrate it in figure 2.1.

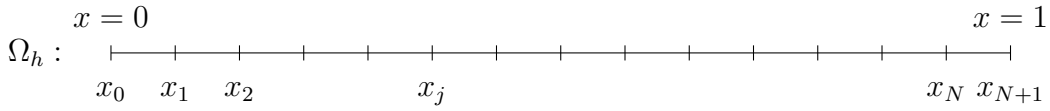


Figure 2.1: One-dimensional grid Ω_h on the interval $\Omega = (0,1)$ with constant grid spacing $h = 1/(N+1)$. The jth grid point is $x_j = jh$ for $0 \leq j \leq N+1$.

To discretize the continuous 1D Poisson problem, for each of the N interior points of Ω_h we substitute (2.19) by a *second-order finite difference approximation*. By using Taylor

series expansions centered on $u(x)$, we get

$$u(x+h) = u(x) + u'(x)h + \frac{u''(x)h^2}{2} + \frac{u'''(x)h^3}{6} + \mathcal{O}\left(h^4\right), \tag{2.20}$$

$$u(x-h) = u(x) - u'(x)h + \frac{u''(x)h^2}{2} - \frac{u'''(x)h^3}{6} + \mathcal{O}\left(h^4\right). \tag{2.21}$$

Adding up both equations above yields

$$u(x+h) + u(x-h) = 2u(x) + u''(x)h^2 + O\left(h^4\right), \tag{2.22}$$

from which we we get

$$u''(x) = \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} + O\left(h^2\right). \tag{2.23}$$

Thus, letting $(\mathbf{v}_h)_j$ be an approximation for $u(x_j)$, the exact solution at the jth grid point, we construct an approximate solution vector $\mathbf{v}_h = ((\mathbf{v}_h)_1, \ldots, (\mathbf{v}_h)_N)^T$. Noting that the boundary points x_0 and x_{N+1}, located at $x = 0$ and $x = 1$, respectively, must satisfy the zero Dirichlet boundary conditions in Γ, we set $(\mathbf{v}_h)_0 = (\mathbf{v}_h)_{N+1} = 0$. Therefore, the discretized 1D Poisson problem takes the form of the following linear system of equations:

$$\begin{aligned} \frac{-(\mathbf{v}_h)_{j-1} + 2(\mathbf{v}_h)_j - (\mathbf{v}_h)_{j+1}}{h^2} &= f\left(x_j\right), \quad 1 \leq j \leq N. \\ (\mathbf{v}_h)_0 = (\mathbf{v}_h)_{N+1} &= 0. \end{aligned} \tag{2.24}$$

Defining the right-hand side vector $\mathbf{f}_h = \left(f\left(x_1\right), \ldots, f\left(x_N\right)\right)^T = ((\mathbf{f}_h)_1, \ldots, (\mathbf{f}_h)_N)^T$, we represent the linear system above as

$$-\frac{1}{h^2}\begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & \ddots & \ddots & \ddots \\ & & 1 & -2 \end{bmatrix}\begin{bmatrix} (\mathbf{v}_h)_1 \\ (\mathbf{v}_h)_2 \\ \vdots \\ (\mathbf{v}_h)_N \end{bmatrix} = \begin{bmatrix} (\mathbf{f}_h)_1 \\ (\mathbf{f}_h)_2 \\ \vdots \\ (\mathbf{f}_h)_N \end{bmatrix}. \tag{2.25}$$

Hence, we can write the above system as the DBVP

$$L_h \mathbf{v}_h = \mathbf{f}_h, \tag{2.26}$$

where L_h here is the 1D Laplacian defined in (2.5). Finally, note that the $(\mathbf{v}_h)_0 = (\mathbf{v}_h)_{N+1} = 0$ condition of (2.24), which is captured by the definition of the 1D Laplacian (2.5), eliminates the Dirichlet zero boundary conditions.

2.2.2 2D elliptic boundary value elliptic problem

Now, we consider $d = 2$ and present the 2D analogue of (2.19), the 2D Poisson PDE with zero Dirichlet boundary conditions in the unit square $\Omega = (0,1)^2$ (with boundary $\Gamma \coloneqq \partial\Omega = \{(0,0),(0,1),(1,0),(1,1)\}$):

$$\begin{aligned} -\frac{\partial^2}{\partial x^2}u(x,y) - \frac{\partial^2}{\partial y^2}u(x,y) &= f(x,y), \quad (x,y) \in \Omega, \\ u(x,y) &= 0, \quad (x,y) \in \Gamma. \end{aligned} \tag{2.27}$$

Similarly to the 1D problem, we consider discretizing the boundary $\Omega = (0,1)^2$ with a Cartesian grid of equispaced points $(x_i, y_j) = (ih_x, ih_y)$, where $h \coloneqq h_x = h_y = 1/(N+1)$. We illustrate this two-dimensional grid Ω_h in figure 2.2 below.

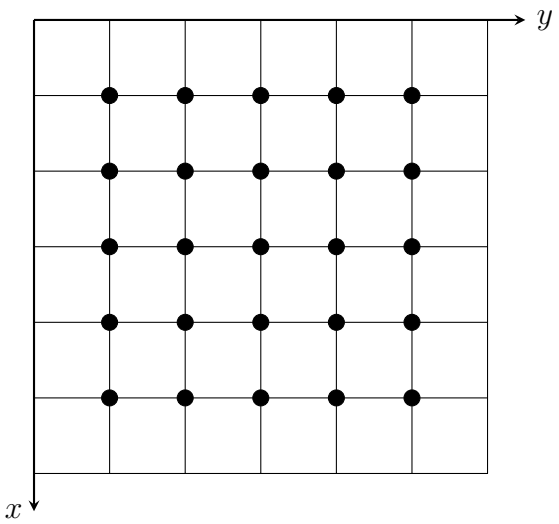


Figure 2.2: Two-dimensional grid Ω_h on the unit square $\Omega = (0,1)^2$.

Once again, we consider replacing the partial derivatives by second-order finite differences which arise from the Taylor expansions

$$\frac{\partial^2 u(x,y)}{\partial x^2} = \frac{u(x+h,y) - 2u(x,y) + u(x-h,y)}{h^2} + O\left(h^2\right), \tag{2.28}$$

$$\frac{\partial^2 u(x,y)}{\partial y^2} = \frac{u(x,y+h) - 2u(x,y) + u(x,y-h)}{h^2} + O\left(h^2\right). \tag{2.29}$$

Thus, the discretized 2D Poisson problem takes the form of a system of linear equations:

$$\begin{aligned} &\frac{-(\mathbf{v}_h)_{i-1,j} + 2(\mathbf{v}_h)_{ij} - (\mathbf{v}_h)_{i+1,j}}{h^2} + \frac{-(\mathbf{v}_h)_{i,j-1} + 2(\mathbf{v}_h)_{ij} - (\mathbf{v}_h)_{i,j+1}}{h^2} = (\mathbf{f}_h)_{ij}, \\ &(\mathbf{v}_h)_{i,0} = (\mathbf{v}_h)_{i,N+1} = (\mathbf{v}_h)_{0,j} = (\mathbf{v}_h)_{N+1,j} = 0, \quad 1 \leq i, j \leq N, \end{aligned} \tag{2.30}$$

where $(\mathbf{v}_h)_{i,j}$ is an approximation to the exact solution $u(x_i, y_j)$ at the grid points and $(\mathbf{f}_h)_{i,j} = f(x_i, y_j)$; note that we now have N^2 interior points. Letting $\mathbf{v}_{h,i} = ((\mathbf{v}_h)_{i,1}, \ldots, (\mathbf{v}_h)_{i,N})^T$ and $\mathbf{f}_{h,i} = ((\mathbf{f}_h)_{i,1}, \ldots, (\mathbf{f}_h)_{i,N})^T$, we obtain the block-linear system

$$-\frac{1}{h^2}\begin{bmatrix} -T & I & & \\ I & -T & I & \\ & \ddots & \ddots & \ddots \\ & & I & -T \end{bmatrix}\begin{bmatrix} \mathbf{v}_{h,1} \\ \mathbf{v}_{h,2} \\ \vdots \\ \mathbf{v}_{h,N} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{h,1} \\ \mathbf{f}_{h,2} \\ \vdots \\ \mathbf{f}_{h,N} \end{bmatrix}, \tag{2.31}$$

where T is defined in (2.10). Hence, we have a DBVP system of the form

$$L_h \mathbf{v}_h = \mathbf{f}_h, \tag{2.32}$$

where L_h is now the 2D Laplacian defined in (2.9). Similarly to the 1D case, note that the condition $(\mathbf{v}_h)_{i,0} = (\mathbf{v}_h)_{i,N+1} = (\mathbf{v}_h)_{0,j} = (\mathbf{v}_h)_{N+1,j} = 0$ for $1 \leq i, j \leq N$, which is absorbed by the definition of the 2D Laplacian, eliminates the zero Dirichlet boundary conditions.

2.3 Iterative Methods

The second interpretation of multigrid methods is related to iterative methods. Consider a linear system of equations of the form

$$A\mathbf{u} = \mathbf{f}, \tag{2.33}$$

such as (2.24) or (2.30), where A is a linear operator, $\mathbf{u}$ is the unique solution vector and $\mathbf{f}$ is some prescribed vector. Let $\mathbf{v}$ be an approximation solution, such that $\mathbf{v} \approx \mathbf{u}$. Then, we define the error as

$$\mathbf{e} = \mathbf{u} - \mathbf{v}. \tag{2.34}$$

Note that to know the error, we need to know the exact solution $\mathbf{u}$, which is what we want to find in the first place. Therefore, an alternative measure for the $\mathbf{v} \approx \mathbf{u}$ approximation is the residual, defined as

$$\mathbf{r} = \mathbf{f} - A\mathbf{v}. \tag{2.35}$$

Since the exact solution satisfies $\mathbf{f} = A\mathbf{u}$, we have that the right-hand side of (2.35) is equivalent to $\mathbf{f} - A\mathbf{v} = A\mathbf{u} - A\mathbf{v} = A(\mathbf{u} - \mathbf{v})$. Then, substituting (2.34) yields the *residual equation*

$$\mathbf{r} = A\mathbf{e}. \tag{2.36}$$

Note that the uniqueness of $\mathbf{u}$ together with (2.36) and (2.34) imply that

$$\mathbf{r} = \mathbf{0} \iff \mathbf{e} = \mathbf{0} \iff \mathbf{u} = \mathbf{v}. \tag{2.37}$$

The equations above inform the fundamental idea behind iterative methods. Suppose we have some approximation $\mathbf{v}$ for $\mathbf{u}$. Then, we compute the residual $\mathbf{r} = \mathbf{f} - A\mathbf{v}$, from which we solve residual equation $A\mathbf{e} = \mathbf{r}$ in order to obtain the error $\mathbf{e}$. Then, using the definition of the error we get $\mathbf{u} = \mathbf{v} + \mathbf{e}$. This is summarized by the *residual correction* update

$$\widehat{\mathbf{u}} = \mathbf{v} + A^{-1}\mathbf{r}. \tag{2.38}$$

By treating the $\widehat{\mathbf{u}}$ vector obtained here as the new $\mathbf{v}$, we can repeat this process iteratively until we finally get the $\widehat{\mathbf{u}} = \mathbf{u}$ such that $A\mathbf{u} = \mathbf{f}$.

Suppose we have some approximation B for A^{-1}. Letting $\mathbf{v}^{(0)}$ denote the initial approximation and $\mathbf{v}^{(1)}$ the update approximation, which play the roles of $\mathbf{v}$ and $\mathbf{u}$ in (2.38), respectively, we obtain the classical iterative method update

$$\mathbf{v}^{(1)} = \mathbf{v}^{(0)} + B\mathbf{r}. \tag{2.39}$$

Let $R = I - BA$ denote the *iteration matrix*. Then, note that by (2.39), we get

$$\begin{aligned} \mathbf{v}^{(1)} &= \mathbf{v}^{(0)} + B\mathbf{r} \\ &= (I - BA)\mathbf{v}^{(0)} + B\mathbf{f} \\ &\equiv R\mathbf{v}^{(0)} + B\mathbf{f}. \end{aligned} \tag{2.40}$$

More generally, after m iterations of this iterative method, we get an update of the form

$$\mathbf{v}^{(m)} = R^m\mathbf{v}^{(0)} + C(\mathbf{f}), \tag{2.41}$$

where m is a positive integer and $C(\mathbf{f})$ represents a series of operations on $\mathbf{f}$.

2.3.1 Relaxation schemes

Classical choices for B are related to the decomposition of A into a diagonal matrix D, a lower triangular matrix $-L$ and an upper triagonal matrix $-U$ that satisfy

$$A = D - L - U. \tag{2.42}$$

Then, (2.33) becomes

$$(D - L - U)\mathbf{u} = \mathbf{f}. \tag{2.43}$$

From the decomposition above, we obtain two classical iterative methods, called Jacobi and Gauss-Seidel (GS). We typically refer to these methods as relaxation schemes rather than iterative methods, and refer to one iteration of a relaxation scheme by a sweep.

The Jacobi relaxation scheme

Note that that isolating $D\mathbf{u}$ and multiplying both sides of the resulting equation by D^{-1} gives

$$D\mathbf{u} = (L + U)\mathbf{u} + \mathbf{f} \implies \mathbf{u} = D^{-1}(L + U)\mathbf{u} + D^{-1}\mathbf{f}. \tag{2.44}$$

Defining the iteration matrix

$$R_J = D^{-1}(L + U), \tag{2.45}$$

we obtain the update

$$\mathbf{v}^{(1)} = R_J\mathbf{v}^{(0)} + D^{-1}\mathbf{f}. \tag{2.46}$$

The relaxation scheme described by the above update is known as the Jacobi method, in which we approximate the inverse A^{-1} by D^{-1}.

For concreteness, we consider the setting of the 1D Poisson DBVP (see subsection 2.2.1), omitting the h subscripts in the notation. In terms of the v_j components of $\mathbf{v}$, the Jacobi update for the (2.24) is given by

$$(\mathbf{v})_j^{(1)} = \frac{1}{2}\left(\mathbf{v}_{j-1}^{(0)} + \mathbf{v}_{j+1}^{(0)} + h^2\mathbf{f}_j\right), \quad 1 \le j \le N; \tag{2.47}$$

note that this identical to solving the jth equation of (2.24) for the jth unknown using the current approximation for the $(j-1)$st and $(j+1)$st unknowns.

A variant of the Jacobi scheme can be obtained given a weight parameter $\omega \in \mathbb{R}$, for which we describe the component-wise ω-(damped) Jacobi (ω-JAC) update

$$\begin{aligned} \mathbf{z}_j^{(1)} &= \frac{1}{2}\left(\mathbf{v}_{j-1}^{(0)} + \mathbf{v}_{j+1}^{(0)} + h^2\mathbf{f}_j\right), \quad 1 \le j \le N, \\ \mathbf{v}_j^{(1)} &= (1-\omega)\mathbf{v}_j^{(0)} + \omega\mathbf{z}_j^{(1)} = \mathbf{v}_j^{(0)} + \omega\left(\mathbf{z}_j^{(1)} - \mathbf{v}_j^{(0)}\right), \quad 1 \le j \le N. \end{aligned} \tag{2.48}$$

Note that $\omega = 1$ recovers the original Jacobi update (2.47). In matrix form, the ω-JAC update is given by

$$\mathbf{v}^{(1)} = [(1-\omega)I + \omega R_J]\,\mathbf{v}^{(0)} + \omega D^{-1}\mathbf{f}. \tag{2.49}$$

where we define the ω-JAC iteration matrix as

$$R_\omega = (1-\omega)I + \omega R_J. \tag{2.50}$$

Note that we can equivalently write the ω-JAC update as

$$\mathbf{v}^{(1)} = R_\omega \mathbf{v}^{(0)} + \omega D^{-1}\mathbf{f} = \mathbf{v}^{(0)} + \omega D^{-1}\mathbf{r}^{(0)}. \tag{2.51}$$

The Gauss-Seidel (GS) relaxation scheme

Now, from (2.43), consider isolating $(D-L)\mathbf{u}$ and multiplying both sides of the resulting equation by $(D-L)^{-1}$:

$$(D-L)\mathbf{u} = U\mathbf{u} + \mathbf{f} \implies \mathbf{u} = (D-L)^{-1}U\mathbf{u} + (D-L)^{-1}\mathbf{f}. \tag{2.52}$$

Defining the iteration matrix

$$R_G = (D-L)^{-1}U, \tag{2.53}$$

we obtain the GS iterative method, described by the update

$$\mathbf{v} \longleftarrow R_G\mathbf{v}^{(0)} + (D-L)^{-1}\mathbf{f}, \tag{2.54}$$

where $\longleftarrow$ stands for overwriting; note that $(D-L)^{-1}$ provides the approximation of A^{-1} here. For the 1D Poisson DBVP (2.24), this is equivalent to

$$\mathbf{v}_j \longleftarrow \frac{1}{2}\left(\mathbf{v}_{j-1} + \mathbf{v}_{j+1} + h^2\mathbf{f}_j\right), \quad 1 \le j \le N. \tag{2.55}$$

Here, we note a key difference between GS and Jacobi. For Jacobi, since components are never overwritten, the order of the (2.47) update does not matter. However, for the GS update (2.55), the components of the new approximation are used as soon as they are computed, overwriting the old ones. Therefore, for GS, the order in which we update the components is significant.

While there are many possibilities for the GS update order, we briefly discuss two which are related to the discussion in subsection 2.1.3. The first is the lexicographic GS update, denoted by GS-LEX, which sweeps through the grid points in ascending order. Secondly,

we consider the red-black GS update, denoted by GS-RB, for which we update all even components first by

$$\mathbf{v}_{2j} \longleftarrow \frac{1}{2}\left(\mathbf{v}_{2j-1} + \mathbf{v}_{2j+1} + h^2\mathbf{f}_{2j}\right), \tag{2.56}$$

and then the odd components by

$$\mathbf{v}_{2j+1} \longleftarrow \frac{1}{2}\left(\mathbf{v}_{2j} + \mathbf{v}_{2j+2} + h^2\mathbf{f}_{2j+1}\right). \tag{2.57}$$

This update order is illustrated in figure 2.3b below. We relate this to the red-black partitioned A_h matrix in (2.18) by noting that the GS-RB iteration matrix can be written as

$$R_G = (I - B_b A)(I - B_r A), \tag{2.58}$$

where

$$B_r = \begin{bmatrix} (D_{rr} - L_{rr})^{-1} & 0 \\ 0 & 0 \end{bmatrix}, \quad B_b = \begin{bmatrix} 0 & 0 \\ 0 & (D_{bb} - L_{bb})^{-1} \end{bmatrix}; \tag{2.59}$$

D_{rr}, L_{rr}, D_{bb}, and L_{bb} denote the diagonals and strictly lower triangular parts of A_{rr} and A_{bb}, respectively. In particular, if $A_{rr} = D_{rr}$ and $A_{bb} = D_{bb}$, the red points can be updated simultaneously and independently using only the values of black points, and vice-versa for black points.

2.3.2 Error smoothing (relaxation)

We recall that the previously discussed relaxation schemes have the form

$$\mathbf{v}^{(1)} = R\mathbf{v}^{(0)} + \mathbf{g}, \tag{2.60}$$

where R is the corresponding iteration matrix and $\mathbf{g}$ is some fixed vector. We note that for methods of this form, if the sequence of approximations does converge to the exact solution, i.e., $\mathbf{v}^{(m)} \to \mathbf{u}$, then the exact solution $\mathbf{u}$ is a fixed point of the iteration:

$$\mathbf{u} = R\mathbf{u} + \mathbf{g}. \tag{2.61}$$

Letting $\mathbf{e}^{(m)} = \mathbf{u} - \mathbf{v}^{(m)}$ as in (2.34), the difference between (2.60) and (2.61) yields

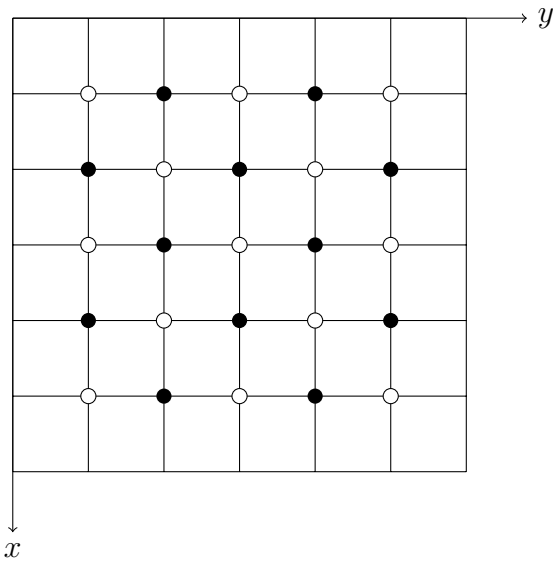
$$\mathbf{e}^{(1)} = R\mathbf{e}^{(0)}. \tag{2.62}$$

Repeating this argument, it follows that after the mth sweep of the relaxation scheme, the error of the mth approximation $\mathbf{v}^{(m)}$ is given by

$$\mathbf{e}^{(m)} = R^m\mathbf{e}^{(0)}. \tag{2.63}$$



(a) 1D GS-RB update.



(b) 2D GS-RB update.

Figure 2.3: Illustrations of the GS-RB update for 1 (a) and 2 (b) dimensions.

In this context, we call R the *error propagation matrix*, and say that the method of the form (2.60) is an *error smoothing* (or *relaxation*) scheme.

Given an error norm and its associated matrix norm, we can bound the error after the m sweeps by

$$\left\|\mathbf{e}^{(m)}\right\| \leq \|R\|^m \left\|\mathbf{e}^{(0)}\right\|. \tag{2.64}$$

Thus, if $\|R\| < 1$, the error goes to zero as the number of sweeps grows. Furthermore, it is a widely known fact [47, 87, 96, 85] that

$$\lim_{m\to\infty} R^m = 0 \iff \rho(R) < 1, \tag{2.65}$$

where $\rho(R)$ is the spectral radius of R. Therefore, it follows that the relaxation scheme associated with the matrix R converges for all initial guesses if and only if $\rho(R) < 1$. Note that we can interpret $\rho(R)$ as roughly the worst factor by which the error is reduced with each relaxation sweep if R is diagonalizable; in particular, if it is symmetric. In this context, the spectral radius $\rho(R)$ is also called the *asymptotic convergence factor*.

Convergence and error smoothing analysis of ω-JAC for the 2D Poisson DBVP

To exemplify the convergence and error smoothing analysis of a given relaxation scheme, through which we discuss key properties of multigrid methods, we follow an example from [94, Section 2.1]. Once again, consider the setting of the 2D Poisson DBVP (see subsection 2.2.2), given by

$$L_h \mathbf{v}_h = \mathbf{f}_h, \tag{2.66}$$

where L_h is 2D Laplacian, (2.9). For this problem, the Jacobi update (2.47) is given by

$$\begin{aligned} \mathbf{z}_h^{m+1}(x_i, y_j) =& \frac{1}{4}\left[h^2 \mathbf{f}_h(x_i, y_j) + \mathbf{v}_h^m(x_i - h, y_j) + \mathbf{v}_h^m(x_i + h, y_j)\right. \\ &\left. + \mathbf{v}_h^m(x_i, y_j - h) + \mathbf{v}_h^m(x_i, y_j + h)\right], \\ \mathbf{v}_h^{m+1} =& \mathbf{z}_h^{m+1}, \end{aligned} \tag{2.67}$$

with $(x_i, y_j) \in \Omega_h$, where $\mathbf{v}_h^m$ denotes the old approximation and $\mathbf{v}_h^{m+1}$ the new approximation of the iteration. The Jacobi iteration can be written as

$$\mathbf{v}_h^{m+1} = R_h \mathbf{v}_h^m + \frac{h^2}{4}\mathbf{f}_h \tag{2.68}$$

with the iteration operator

$$R_h = I_h - \frac{h^2}{4} L_h, \tag{2.69}$$

where I_h is the identity operator on Ω_h. Then, the ω-JAC update is given by

$$\mathbf{v}_h^{m+1} = \mathbf{v}_h^m + \omega\left(\mathbf{z}_h^{m+1} - \mathbf{v}_h^m\right) \tag{2.70}$$

for some $\omega \in \mathbb{R}$. The ω-JAC iteration operator $(R_\omega)_h = (1-\omega)I_h + \omega(R_J)_h$ on the Ω_h grid can then be written as

$$(R_\omega)_h = I_h - \frac{\omega h^2}{4} L_h = \frac{\omega}{4}\begin{bmatrix} & 1 & \\ 1 & 4\left(\frac{1}{\omega} - 1\right) & 1 \\ & 1 & \end{bmatrix}_h. \tag{2.71}$$

To analyze the convergence of ω-JAC, we consider the eigenfunctions of $(R_\omega)_h$, which are the eigenvectors of the 2D Laplacian L_h given in (2.12):

$$(\mathbf{w}_{k,\ell})_h(x, y) = \sin k\pi x \ \sin \ell\pi y, \quad (x, y) \in \Omega_h, \ k, \ell = 1, \ldots, N. \tag{2.72}$$

The corresponding eigenvalues of $(R_\omega)_h$, similarly related to the eigenvalues $\lambda_{k,\ell}$ in (2.11), are

$$(\lambda_{k,\ell})_h = (\lambda_{k,\ell})_h(\omega) = 1 - \frac{\omega}{2}(2 - \cos k\pi h - \cos \ell\pi h), \quad k, \ell = 1, \dots, N. \tag{2.73}$$

For the spectral radius $\rho\left((R_\omega)_h\right) = \max\left\{|(\lambda_{k,\ell})_h| : k, \ell = 1, \dots, N\right\}$, we obtain

$$\rho\left((R_\omega)_h\right) = |(\lambda_{1,1})_h| = |1 - \omega(1 - \cos \pi h)| = 1 - O\left(\omega h^2\right), \quad 0 < \omega \leq 1 \tag{2.74}$$

and $\rho\left((R_\omega)_h\right) \geq 1$ for $\omega \notin (0, 1]$ for sufficiently small h. Note that this spectral radius is approximately 1 regardless of the chosen weight $\omega \in (0, 1]$. Furthermore, the spectral radius gets closer to 1 as h increases; perhaps surprisingly, improving the accuracy of the solution by decreasing the grid spacing worsens the asymptotic convergence factor.

To analyze the error smoothing properties of ω-JAC, we consider the approximations before $(\mathbf{v}_h)$ and after $(\overline{\mathbf{v}}_h)$ one relaxation sweep and expand the errors before $(\mathbf{e}_h)$ and after $(\overline{\mathbf{e}}_h)$ the relaxation sweep, defined as

$$\mathbf{e}_h := \mathbf{u}_h - \mathbf{v}_h \quad \text{and} \quad \overline{\mathbf{e}}_h := \mathbf{u}_h - \overline{\mathbf{v}}_h, \tag{2.75}$$

into discrete eigenfunction series

$$\mathbf{e}_h = \sum_{k,\ell=1}^{N} c_{k,\ell}(\mathbf{w}_{k,\ell})_h, \quad \overline{\mathbf{e}}_h = \sum_{k,\ell=1}^{N} (\lambda_{k,\ell})_h c_{k,\ell}(\mathbf{w}_{k,\ell})_h, \tag{2.76}$$

where $c_{k,\ell}$ are Fourier expansion constants. In order to analyze the relaxation properties of $(R_\omega)_h$, we distinguish between *low* and *high* frequencies in this Ω_h grid.

Definition 6 (High and low frequencies of ω-JAC for the 2D Poisson DBVP)**.** *Let* $(x, y) \in \Omega_h$*, and consider the eigenfunctions*

$$(\mathbf{w}_{k,\ell})_h(x, y) = \sin k\pi x \sin \ell\pi y, \quad k, \ell = 1, \dots, N.$$

For $k, \ell \in \{1, \dots, N\}$*, we denote* $(\mathbf{w}_{k,\ell})_h$ *to be an eigenfunction of low frequency if* $\max(k, \ell) < (N + 1)/2$ *and of high frequency if* $(N + 1)/2 \leq \max(k, \ell) < N + 1$*.*

We note that low and high frequencies are qualitatively described as *smooth* and *oscillatory* modes, respectively. The property where the oscillatory modes are quickly eliminated while the smooth modes persist, which is a feature of many relaxation schemes, is known as *smoothing property* [94, 12]. In order to measure the smoothing property of $(R_\omega)_h$ quantitatively, we estimate its *smoothing factor*, which represents the worst factor by which high frequency error components are reduced with each sweep; note that the corresponding factor for low frequency components is simply the spectral radius (2.74). We define the smoothing factor below.

Definition 7 (Smoothing factor of ω-JAC for the 2D Poisson DBVP)**.** *The smoothing factor $\mu(h;\omega)$ of $(R_\omega)_h$, and its supremum μ^* over h, are defined as*

$$\begin{aligned}\mu(h;\omega) &:= \max\left\{|(\mathbf{w}_{k,\ell})_h(\omega)| : \frac{N+1}{2} \leq \max(k,\ell) \leq N\right\},\\ \mu^*(\omega) &:= \sup_{h\in\mathcal{H}} \mu(h;\omega),\end{aligned} \tag{2.77}$$

where $\mathcal{H}$ denotes the set of admissible mesh sizes.

Substituting the expression for $(\mathbf{w}_{k,\ell})_h(\omega)$ from (2.72) into (2.77), we get

$$\begin{aligned}\mu(h;\omega) &= \max\left\{\left|1-\frac{\omega}{2}(2-\cos k\pi h-\cos \ell\pi h)\right| : \frac{N+1}{2} \leq \max(k,\ell) \leq N\right\}\\ \mu^*(\omega) &= \max\{|1-\omega/2|, |1-2\omega|\}.\end{aligned} \tag{2.78}$$

In particular, note that $\mu(h;\omega) \geq 1$ if $\omega \leq 0$ or $\omega > 1$ and sufficiently small h, such that ω-JAC has no relaxation properties. However, for $\omega \in (0,1)$, the smoothing factor has an upper-bound smaller than 1 that is independent of h. For $\omega = 1$, we have a smoothing factor of $1 - O(h^2)$ only. In particular, we find that

$$\mu(h;\omega) = \begin{cases} \cos \pi h & \text{if } \omega = 1\\ (2+\cos \pi h)/4 & \text{if } \omega = 1/2\\ (1+2\cos \pi h)/5 & \text{if } \omega = 4/5\end{cases}, \quad \mu^*(\omega) = \begin{cases} 1 & \text{if } \omega = 1\\ 3/4 & \text{if } \omega = 1/2\\ 3/5 & \text{if } \omega = 4/5.\end{cases} \tag{2.79}$$

We note $\omega = 4/5$ is the optimal choice of $\omega \in [0,1]$, in the sense that

$$\inf\{\mu^*(\omega) : 0 \leq \omega \leq 1\} = \mu^*(4/5) = 3/5 \tag{2.80}$$

and

$$\inf\{\mu(h;\omega) : 0 \leq \omega \leq 1\} = \mu\left(h; \frac{4}{4+\cos \pi h}\right) = \frac{3\cos \pi h}{4+\cos \pi h} = \frac{3}{5} - \left|O\left(h^2\right)\right|. \tag{2.81}$$

Therefore, one step of ω-JAC with $\omega = 4/5$ reduces all high-frequency error components by at least a factor of $3/5$, independent of the grid size h.

We do not present the corresponding analysis for GS due to the fact that the eigenfunctions of L_h (i.e., the 2D Laplacian (2.9)) are not eigenfunctions of the GS operator (the 2D analog of (2.53)), such that this analysis becomes substantially more complicated. However, we present the results from [94, Section 2.1.3] of smoothing factors $\mu(\text{GS}-\text{LEX}) = 0.50$ and $\mu(\text{GS}-\text{RB}) = 0.25$ for this 2D Poisson DBVP setting.

2.3.3 The coarse grid approximation principle

Still in the 2D Poisson DBVP setting, we consider introducing a coarser grid Ω_{2h}; i.e., a grid with 2 times the spacing between each subinterval. For any given (k, ℓ), from (2.72) we have that for any $\mathbf{x} = (x, y) \in \Omega_{2h}$,

$$(\mathbf{w}_{k,\ell})_h(\mathbf{x}) = -(w_{N+1-k,\ell})_h(\mathbf{x}) = -(w_{k,N+1-\ell})_h(\mathbf{x}) = (w_{N+1-k,N+1-\ell})_h(\mathbf{x}) = (\mathbf{w}_{k,\ell})_{2h}(\mathbf{x}). \tag{2.82}$$

That is, these four distinct eigenfunctions in Ω_h, $(\mathbf{w}_{k,\ell})_h$, $-(w_{N+1-k,\ell})_h$, $-(w_{k,N+1-\ell})_h$ and $(w_{N+1-k,N+1-\ell})_h$ (also known as *harmonics* [94]), are indistinguishable in the coarser grid Ω_{2h}; we call this phenomenon *frequency aliasing* [94, 12]. We also note that for k or $\ell = (N+1)/2$, the $(w^{k,\ell})_h$ vanish on Ω_{2h}. From definition 6, we see that aliasing or vanishing described here occurs for all high frequency modes, such that only the low frequencies are visible on Ω_{2h}. Notably, the aliasing process makes it so that (k, ℓ)th mode on Ω^h becomes the $(n-k, n-\ell)$th mode on Ω^{2h} when $k, \ell > \frac{N+1}{2}$ [12]. That is, the high-frequency, oscillatory components of Ω^h effectively become low-frequency, smooth modes on Ω^{2h}.

Applying the distinction of low and high frequencies to the Fourier expansion representation of $\mathbf{e}_h$ (2.76), we can decompose it into partial sums:

$$\sum_{k,\ell=1}^{N} c_{k,\ell}(\mathbf{w}_{k,\ell})_h = \sum_{k,\ell=1}^{(N+1)/2-1} c_{k,\ell}(\mathbf{w}_{k,\ell})_h + \sum_{k,\ell=(N+1)/2 \le \max(k,\ell)}^{N} c_{k,\ell}(\mathbf{w}_{k,\ell})_h, \tag{2.83}$$

where the first and second partial sums correspond to the sums at low and high frequencies, respectively. By the smoothing property of the relaxation, after a certain number of sweeps the oscillatory high-frequency components become low-frequency, smooth components, such that an equivalent decomposition of the post-relaxation error $\bar{\mathbf{e}}_h$ (2.76) is dominated by its low-frequency partial sum. These low-frequency components are visible in Ω_{2h}, which means they can be approximated on the coarser grid.

Now, we assume additionally that $N+1 = 2^p$ for some power $p \in \mathbb{N}$, such that $h = 2^{-p}$. Then, the grid sequence obtained by doubling the mesh size successively is well-defined:

$$\mathcal{H} = \{\Omega_h, \Omega_{2h}, \Omega_{4h}, \ldots, \Omega_{h_0}\}, \tag{2.84}$$

where the coarsest grid Ω_{h_0} may possibly consist of only one interior grid point $(x, y) = (1/2, 1/2)$, such that $h_0 = 1/2$. For each pair of grids Ω_{mh} and Ω_{2mh} for $m \ge 1$, we can define a high-low frequency distinction similar to that in definition 6. As previously mentioned, by applying relaxation sweeps on the original Ω_h, the $(h, 2h)$-high frequency

error components are quickly reduced; the surviving low frequency error components are visible and can thus be approximated on Ω_{2h}. We repeat this process for the Ω_{2h}, Ω_{4h}, Ω_{8h}, etc. such that the high-frequency error components of each grid are eliminated and only the lowest-frequency components make it to the coarsest grid, Ω_{h_0}.

2.4 The multigrid method cycles

The *coarse-grid approximation principle* discussed in subsection 2.3.3, which says that quantities (namely, errors) that are smooth on a certain grid can be approximated on a coarser grid without any essential loss of information, together with the *error smoothing* property discussed in subsection 2.3.2, by which applying relaxation sweeps makes the error of the approximation smooth (not necessarily small) after a few sweeps, forms the core behind the multigrid method. We finally outline and define the general structure of multigrid methods in this section, combining the perspectives of multigrid as a boundary value problem solver (see section 2.2) and as an iterative method for linear systems of equations (see section 2.3).

2.4.1 Two-grid correction scheme

Here, on a grid Ω_h we consider a general DBVP problem of the form

$$L_h \mathbf{u}_h = \mathbf{f}_h \tag{2.85}$$

and assume that L_h^{-1} exists, and recall the discussion from section 2.3. For any approximation $\mathbf{v}_h$ of the solution $\mathbf{u}_h$, we denote the error by

$$\mathbf{e}_h := \mathbf{u}_h - \mathbf{v}_h \tag{2.86}$$

and the residual by

$$\mathbf{r}_h := \mathbf{f}_h - L_h \mathbf{v}_h, \tag{2.87}$$

for whom the residual equation (2.36) is given by

$$L_h \mathbf{e}_h = \mathbf{r}_h. \tag{2.88}$$

Given an approximation $\widehat{L}_h \approx L_h$, where $\widehat{L}_h$ is a "simpler" operator such that $\widehat{L}_h^{-1}$ exists, we can update the approximation via

$$\mathbf{v}_h \leftarrow \mathbf{v}_h + \widehat{\mathbf{e}}_h = \mathbf{v}_h + \widehat{L}_h^{-1} \mathbf{r}_h. \tag{2.89}$$

The iteration operator of the method obtained by successive applying this update, starting from some $\mathbf{v}_h^0$, is given by

$$M_h = I_h - B_h L_h, \quad M_h : \mathcal{G}(\Omega_h) \to \mathcal{G}(\Omega_h), \tag{2.90}$$

where $B_h := \left(\widehat{L}_h\right)^{-1}$ and I_h denotes the identity on $\mathcal{G}(\Omega_h)$. For the errors and for the residuals, respectively, we get

$$\begin{aligned} \mathbf{e}_h^{m+1} =& M_h \mathbf{e}_h = (I_h - B_h L_h)\,\mathbf{e}_h, && m = 0, 1, 2, \ldots, \\ \mathbf{r}_h^{m+1} =& L_h M_h L_h^{-1} \mathbf{r}_h = (I_h - L_h B_h)\,\mathbf{r}_h, && m = 0, 1, 2, \ldots. \end{aligned} \tag{2.91}$$

Letting the initial guess be $\mathbf{v}_h^0 = 0$, then we can represent the mth approximate solution $\mathbf{v}_h^m$ as

$$\begin{aligned} \mathbf{v}_h^m &= \left(I_h + M_h + M_h^2 + \cdots + M_h^{m-1}\right)\left(\widehat{L}_h\right)^{-1}\mathbf{f}_h \\ &= \left(I_h - M_h^m\right)\left(I_h - M_h\right)^{-1}\left(\widehat{L}_h\right)^{-1}\mathbf{f}_h \\ &= \left(I_h - M_h^m\right) L_h^{-1}\mathbf{f}_h. \end{aligned} \tag{2.92}$$

The asymptotic convergence properties of the above iterative process are characterized by the spectral radius (i.e., the asymptotic convergence factor) of the iteration operator M_h,

$$\rho(M_h) = \rho(I_h - B_h L_h) = \rho(I_h - L_h B_h). \tag{2.93}$$

Now, we recall the discussion from section 2.3 and introduce a coarser grid Ω_H (e.g., $H = 2h$), in which we use an invertible approximation L_H of L_h. Then, the residual equation (2.88) is approximated by the coarse-grid residual equation

$$L_H \widehat{\mathbf{e}}_H^m = \mathbf{r}_H^m, \tag{2.94}$$

where $L_H : \mathcal{G}(\Omega_H) \to \mathcal{G}(\Omega_H)$; we further assume that $\dim \mathcal{G}(\Omega_H) < \dim \mathcal{G}(\Omega_h)$ and that L_H^{-1} exists. Since $\mathbf{r}_H^m$ and $\widehat{\mathbf{e}}_H^m$ are grid functions on the coarser grid Ω_H, we assume the existence and availability of two linear transfer operators, one from Ω_h to Ω_H and another from Ω_h to Ω_H. The first, $I_h^H : \mathcal{G}(\Omega_h) \to \mathcal{G}(\Omega_H)$, is a restriction operator used to restrict $\mathbf{r}_h^m$ to Ω_H:

$$\mathbf{r}_H^m := I_h^H \mathbf{r}_h^m. \tag{2.95}$$

For concreteness, we consider the injection operator, which identifies grid functions at coarse grid points with the corresponding grid functions at fine grid points; namely, applying it to the residual yields

$$r_H(P) = I_h^H r_h(P) := \mathbf{r}_h(P) \quad \text{for } P \in \Omega_H \subset \Omega_h. \tag{2.96}$$

The second, $I_H^h : \mathcal{G}(\Omega_H) \to \mathcal{G}(\Omega_h)$, is an interpolation operator used to prolongate the correction $\widehat{\mathbf{e}}_H^m$ to Ω_h:

$$\widehat{\mathbf{e}}_h^m := I_H^h \widehat{\mathbf{e}}_H^m. \tag{2.97}$$

The process described thus far is known as coarse-grid correction (CGC) [94, 12]. We define one CGC step, which provides the update from $\mathbf{v}_h^m$ to $\mathbf{v}_h^{m+1}$, as follows.

Algorithm 1 Coarse-grid correction $\mathbf{v}_h \leftarrow CG_h(\mathbf{v}_h, \mathbf{f}_h)$

1: Compute the fine-grid residual $\mathbf{r}_h = \mathbf{f}_h - L_h\mathbf{v}_h$ and restrict it to the coarse grid by $\mathbf{r}_H = I_h^H \mathbf{r}_h$
2: Solve $L_H \widehat{\mathbf{e}}_H = \mathbf{r}_H$ on Ω_H.
3: Interpolate the coarse-grid error to the fine grid by $\mathbf{e}_h = I_H^h \mathbf{e}_H$ and correct the fine-grid approximation by $\mathbf{v}_h \leftarrow \mathbf{v}_h + \mathbf{e}_h$.

The operator representing one iteration of algorithm 1 is given by [94]

$$I_h - B_h L_h, \qquad \text{where } B_h = I_H^h L_H^{-1} I_h^H. \tag{2.98}$$

Note that since I_h^H maps $\mathcal{G}(\Omega_h)$ into $\mathcal{G}(\Omega_H)$, where $\dim \mathcal{G}(\Omega_H) < \dim \mathcal{G}(\Omega_h)$ by hypothesis, it is not bijective. Hence, it follows that $B_h = I_H^h L_H^{-1} I_h^H$ is not invertible, such that there exists some $\mathbf{v}_h \neq 0$ such that $B_h L_h \mathbf{v}_h = 0$. For such a $\mathbf{v}_h$, we have that $(I_h - B_h L_h)\mathbf{v}_h = \mathbf{v}_h$; that is, the coarse-grid correction scheme does not update this grid function error, such that the error is not reduced. Therefore, we have that

$$\rho\left(I_h - I_H^h L_H^{-1} I_h^H L_h\right) \geq 1; \tag{2.99}$$

that is, the CGC process does not converge on its own to a solution of the DBVP (2.85).

We then consider combining the CGC scheme with a relaxation scheme, which slowly but surely converges to a solution of the DBVP. We relax on the fine grid until the oscillatory modes have been succesfully smoothed and the convergence stalls, as it begins to be dictated by the slow-decaying smooth modes. Then, we relax on the residual equation on a coarser grid to obtain an approximation to the error itself. Finally, we return to the fine grid to correct the previous approximation with the approximate error. Each iteration step of a two-grid method, which we denote by *cycle*, consists of a pre-relaxation, a coarse grid correction and a post-relaxation part. One step of such an iterative two-grid method (calculating $\mathbf{v}_h^{m+1}$ from $\mathbf{v}_h^m$) proceeds as follows:

Algorithm 2 Two-grid correction scheme: $\mathbf{v}_h \leftarrow M_h(\mathbf{v}_h, \mathbf{f}_h)$

1: Relax ν_1 times on $L_h\mathbf{u}_h = \mathbf{f}_h$ on Ω_h with initial guess $\mathbf{v}_h$.
2: Compute the fine grid residual $\mathbf{r}_h = \mathbf{f}_h - L_h\mathbf{v}_h$ and restrict it to the coarse grid by $\mathbf{r}_H = I_h^H \mathbf{r}_h$
3: Solve $L_H\widehat{\mathbf{e}}_H = \mathbf{r}_H$ on Ω_H.
4: Interpolate the coarse-grid error to the fine grid by $\mathbf{e}_h = I_{2h}^h \mathbf{e}_{2h}$ and correct the fine-grid approximation by $\mathbf{v}_h \leftarrow \mathbf{v}_h + \mathbf{e}_h$.
5: Relax ν_2 times on $L_h\mathbf{u}_h = \mathbf{f}_h$ on Ω_h with initial guess $\mathbf{v}_h$.

We note that the following individual components of the two-grid cycle have to be specified: the relaxation scheme, the numbers ν_1, ν_2 of pre- and post-relaxation sweeps, respectively, the coarse grid Ω_H, the restriction operator I_h^H, the coarse grid operator L_H and the interpolation operator I_H^h. The choice of each of these components algorithm 2 may tremendously affect the convergence and efficiency of the two-grid method. Unfortunately, there are no general rules on how to choose them, although heuristics exist; for example ν_1 is often taken to be 1 or 2 [94, 12]. Hence, optimizing the choice of all parameters and operators for more complex problems may be very difficult.

2.4.2 V-cycle

We observe that algorithm 2 may converge very efficiently even though the coarse-grid residual equation is solved approximately via (2.94). Therefore, we consider replacing $\widehat{\mathbf{e}}_H^m$ by an approximation on an even coarser grid than Ω_H. Hence, we consider a sequence of progressively coarser and coarser grids Ω_{h_k} characterized by a sequence of mesh sizes h_k:

$$\{\Omega_{h_\ell}, \Omega_{h_{\ell-1}}, \ldots, \Omega_{h_0}\}. \tag{2.100}$$

For each Ω_{h_k}, we assume that linear operators

$$\begin{aligned} L_{h_k} &: \mathcal{G}(\Omega_{h_k}) \to \mathcal{G}(\Omega_{h_k}), \qquad & R_{h_k} &: \mathcal{G}(\Omega_{h_k}) \to \mathcal{G}(\Omega_{h_k}) \\ I_{h_k}^{k-1} &: \mathcal{G}(\Omega_{h_k}) \to \mathcal{G}(\Omega_{k-1}), \qquad & I_{k-1}^{m} &: \mathcal{G}(\Omega_{k-1}) \to \mathcal{G}(\Omega_{h_k}) \end{aligned} \tag{2.101}$$

are given, where the L_{h_k} are discretizations of L on Ω_{h_k} and R_{h_k} the linear iteration operators corresponding to the given relaxation methods on Ω_{h_k} for $k = \ell, \ldots, 0$.

We now describe a **multigrid cycle** of $(\ell + 1)$ grids to the DBVP

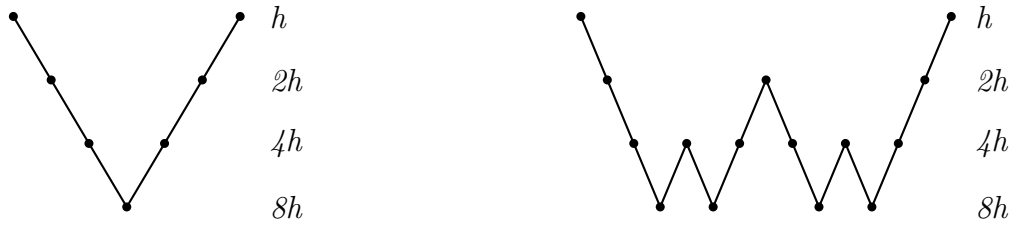
$$L_{h_\ell}\mathbf{u}_{h_\ell} = \mathbf{f}_{h_\ell} \tag{2.102}$$

on $\Omega_{h_\ell}$for a fixed $\ell \geq 1$. Given the operators L_{h_k}, for $k = 0, \ldots, \ell$ and R_{h_k}, $I_{h_k}^{h_k-1}$, $I_{h_k-1}^{h_k}$ for $k = 1, \ldots, \ell$, the numbers ν_1, ν_2 of pre- and post-relaxation sweeps and the starting fine grid $k = \ell$, we compute a new approximate solution $\mathbf{v}_k^{m+1}$ from a given approximation $\mathbf{v}_k^m$ to the solution $\mathbf{u}_k$ as defined in the following algorithm, which we call a *V-cycle* [94, 12]:

Algorithm 3 V-cycle Scheme: $\mathbf{v}_h \leftarrow V_h(\mathbf{v}_h, \mathbf{f}_h)$

1: Relax ν_1 times on $L_{h_k}\mathbf{u}_{h_k} = \mathbf{f}_{h_k}$ with a given initial guess $\mathbf{v}_h$.
2: **if** $k = 0$ (i.e., if on the coarsest grid) **then**
3: Solve $L_0\mathbf{u}_0 = \mathbf{f}_0$ and let $\mathbf{v}_0 \leftarrow \mathbf{u}_0$.
4: **else**
5: $\mathbf{f}_{h_k-1} \leftarrow I_{h_k}^{h_k-1}(\mathbf{f}_{h_k} - L_{h_k}\mathbf{v}_{h_k})$
6: $\mathbf{v}_{h_k-1} \leftarrow \mathbf{0}$
7: $\mathbf{v}_{h_k-1} \leftarrow V_{h_k-1}(\mathbf{v}_{h_k-1}, \mathbf{f}_{h_k-1})$
8: Correct $\mathbf{v}_{h_k} \leftarrow \mathbf{v}_{h_k} + I_{h_k-1}^{k}\mathbf{v}_{h_k-1}$
9: **end if**
10: Relax ν_2 times on $L_{h_k}\mathbf{u}_{h_k} = \mathbf{f}_{h_k}$ with initial guess $\mathbf{v}_{h_k}$.

We denote a V-cycle with ν_1 relaxation sweeps before the correction step and ν_2 relaxation sweeps after the correction step as a V(ν_1, ν_2)-cycle; the $V(1,1)$-cycle is often taken to be the default heuristic choice [94, 12]. The V-cycle defined in algorithm 3 is only one of multiple multigrid cycling schemes. For example, repeating step 7. from algorithm 3 twice yields the also commonly used *W-cycle* [94, 12]. We illustrate the V- and W-cycles in figure 2.4:



(a) V-cycle **(b)** W-cycle

Figure 2.4: Illustration of 4-level V-cycle (algorithm 3) and W-cycle schemes, where h denotes the fine grid and $8h$ the coarsest grid.

2.4.3 Full multigrid cycle

We expand on the V-cycle scheme, algorithm 3, by considering the *nested iterations* strategy for iterative methods [94, 12], which attempts to provide improved initial guesses for finer-grid problems by computing and interpolating approximations on coarser grids. After solving the DBVP on the coarsest grid, we interpolate (or prolongate) this solution to serve as an initial guess on the next finer grid level. On this new level, a V-cycle is applied to approximate the solution, which is then interpolated to the subsequent finer grid. This process is repeated until the finest grid is reached. We denote this scheme combining a V-cycle and nested iterations by full multigrid (FMG) V-cycle; we define it in algorithm 4 and illustrate it in figure 2.5, as follows.

Algorithm 4 Full Multigrid V-Cycle: $\mathbf{v}_h \leftarrow FMG_h(\mathbf{f}_h)$

if $k = 0$ (i.e., if on the coarsest grid) **then**
    Solve $L_0\mathbf{u}_0 = \mathbf{f}_0$ and let $\mathbf{v}_0 \leftarrow \mathbf{u}_0$.
else
    $\mathbf{f}_{h_k-1} \leftarrow I_{h_k}^{h_k-1}\mathbf{f}_{h_k}$
    $\mathbf{v}_{h_k-1} \leftarrow FMG_{h_k-1}(\mathbf{f}_{h_k-1})$
    Correct $\mathbf{v}_h \leftarrow I_{h_k-1}^{h_k}\mathbf{v}_{h_k-1}$
end if
Repeat $\mathbf{v}_{h_k} \leftarrow V_{h_k}(\mathbf{v}_{h_k}, \mathbf{f}_{h_k})$ ν_0 times.

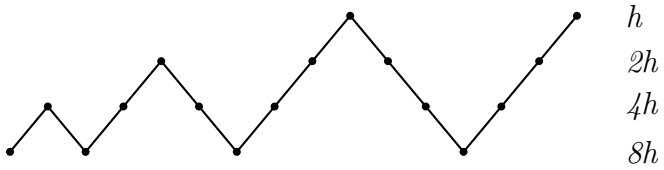


Figure 2.5: Illustration of a 4-level full multigrid V-cycle scheme, where h denotes the fine grid and $8h$ the coarsest grid.

2.5 Multigrid for nonlinear methods

Now, we consider applying multigrid methods to nonlinear problems. We consider the nonlinear equivalent of the CBVP defined in definition 1, described by

$$\begin{aligned} Nu &= f^{\Omega}, \quad (\Omega) \\ Bu &= f^{\Gamma}, \quad (\Gamma) \end{aligned} \tag{2.103}$$

where N is a nonlinear differential operator and B is a boundary operator acting on $\Gamma \coloneqq \partial\Omega$. Discretizing the nonlinear CBVP above on an Ω_h grid yields the nonlinear DBVP, analogue to the definition 3, described by the nonlinear system of discrete equations by

$$N_h(\mathbf{u}_h) = \mathbf{f}_h, \tag{2.104}$$

where $N_h : \mathcal{G}(\Omega_h) \to \mathcal{G}(\Omega_h)$ is a nonlinear operator and $\mathbf{u}_h \in \mathbb{R}^N$; we further assume the elimination of the boundary conditions, once again in the sense that they are implicitly contained in the discrete right-hand side term $\mathbf{f}_h \in \mathbb{R}^N$.

The nonlinear multigrid methods we consider require use of iterative methods for solving nonlinear systems of equations such as (2.104). For a nonlinear multigrid implementation, nonlinear relaxation schemes are needed as well. In particular, we consider the classical Newton-Raphson (NR) method (also known as Newton's method) for solving nonlinear equations and nonlinear GS and Jacobi relaxation schemes, the nonlinear analogues to the schemes discussed in subsection 2.3.1. We briefy discuss these methods before moving on to establish the nonlinear multigrid schemes of our interest. Note that the following discussion is once again based on [12, 94].

2.5.1 Newton's method

Newton's method for scalar equations

First, consider solving a scalar equation of the form $F(\mathbf{x}) = 0$ for some $F : \mathbb{R} \to \mathbb{R}$ and $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T \in \mathbb{R}^N$. Consider a truncated first-order Taylor expansion for F centred on an initial guess x,

$$F(x + s) = F(x) + sF'(x), \tag{2.105}$$

where the higher-order Taylor expansion terms are ignored. If $x + s$ is the solution, i.e. $F(x + s) = 0$, then we have

$$0 = F(x) + sF'(x) \implies s = -\frac{F(x)}{F'(x)}. \tag{2.106}$$

Then, the NR method consists of updating the initial guess x by

$$x \longleftarrow x - \frac{F(x)}{F'(x)}. \tag{2.107}$$

Newton's method for nonlinear systems of equations

Now, we consider expanding the NR method for a nonlinear system of equations of the form

$$\mathbf{F}(\mathbf{x}) \coloneqq \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_N(\mathbf{x}) \end{bmatrix} = \mathbf{0}, \tag{2.108}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_N) \in \mathbb{R}^N$, $f_i : \mathbb{R}^N \to \mathbb{R}$ for $i = 1, \ldots, N$, and $\mathbf{0} = \begin{bmatrix} 0 & 0 & \cdots & 0 \end{bmatrix}^T$ is the N-dimensional zero vector. Analogously to the scalar case, the truncated first-order Taylor series is given by

$$\mathbf{F}(\mathbf{x} + \mathbf{s}) = \mathbf{F}(\mathbf{x}) + J(\mathbf{x})\mathbf{s}, \tag{2.109}$$

where $J = (\partial(\mathbf{F})_i / \partial(\mathbf{x})_j)$ is the $N \times N$ Jacobian matrix. If $\mathbf{x} + \mathbf{s}$ is the solution, we have $\mathbf{F}(\mathbf{x} + \mathbf{s}) = \mathbf{0}$, which implies

$$\mathbf{0} = \mathbf{F}(\mathbf{x}) + J(\mathbf{x})\mathbf{s} \implies \mathbf{s} = -[J(\mathbf{x})]^{-1}\mathbf{F}(\mathbf{x}). \tag{2.110}$$

Then, the NR iteration is given by the update

$$\mathbf{x} \longleftarrow \mathbf{x} - [J(\mathbf{x})]^{-1}\mathbf{F}(\mathbf{x}). \tag{2.111}$$

Newton's method variants

In the following nonlinear multigrid discussion, we only consider the recently discussed classical NR method. However, in practice, the computation of the exact inverse of the Jacobian matrix, $[J(\mathbf{x})]^{-1}$, is often replaced by a suitable approximation. This results in a family of methods known as *quasi-Newton methods*, which differ in how this approximation for $[J(\mathbf{x})]^{-1}$ (or $J(\mathbf{x})$, directly) are chosen.

One example is *Picard's method* [94], in which we replace the Jacobian by some fixed linear approximation of the form $J(\mathbf{x}) \approx \gamma L$, where γ is some constant. Thus, Picard's update becomes

$$\mathbf{x} \longleftarrow \mathbf{x} - (\gamma L)^{-1}\mathbf{F}(\mathbf{x}). \tag{2.112}$$

For example, in a multigrid setting where on the Ω_h grid we must specify a linear operator L_h for (2.112), for some problems we consider taking L_h to be a Laplacian operator such as (2.5) and (2.9) [Chapter 5.3][94].

Another possible simplification of NR in an iterative setting, such as that in subsection 2.5.3, is that if the Jacobian needs to be applied to an iteratively generated sequence $\mathbf{x}_0,\ \mathbf{x}_1, \ldots$, instead of evaluating $J(\mathbf{x}_i)$ for every $i = 0, 1, \ldots$, we can simply consider the approximation $J(\mathbf{x}_i) \approx J(\mathbf{x}_0)$ for all $i = 1, 2, \ldots$; that is, we only to evaluate and invert the Jacobian once, and apply it as a fixed operator for all subsequent updates. This yields the *simplified NR method* [94] update

$$\mathbf{x} \longleftarrow \mathbf{x} - [J(\mathbf{x}_0)]^{-1}\mathbf{F}(\mathbf{x}). \tag{2.113}$$

2.5.2 Nonlinear Gauss-Seidel relaxation scheme

We describe nonlinear relaxation schemes abstractly as follows. Let $\mathbf{u} = (u_1, \ldots, u_N) \in \mathbb{R}^n$ and suppose the jth equation of the nonlinear system of equations

$$F_j\left(u_1, \ldots, u_N\right) = 0, \quad j = 1, \ldots, N \tag{2.114}$$

can be solved for the jth variable in terms of the other variables $(u_1, u_2, \ldots, u_{j-1}, u_{j+1}, \ldots, u_n)$. Then, we write the result as

$$u_j = N_j\left(u_1, u_2, \ldots, u_{j-1}, u_{j+1}, \ldots, u_N\right), \quad j = 1, \ldots, N. \tag{2.115}$$

Letting m and $m+1$ denote the indices of the current and new approximations, respectively, a nonlinear Jacobi iteration then reads

$$v_j = N_j\left(u_1, \ldots, u_{j-1}, u_{j+1}, \ldots, u_N\right), \tag{2.116}$$

where by the equality above we mean that we use the previous values at each of the components, while the nonlinear GS reads

$$v_j \leftarrow N_j\left(u_1, \ldots, u_{j-1}, u_{j+1}, \ldots, u_n\right), \tag{2.117}$$

where the arrow notation describes the overwriting of the updated right-hand side components; i.e., the current value of each component are used. In both cases, the relaxation updates above amount to solving one nonlinear equation for a single unknown, which can be solved with the scalar NR method. We can also consider variants of both the relaxation schemes and Newton's method, such as a nonlinear weighted Jacobi with Picard's method for the nonlinear scalar solve.

2.5.3 Nonlinear iterative methods and Newton-multigrid

Recall the discrete nonlinear system of equations from (2.104) given by

$$N_h(\mathbf{u}_h) = \mathbf{f}_h. \tag{2.118}$$

Let $\mathbf{v}_h$ be the current approximation to the exact solution $\mathbf{u}_h$, such that we define the error as in the linear case, $\mathbf{e}_h = \mathbf{u}_h - \mathbf{v}_h$ (2.34). Similarly to (2.35), we define the nonlinear residual as

$$\mathbf{r}_h = \mathbf{f}_h - N_h(\mathbf{v}_h). \tag{2.119}$$

Together, (2.104) and (2.119) imply

$$N_h(\mathbf{u}_h) - N_h(\mathbf{v}_h) = \mathbf{r}_h, \tag{2.120}$$

which is the nonlinear equivalent of (2.36). The key difference with respect to the linear setting is that $\mathbf{u}_h - \mathbf{v}_h = \mathbf{e}_h$ (2.34) does not imply that $N_h(\mathbf{u}_h) - N_h(\mathbf{v}_h) = N_h(\mathbf{e}_h)$, as is true when we have a linear operator L_h in place of the nonlinear operator N_h (2.36). Instead, the nonlinear residual equation (2.120) becomes

$$N_h(\mathbf{v}_h + \mathbf{e}_h) - N_h(\mathbf{v}_h) = \mathbf{r}_h. \tag{2.121}$$

A truncated first-order Taylor series expansion (i.e., omitting second- and higher-order terms) of $N_h(\mathbf{v}_h + \mathbf{e}_h)$ centred on $\mathbf{v}_h$ is given by

$$N_h(\mathbf{v}_h + \mathbf{e}_h) = N_h(\mathbf{v}_h) + \mathbf{e}_h J_h(\mathbf{v}_h), \tag{2.122}$$

where $J_h = (\partial(N_h)_i/\partial(\mathbf{u}_h)_j) \in \mathbb{R}^{N\times N}$ is the Jacobian matrix of N_h. Substituting this into (2.121) yields the linear system of equations

$$J_h(\mathbf{v}_h)\mathbf{e}_h = \mathbf{r}_h, \tag{2.123}$$

which approximates the exact nonlinear residual equation (2.120); note that (2.123) is a linear system of equations that can be computed using the linear multigrid methods we previously described. We also have by (2.123) that the NR update (2.111) for $\mathbf{v}_h$ is given by

$$\mathbf{v}_h \leftarrow \mathbf{v}_h - [(J_h)^{-1}(\mathbf{v}_h)]\mathbf{r}_h = \mathbf{v}_h + \mathbf{e}_h. \tag{2.124}$$

Hence, by solving (2.123) for $\mathbf{e}_h$ and iteratively updating $\mathbf{v}_h \leftarrow \mathbf{v}_h + \mathbf{e}_h$, we have obtained a method for approximately solving (2.104).

This combination of using NR for the *outer* nonlinear iteration (2.124) and multigrid for the *inner* linear iteration (2.123) is known as *Newton-multigrid* [12]. Notably, Newton-multigrid only makes use of multigrid for a linearization of one nonlinear equation, but does not employ multgrid for a nonlinear problem directly. Thus, we now consider a more fully-featured nonlinear multigrid solver.

2.5.4 The Full Approximation Scheme (FAS)

The nonlinear FAS multigrid method [94, 12] follows the same fundamental principles of linear multigrid methods and can also can be recursively defined in terms of a two-grid method. Recall the nonlinear residual equation (2.120) and suppose we have an approximation $\mathbf{v}_h$ for the exact solution $\mathbf{u}_h$ of (2.104). As in the linear problem, we consider approximating the error $\mathbf{e}_h$ (2.34) using the residual equation on the coarse grid Ω_H:

$$N_H\left(\mathbf{v}_H + \mathbf{e}_H\right) - N_H\left(\mathbf{v}_H\right) = \mathbf{r}_H, \tag{2.125}$$

where N_H denotes the coarse-grid operator, $\mathbf{r}_H$ is the coarse-grid residual, $\mathbf{v}_H$ is a coarse-grid approximation to $\mathbf{v}_h$, and $\mathbf{e}_H$ is a coarse-grid approximation to $\mathbf{e}_h$. Note the coarse-grid residual is obtained (in the exact same fashion as in the linear problem) by restricting the fine-grid residual to the coarse grid:

$$\mathbf{r}_H = I_h^H \mathbf{r}_h = I_h^H\left(\mathbf{f}_h - N_h\left(\mathbf{v}_h\right)\right), \tag{2.126}$$

where I_h^H is a restriction operator as in (2.101). Similarly, we restrict the current fine-grid approximation $\mathbf{v}_h$ to the coarse grid $\mathbf{v}_H$:

$$\mathbf{v}_H = I_h^H \mathbf{v}_h. \tag{2.127}$$

Substituting (2.126) and (2.127) on (2.125) yields

$$N_H(I_h^H \mathbf{v}_h + \mathbf{e}_H) = N_H\left(I_h^H \mathbf{v}_h\right) + I_h^H\left(\mathbf{f}_h - N_h\left(\mathbf{v}_h\right)\right), \tag{2.128}$$

where the only unknown is the exact solution

$$\mathbf{u}_H = I_h^H \mathbf{v}_h + \mathbf{e}_H. \tag{2.129}$$

Note that this is where the name *full approximation scheme* (FAS) comes from: instead of solving coarse-grid problem for the error $\mathbf{e}_H$, we solve a coarse-grid problem for the full approximation $\mathbf{u}_H = I_h^H \mathbf{v}_h + \mathbf{e}_H$. Having solved (2.128), we can interpolate the coarse-grid error approximation, $\mathbf{e}_H = \mathbf{u}_H - I_h^H \mathbf{v}_h$, up to the fine grid and used to correct the fine-grid approximation $\mathbf{v}_h$ by

$$\mathbf{v}_h \leftarrow \mathbf{v}_h + I_H^h \mathbf{e}_H = \mathbf{v}_h \leftarrow \mathbf{v}_h + I_H^h\left(\mathbf{u}_H - I_h^H \mathbf{v}_h\right). \tag{2.130}$$

A two-grid version of this scheme, the nonlinear analogue to the linear two-grid scheme, algorithm 2, is described as follows:

Algorithm 5 FAS two-grid correction scheme: $\mathbf{v}_h \leftarrow FAS_h(\mathbf{v}_h, \mathbf{f}_h)$

1: Restrict the current approximation and its fine-grid residual to the coarse grid: $\mathbf{r}_H = I_h^H(\mathbf{f}_h - N_h(\mathbf{v}_h))$ and $\mathbf{v}_H = I_h^H \mathbf{v}_h$.
2: Solve the coarse-grid problem $N_H(\mathbf{u}_H) = N_H(\mathbf{v}_H) + \mathbf{r}_H$.
3: Compute the coarse-grid approximation to the error for $\mathbf{u}_H$: $\mathbf{e}_H = \mathbf{u}_H - \mathbf{v}_H$.
4: Interpolate the error approximation up to the fine grid and correct the current fine-grid approximation: $\mathbf{v}_h \leftarrow \mathbf{v}_h + I_H^h \mathbf{e}_H$.

Note that if we apply algorithm 5 to a linear problem, i.e. with linear L_h, L_H operators instead of the nonlinear N_h, N_H operators, (2.125) simply recovers the linear coarse-grid problem, i.e., (2.94) with the exact error $\widehat{\mathbf{e}}_H = \mathbf{e}_H$. It follows that the corrections to the fine-grid approximations of algorithm 2 and algorithm 5 are also the same, such that algorithm 5 reduces exactly to the linear two-grid scheme, algorithm 2 [12]. Thus, FAS can be viewed as a generalization of the two-grid correction scheme to nonlinear problems. By recursively applying algorithm 5 as done in algorithm 3 and algorithm 4 for the linear algorithm 2, we can obtain FAS V-cycle and FMG schemes.

Another observation is that (2.128) can be written as

$$N_H(\mathbf{u}_H) = \mathbf{r}_H + \tau_h^H, \tag{2.131}$$

where

$$\tau_h^H \coloneqq N_H\left(I_h^H \mathbf{v}_h\right) - I_h^H N_h\left(\mathbf{v}_h\right). \tag{2.132}$$

We note that generally $\tau_h^H \neq 0$, such that solution $\mathbf{u}_H$ of the coarse-grid problem (2.128) is not the same as the solution of the original coarse-grid equation $N_H\left(\mathbf{u}_H\right) = \mathbf{f}_H$. However, as FAS iterates, we ideally expect $\mathbf{u}_H$ to eventually approximate the exact coarse-grid solution with an accuracy that compares to that of the solution on the finest grid, $\mathbf{u}_h$, but at the resolution of grid H [12].

2.6 Multigrid for time-dependent problems

Even though the multigrid schemes from section 2.4 and section 2.5 were defined for general DBVPs, the only model problem considered so far have been the elliptic Poisson problems from section 2.2, which are purely spatial problems. However, multigrid methods can also be applied to time-dependent CBVPs, such as those associated with hyperbolic problems, such as the wave equation, or parabolic problems, such as the heat equation. In order

to exemplify the use of multigrid for time-dependent problems, we briefly consider using multigrid methods for solving such problems, beginning with the heat equation. Our goal in the following is to illustrate how to handle the temporal components of a time-dependent problem in a multigrid setting, with a focus on the discretization of the temporal domain.

2.6.1 Heat equation

We consider the parabolic Initial Boundary Value Problem (IBVP) (similar to a CBVP, but with the addition of an initial condition that must be satisfied at the start of the temporal domain) for the 2D heat equation:

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}u(x,y,t) &= \frac{\mathrm{d}^2}{\mathrm{d}x^2}u(x,y,t) + \frac{\mathrm{d}^2}{\mathrm{d}y^2}u(x,y,t), && (x,y) \in \Omega, \quad t > 0, \\ u(x,y,0) &= u_0(x,y), && (x,y) \in \Omega, \quad t = 0, \\ u(x,y,t) &= f^{\Gamma}(x,y,t), && (x,y) \in \Gamma, \quad t > 0, \end{aligned} \tag{2.133}$$

where $(x,y) \in \Omega = (0,1)^2$ and $t \geq 0$, with the boundary of Ω given by $\Gamma \coloneqq \partial\Omega$. In order to discretize the IBVP above, we must consider whether to employ explicit or implicit *time discretization* schemes. A widely known family of both explicit and implicit discretization methods is that of Runge-Kutta (RK) methods. For brevity, we consider the simplest method in each respective category, which are the classical Euler methods.

The explicit *forward Euler* scheme for discretizing a function $u(x,y,t)$, with two spatial dimensions x, y and one temporal dimension t, reads

$$\frac{u_{h,\tau}(x,y,t+\tau) - u_{h,\tau}(x,y,t)}{\tau} = -L_h u_{h,\tau}(x,y,t), \tag{2.134}$$

while the implicit *backward Euler* scheme is given by

$$\frac{u_{h,\tau}(x,y,t+\tau) - u_{h,\tau}(x,y,t)}{\tau} = -L_h u_{h,\tau}(x,y,t+\tau), \tag{2.135}$$

where τ is the step size in the t-direction, h the grid spacing of the Ω_h discretization mesh, and $L_h = A_2$ is the 2D Laplacian operator defined in (2.9). Both Euler schemes' time discretization accuracy is of the order $O(\tau)$; the smaller the time step, the more accurate the discretizations are. We say that forward Euler is an explicit method since the values of $u_{h,\tau}(x,y,t+\tau)$ can be explicitly computed given $u_{h,\tau}(x,y,t)$:

$$\begin{aligned} u_{h,\tau}(x,y,t+\tau) &= -\tau L_h u_{h,\tau}(x,y,t) + u_{h,\tau}(x,y,t) \\ &= (I - \tau L_h) u_{h,\tau}(x,y,t). \end{aligned} \tag{2.136}$$

On the other hand, the implicit backward Euler scheme requires the updated values of $u_{h,\tau}$ at time $t+\tau$:

$$u_{h,\tau}(x,y,t+\tau) = u_{h,\tau}(x,y,t) - \tau L_h u_{h,\tau}(x,y,t+\tau), \tag{2.137}$$

which we rearrange to get

$$(I+\tau L_h)u_{h,\tau}(x,y,t+\tau) = u_{h,\tau}(x,y,t). \tag{2.138}$$

or, equivalently,

$$u_{h,\tau}(x,y,t+\tau) = (I+\tau L_h)^{-1}u_{h,\tau}(x,y,t) \tag{2.139}$$

While simpler and computationally cheaper than their implicit counterparts, explicit discretization schemes such as forward Euler are limited by their Courant-Friedrichs-Lewy (CFL) stability condition, which establishes an upper bound that must be satisfied by the stepsize τ. For forward Euler, this CFL condition is given by

$$\tau \leq \frac{h^2}{2}. \tag{2.140}$$

This restriction is absent in certain implicit time discretization schemes such as backward Euler, which is unconditionally stable.

For example, if we consider using a backward Euler discretization for the discretized parabolic IBVP, for every time step (from t to $t+\tau$) we must solve local problems of the form

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4+h^2/\tau & -1 \\ & -1 & \end{bmatrix}_{h,\tau} u_{h,\tau}(x,y,t+\tau) = \frac{h^2}{\tau}u_{h,\tau}(x,y,t), \tag{2.141}$$

where the stencil above corresponds to the $\frac{1}{\tau}(I+\tau L_h)$ operator; more generally, the right-hand side above for implicit schemes depends only on known values from previous time steps t and $t-\tau$. Given the unboundedness of implicit time steps, in many practical cases we consider taking arbitrarly large time steps. We note that as $\tau \to \infty$, the stencil of $\frac{1}{\tau}(I+\tau L_h)$ approaches that of L_h, i.e. of the (2.9), in which case the considerations for the elliptic Poisson problem apply. On the other hand, the smaller τ becomes, i.e., $\tau \to 0$, the more diagonally dominant the grid operator becomes, which is associated with faster convergence of iterative methods [94, 12].

2.6.2 Wave equation

As a canonical hyperbolic problem, we present the 2D wave equation IBVP,

$$\begin{aligned}
\frac{\mathrm{d}^2}{\mathrm{d}t^2}u(x,y,t) &= \frac{\mathrm{d}^2}{\mathrm{d}x^2}u(x,y,t) + \frac{\mathrm{d}^2}{\mathrm{d}y^2}u(x,y,t) && (x,y)\in\Omega, \quad t\in\mathbb{R}, \\
u(x,y,0) &= u_0(x,y) && (x,y)\in\Omega, \quad t=0 \\
u_t(x,y,0) &= u_1(x,y) && (x,y)\in\Omega, \quad t=0 \\
u(x,y,t) &= f^{\Gamma}(x,y,t) && (x,y)\in\Gamma, \quad t\in\mathbb{R}
\end{aligned} \tag{2.142}$$

for $(x,y)\in\Omega=(0,1)^2$ with $\Gamma\coloneqq\partial\Omega$. While for brevity we do not go into further detail (see [94, Example 2.8.3] for a detailed analysis of this example), in general the approach is similar to that for the parabolic IBVP after accounting for the additional initial condition for the first time derivative. We also note that the CFL stability condition for explicit schemes applied to hyperbolic problems such as these are typically more moderate, with $\tau\sim\mathcal{O}(h)$) rather than $\mathcal{O}(h^2)$ as for parabolic problems; many implicit time discretization schemes remain unconditionally stable.

Remark 1 (Multigrid for time-dependent problems)**.** The discussion above for these simple heat and wave equation problems in many ways apply to general time-dependent parabolic and hyperbolic problems, in the sense that the multigrid method for L_h, the discretization of an operator L, can be used for the time-step wise problems that arise from discretizations of equations of the form $u_t=-Lu$ and $u_{tt}=-Lu$. Often, implicit discretizations are preferred due to the fact that large time steps can be used; if the time steps are very small, then simply solving the problem with a relaxation method may be more effective [94]. Finally, we note that spatial and temporal coarsening may be combined, such that we may also discretize the spatial domain Ω into some discrete domain Ω_h, as discussed throughout most of this chapter.

Chapter 3

Convex optimization algorithms

3.1 Preliminaries

Here, we briefly introduce some key definitions and facts fundamental convex optimization definitions and facts that are relevant to the optimization algorithms and methods to which we apply our parallel-in-iteration (P-in-I) framework. Our main references for this chapter are [6, 7, 9, 75, 71]. For readers that are familiar with the basics of convex optimization and some classical algorithms, namely gradient descent and proximal gradient descent, this chapter can be treated as non-essential reading; those only interested in the optimization algorithms may skip to section 3.2 and section 3.3. In particular, we note that many of the definitions and results in this section are not directly referenced in this thesis, but are relevant for some of the ommitted proofs, as well as for a more general analysis of the optimization methods and problems of our interest.

Note that many optimization textbooks such as [7, 9] present definitions for vectors in a general Euclidean space $\mathbb{E}$. Since our goal with this chapter is simply to provide easily accessible definitions and facts pertaining to our practical applications, we take $\mathbb{E} = \mathbb{R}^N$, where $n \in \mathbb{N}$ is arbitrary. Thus, we take $\langle \cdot, \cdot \rangle$ to denote the standard Euclidean inner product (i.e., the dot product) and, unless noted otherwise, we let $\| \cdot \|$ denote the Euclidean norm, $\| \cdot \|_2 = \sqrt{\langle \cdot, \cdot \rangle}$.

3.1.1 Affine and convex sets

We begin by defining some special sets which appear in future definitions and facts.

Definition 8 (Affine subspace)**.** *Let S be a nonempty subset of $\mathbb{R}^N$. Then S is an affine subspace if*

$$(1-\lambda)\mathbf{u}+\lambda\mathbf{v}\in S,\quad \forall\mathbf{u}\in S,\ \forall\mathbf{v}\in S,\ \forall\lambda\in\mathbb{R}. \tag{3.1}$$

The smallest affine subspace containing S is the affine hull of S, written $\operatorname{aff} S$.

Definition 9 (Convex set)**.** *Let C be a (possibly empty) subset of $\mathbb{R}^N$. Then C is convex if*

$$(1-\lambda)\mathbf{u}+\lambda\mathbf{v}\in C,\quad \forall\mathbf{u}\in C,\ \forall\mathbf{v}\in C,\ \forall\lambda\in[0,1].$$

The smallest convex set containing C is the convex hull of C, is written $\operatorname{conv} C$. *In particular, for an arbitrary subset $C\subseteq\mathbb{R}^N$, $\operatorname{conv} C$ is equal to the set of all convex combinations of elements of C [7, Theorem 2.17].*

Definition 10 (Closed and open balls)**.** *Let $\mathbf{c}\in\mathbb{R}^N$, $r\geq 0$ and $\|\cdot\|$ be any norm. The closed ball with center $\mathbf{c}$ and radius r is given by*

$$B[\mathbf{c};r]=\{x\in X\mid\|x-\mathbf{c}\|\leq r\}, \tag{3.2}$$

while the open ball with same center and radius is given by

$$B(\mathbf{c};r)=\{x\in X\mid\|x-\mathbf{c}\|<r\}, \tag{3.3}$$

By [7, Example 2.3], the closed ball is a convex set; open balls are also convex sets. In particular, we use closed and open balls to define open and closed sets, which are generalizations of open and closed intervals, respectively, on n-dimensional spaces.

Definition 11 (Open and closed sets)**.** *Let $S\subseteq\mathbb{R}^N$. We say that S is open if*

$$\forall\mathbf{u}\in S,\quad \exists r>0 \text{ such that } B(\mathbf{u};r)\subseteq S. \tag{3.4}$$

Similarly, S is closed if

$$\forall\mathbf{u}\notin S,\quad \exists r>0 \text{ such that } B(\mathbf{u};r)\cap S=\emptyset; \tag{3.5}$$

this is equivalently to the complement of S, $\mathbb{R}^N\setminus S$, being an open set.

Definition 12 (Interior and relative interior)**.** *Let $S\subseteq\mathbb{R}^N$. The interior of S is given by*

$$\operatorname{int} S=\{\mathbf{u}\in S\mid(\exists\rho>0)\quad B[\mathbf{u};\rho]\subseteq S\}, \tag{3.6}$$

while the relative interior of S is given by

$$\operatorname{ri} S:=\{\mathbf{u}\in C\mid(\exists\rho>0)\quad \operatorname{aff}(S)\cap B[\mathbf{u};\rho]\subseteq S\}. \tag{3.7}$$

Note that $\operatorname{int} C\subseteq\operatorname{ri} S$, but the converse is false; e.g., for $C=\mathbb{R}\times\mathbf{0}\in\mathbb{R}^2$, $\operatorname{int} C=\emptyset$ but $\operatorname{ri} C=C$.

3.1.2 Proper, lower semicontinuous and convex functions

In this chapter, for the sake of generality we consider extended real-valued functions of the form $f : \mathbb{R}^N \to [-\infty, +\infty]$. That is, these functions attain values in $\mathbb{R} \cup \{-\infty, +\infty\}$; we refer to [6, 9] for general properties of extended real numbers and functions. However, in practice we mostly (i.e., with the exception of indicator functions) consider real-valued functions $f : \mathbb{R}^N \to \mathbb{R}$, such that we highlight this special case when relevant.

The optimization algorithms we consider in this thesis are proven to converge (given certain choices of parameters) to a minimizer of functions that are proper, convex and lower semicontinuous (lsc). We begin by defining proper functions.

Definition 13 (Proper function)**.** *Let $f : \mathbb{R}^N \to [-\infty, +\infty]$. The (effective) domain of f is*

$$\operatorname{dom} f := \{\mathbf{u} \in \mathbb{R}^N \mid f(\mathbf{u}) < +\infty\}; \tag{3.8}$$

f is called proper if $\operatorname{dom} f \neq \emptyset$ and $-\infty < f(\mathbf{u})$ for every $\mathbf{u} \in \mathbb{R}^N$.

In particular, any real-valued function $f : \mathbb{R}^N \to \mathbb{R}$ is proper. While in practice we mostly work with continuous functions, in the extended real numbers setting a weaker version of continuity, known as *lower semicontinuity*, is more generally considered. We define both lsc and continuous functions below.

Definition 14 (Lower semicontinuous function)**.** *Let $f : \mathbb{R}^N \to [-\infty, +\infty]$. Then f is lsc at $\mathbf{u} \in \mathbb{R}^N$ if, for every sequence $(\mathbf{u}_k)_{k\in\mathbb{N}}$ in $\mathbb{R}^N$, we have*

$$\lim_{k\in\mathbb{N}} \mathbf{u}_k = \mathbf{u} \implies f(\mathbf{u}) \leq \liminf_{k\in\mathbb{N}} f(\mathbf{u}_k). \tag{3.9}$$

If f is lsc for all $\mathbf{u} \in \mathbb{R}^N$, we simply say that f is lsc.

Definition 15 (Continuous function)**.** *Let $f : \mathbb{R}^N \to [-\infty, +\infty]$. Then f is continuous at $\mathbf{u} \in \mathbb{R}^N$ if, for every sequence $(\mathbf{u}_n)_{n\in\mathbb{N}}$ in $\mathbb{R}^N$, we have*

$$\lim_{n\in\mathbb{N}} \mathbf{u}_n = \mathbf{u} \implies \lim_{n\in\mathbb{N}} f(\mathbf{u}_n) = f(\mathbf{u}). \tag{3.10}$$

If f is continuous for all $\mathbf{u} \in \mathbb{R}^N$, we simply say that f is continuous.

Note that continuity implies lower-semicontinuity, but the converse is generally false. Finally, we define convex functions, for which many key convergence results and properties related to the minimization of these functions are well-established. While not all algorithms we present later in this chapter require that the objective function to be minimized be convex, in particular we only consider applying them to convex functions.

Definition 16 (Convex function)**.** *[7, Theorem 3.6] Let $f : \mathbb{R}^N \to (-\infty, +\infty]$. Then f is convex if and only if it satisfies Jensen's inequality:*

$$f((1-\lambda)\mathbf{u} + \lambda\mathbf{v}) \leq (1-\lambda)f(\mathbf{u}) + \lambda f(\mathbf{v}) \quad \forall \mathbf{u}, \mathbf{v} \in \operatorname{dom} f,\ \forall \lambda \in [0,1]. \tag{3.11}$$

If we have

$$f((1-\lambda)\mathbf{u} + \lambda\mathbf{v}) < (1-\lambda)f(\mathbf{u}) + \lambda f(\mathbf{v}) \quad \forall \mathbf{u}, \mathbf{v} \in \operatorname{dom} f,\ \mathbf{u} \neq \mathbf{v},\ \forall \lambda \in (0,1), \tag{3.12}$$

we say that f is strongly convex.

Note that when f is a convex, lsc and proper, real-valued scalar function, in practice we may replace lower-semicontinuity (definition 14) by continuity (definition 15).

Fact 1. *[7, Corollary 6.12] Let $f : \mathbb{R} \to (-\infty, +\infty]$ be convex, lower semicontinuous, and proper. Then $f|_{\operatorname{dom} f}$ (i.e., f with the domain $\mathbb{R}$ restricted to $\operatorname{dom} f$ (3.8)) is continuous.*

Finally, we note that positive multiples of convex and lsc functions are also convex and lsc; we implicitly use this fact many times throughout this thesis.

Fact 2 (Positive multiple rule for convex and lsc functions)**.** *[Proposition 4.5][7] Let $f : X \to [-\infty, +\infty]$, and let $\alpha > 0$. Then the following hold:*

(i) If f is convex, then so is αf.

(ii) If f is lsc, then so is αf.

3.1.3 Subgradients and derivatives

We recall the definition of the classical derivative of a function f.

Definition 17 (Classical derivative and gradient)**.** *Let $f : \mathbb{R}^N \to \mathbb{R}$. We say that f is differentiable at $\mathbf{u} \in \operatorname{int} \operatorname{dom} f$ if there exists $\mathbf{g} \in \mathbb{R}^N$ such that*

$$\lim_{\mathbf{h} \to 0} \frac{f(\mathbf{u}+\mathbf{h}) - f(\mathbf{u}) - \langle \mathbf{h}, \mathbf{g} \rangle}{\|\mathbf{h}\|}, \tag{3.13}$$

where $\langle \cdot, \cdot \rangle$ is the Euclidean inner product, in which case $\mathbf{g}$ is unique. We then say that $\mathbf{g} = \nabla f(\mathbf{u})$ is the gradient of f at $\mathbf{u}$.

In particular, when f is convex and differentiable, instead of referring to definition 16 for determining convexity it is often helpful to consider the following equivalent characterizations of convexity.

Fact 3. *[7, Theorem 11.10] Let f be proper on $\mathbb{R}^N$, suppose that $D := \operatorname{dom} f$ is convex and open, and that f is differentiable on D. Then the following are equivalent:*

(i) *f is convex (definition 16).*

(ii) $f(\mathbf{u}) + \langle \mathbf{v} - \mathbf{u}, \nabla f(\mathbf{u}) \rangle \leq f(\mathbf{v})\ \forall \mathbf{u} \in D,\ \forall \mathbf{v} \in D.$

(iii) $\langle \mathbf{u} - \mathbf{v}, \nabla f(\mathbf{u}) - \nabla f(\mathbf{v}) \rangle \geq 0\ \forall \mathbf{u} \in D,\ \forall \mathbf{v} \in D.$

(iv) *If f is twice differentiable on D, $\nabla^2 f(\mathbf{u}) \succeq 0\ \forall \mathbf{u} \in D$, where $\succeq$ denotes positive semi-definiteness.*

For proper functions, a more general notion known as *subdifferentiability* exists.

Definition 18 (Subdifferential and subgradients)**.** *Let $f : \mathbb{R}^N \to [-\infty, +\infty]$ be proper. The subdifferential of f, denoted by ∂f, is given by*

$$\partial f(\mathbf{u}) \coloneqq \{\mathbf{z} \in \mathbb{R}^N : \forall \mathbf{w} \in \mathbb{R}^N,\ f(\mathbf{u}) + \langle \mathbf{w} - \mathbf{u}, \mathbf{z} \rangle \leq f(\mathbf{w})\}. \tag{3.14}$$

If $\mathbf{z} \in \partial f(\mathbf{u})$, we call $\mathbf{z}$ a subgradient of f at $\mathbf{u}$.

The following rules regarding positive multiples and sums of subdifferentials are particularly useful.

Fact 4 (Positive-multiple rule for subdifferentials)**.** *[7, Proposition 13.1] Let f be proper on X, and let $\alpha > 0$. Then*

$$\partial(\alpha f)(x) = \alpha \partial f(x). \tag{3.15}$$

Fact 5 (Strong sum rule for subdifferentials)**.** *[6, Corolloary 16.48(ii)] Let f_1, f_2 be convex and proper on $\mathbb{R}^N$. Suppose that either one of the following is true:*

1. $\operatorname{ri} \operatorname{dom} f_1 \cap \operatorname{ri} \operatorname{dom} f_2 \neq \varnothing$,
2. $\operatorname{dom} f_1 \cap \operatorname{int} \operatorname{dom} f_2 \neq \varnothing$, *or*
3. $\operatorname{ri} \operatorname{dom} f_1 \cap \operatorname{dom} f_2 \neq \varnothing$ *and f_2 is polyhedral (i.e., the sum of the indicator function of the nonempty intersection of finitely many halfspaces and the finite maximum of affine functions).*

Then,

$$\partial\left(f_1 + f_2\right) = \partial f_1 + \partial f_2 \tag{3.16}$$

everywhere.

For convex and proper functions, subgradients and gradients are related as follows.

Fact 6 ([6], Proposition 17.31)**.** *Let f be convex and proper on $\mathbb{R}^N$, and suppose that $\mathbf{u} \in$ int dom f. Then f is differentiable at $\mathbf{u}$ if and only if $\partial f(\mathbf{u})$ is a singleton, such that $\partial f(\mathbf{u}) = \{\nabla f(\mathbf{u})\}$.*

Also related to and more general than classical derivatives are directional derivatives, which we define below.

Definition 19 (Directional derivative)**.** *Let f be proper on $\mathbb{R}^N$, suppose that $\mathbf{u} \in \operatorname{dom} f$, and let $\mathbf{d} \in \mathbb{R}^N$. Then the directional derivative of f at $\mathbf{u}$ in direction $\mathbf{d}$ is*

$$f'(\mathbf{u}; \mathbf{d}) := \lim_{\alpha \to 0^+} \frac{f(\mathbf{u} + \alpha\mathbf{d}) - f(\mathbf{u})}{\alpha}, \tag{3.17}$$

provided this limit exists in $[-\infty, +\infty]$.

Fact 7. *[7, Theorem 11.3] Let f be convex and proper on $\mathbb{R}^N$, suppose that $\mathbf{u} \in \operatorname{int}\operatorname{dom} f$, and let $\mathbf{d} \in \mathbb{R}^N$. Then $f'(\mathbf{u}; \mathbf{d})$ exists and*

$$f'(\mathbf{u}; \mathbf{d}) = \inf_{\alpha > 0} \frac{f(\mathbf{u} + \alpha\mathbf{d}) - f(\mathbf{u})}{\alpha}. \tag{3.18}$$

Moreover, $-\infty < -f'(\mathbf{u}; -d) \leq f'(\mathbf{u}; \mathbf{d}) < +\infty$.

Unlike the classical derivative, directional derivatives are more easily available for convex and proper functions, without requiring any (semi)continuity. Particularly important are the descent directions for which the directional derivative is negative; we call them *descent directions*.

Definition 20 (Descent direction)**.** *Let $f : X \to (-\infty, +\infty]$ be proper, let $\mathbf{u} \in \operatorname{dom} f$, and let $\mathbf{d} \in X \backslash \{0\}$. Then $\mathbf{d}$ is a descent direction of f at $\mathbf{u}$ if the directional derivative of f at $\mathbf{u}$ in direction $\mathbf{d}$ exists and is negative:*

$$f'(\mathbf{u}; \mathbf{d}) = \lim_{t \to 0^+} \frac{f(\mathbf{u} + t\mathbf{d}) - f(\mathbf{u})}{t} < 0. \tag{3.19}$$

Descent directions receive that name because moving from $\mathbf{u}$ in the direction of a descent direction $\mathbf{d}$ of f decreases the function value, as the following fact states.

Fact 8 (Descent property of descent directions)**.** *[8, Lemma 4.2] Let $f : \mathbb{R}^N \to (-\infty, \infty]$, $\mathbf{u} \in \operatorname{int} \operatorname{dom} f$, and assume that $\mathbf{0} \neq \mathbf{d} \in \mathbb{R}^N$ is a descent direction of f at $\mathbf{u}$. Then there exists $\varepsilon > 0$ such that $\mathbf{u} + t\mathbf{d} \in \operatorname{dom} f$ and*

$$f(\mathbf{u} + t\mathbf{d}) < f(\mathbf{u}) \tag{3.20}$$

for any $t \in (0, \varepsilon]$.

3.1.4 Lipschitz smoothness and strong convexity

We often consider stronger definitions of continuity and differentiability known as Lipschitz continuity and Lipschitz smoothness, respectively. Related to the former is the notion of nonexpansivity, a property which is exploited by the optimization algorithms of our interest.

Definition 21 (Lipschitz continuity)**.** *A mapping $T : D \to \mathbb{R}^N$, where $D \subseteq \mathbb{R}^N$, is Lipschitz continuous with constant $L \geq 0$, or simply L-Lipschitz, if*

$$\|T(\mathbf{u}) - T(\mathbf{v})\| \leq L\|\mathbf{u} - \mathbf{v}\| \quad \forall \mathbf{u}, \mathbf{v} \in D, \tag{3.21}$$

where $L \geq 0$. If $L = 1$, we say that T is nonexpansive (NE), while if $L < 1$, we say that T is a (Banach) contraction.

We say that a function is L-smooth if its gradient is L-Lipschitz, as per the following definition.

Definition 22 (Lipschitz smoothness)**.** *Let $f : \mathbb{R}^N \to (-\infty, +\infty]$ be proper. Assume that $D \subseteq \operatorname{dom} f$ is nonempty and open and that f is differentiable on D. Let $L \geq 0$. Then f is L-smooth on D if ∇f is L-Lipschitz on D, i.e.,*

$$\|\nabla f(\mathbf{u}) - \nabla f(\mathbf{v})\| \leq L\|\mathbf{u} - \mathbf{v}\| \quad \forall \mathbf{u} \in D,\ \mathbf{v} \in D. \tag{3.22}$$

Much like fact 3, for differentiable functions we may verify that f is L-smooth via simpler calculus characterizations rather than definition 22.

Fact 9. *[9, Theorem 5.8] Let $f : \mathbb{R}^N \to \mathbb{R}$ be convex and twice continuously differentiable, and let $L > 0$. Then, the following are equivalent:*

(i) f *is* L*-smooth.*

(ii) $\nabla^2 f(\mathbf{u}) \preceq LI \ \forall \mathbf{u} \in \mathbb{R}^N$.

(iii) $\lambda_{\max}\left(\nabla^2 f(\mathbf{u})\right) \leq L \ \forall \mathbf{u} \in \mathbb{R}^N$.

(iv) $\|\nabla^2 f(\mathbf{u})\| \leq L \ \forall \mathbf{u} \in \mathbb{R}^N$.

A useful property of L-smooth functions is that they are bounded from above by a quadratic functions, as the following fact states.

Fact 10 (Descent Lemma)**.** *[7, Lemma 23.6] Let* $f : \mathbb{R}^N \to (-\infty, +\infty]$ *be proper and assume that* $D \subseteq \operatorname{dom} f$ *is nonempty, open and convex, and that* f *is differentiable on* D*. Suppose that* f *is* L*-smooth on* D*. Then,*

$$f(\mathbf{v}) \leq f(\mathbf{u}) + \langle \mathbf{v} - \mathbf{u}, \nabla f(\mathbf{u}) \rangle + \frac{L}{2}\|\mathbf{u} - \mathbf{v}\|^2 \quad \forall \mathbf{u}, \mathbf{v} \in D. \tag{3.23}$$

Note that the Descent Lemma above can be rewritten, for some $\mathbf{u}, \mathbf{v} \in D$, as

$$f(\mathbf{v}) - f(\mathbf{u}) - \langle \mathbf{v} - \mathbf{u}, \nabla f(\mathbf{u}) \rangle \leq \frac{L}{2}\|\mathbf{u} - \mathbf{v}\|^2. \tag{3.24}$$

This left-hand side term receives the special name of *Bregman distance*, which we now define. Despite the name, it is more rigorously a divergence, since it is not symmetric.

Definition 23 (Bregman distance)**.** *Let* f *be proper on* $\mathbb{R}^N$*, and assume that* f *is differentiable on* $\operatorname{int} \operatorname{dom} f$*. Then*

$$D_f(\mathbf{u}, \mathbf{v}) := \begin{cases} f(\mathbf{u}) - f(\mathbf{v}) - \langle \mathbf{u} - \mathbf{v}, \nabla f(\mathbf{v}) \rangle & \text{if } \mathbf{v} \in \operatorname{int} \operatorname{dom} f; \\ +\infty & \text{if } \mathbf{v} \notin \operatorname{int} \operatorname{dom} f \end{cases} \tag{3.25}$$

is called the Bregman distance between $\mathbf{u}$ *and* $\mathbf{v}$*.*

3.1.5 Strong convexity

Besides strict convexity, which we defined in (3.12), we consider another refinement on the notion of convexity which we denote by *strong convexity*.

Definition 24 (Strongly convex function)**.** *Let f be proper on $\mathbb{R}^N$, and let $\beta > 0$. Then f is β-strongly convex if $\forall \mathbf{u} \in \mathbb{R}^N$, $\forall \mathbf{v} \in \mathbb{R}^N$, $\forall \lambda \in (0,1)$, we have*

$$f((1-\lambda)\mathbf{u} + \lambda\mathbf{v}) \leq (1-\lambda)f(\mathbf{u}) + \lambda f(\mathbf{v}) - \beta\frac{(1-\lambda)\lambda}{2}\|\mathbf{u}-\mathbf{v}\|^2. \tag{3.26}$$

In particular, strong convexity implies strict convexity [7, Proposition 24.2]. Once again, we provide useful characterizations of strong convexity which can be used in lieu of definition 24.

Fact 11. *[9, Theorem 5.24] Let f be convex, lower semicontinuous, and proper on $\mathbb{R}^N$, and let $\beta > 0$. Then the following are equivalent:*

(i) *f is β-strongly convex.*

(ii) $f(\mathbf{u}) + \langle \mathbf{z_u}, \mathbf{v}-\mathbf{u}\rangle \leq f(\mathbf{v}) - \frac{\beta}{2}\|\mathbf{u}-\mathbf{v}\|^2$ $\forall \mathbf{u} \in \operatorname{dom} f,\ \forall \mathbf{z_u} \in \partial f(\mathbf{u}),\ \forall \mathbf{v} \in \operatorname{dom} f.$

(iii) $\langle \mathbf{u}-\mathbf{v}, \mathbf{z_u}-\mathbf{z_v}\rangle \geq \beta\|\mathbf{u}-\mathbf{v}\|^2$ $\forall \mathbf{u},\mathbf{v} \in \operatorname{dom}\partial f,\ \forall \mathbf{z_u} \in \partial f(\mathbf{u}),\ \forall \mathbf{z_v} \in \partial f(\mathbf{v}).$

(iv) *$f - \beta\frac{1}{2}\|\cdot\|^2$ is convex.*

Fact 12. *[71, Theorem 2.1.11] Let f be twice continuously differentiable in $\mathbb{R}^N$. Then, f is β-strongly convex if and only if $\nabla^2 f(\mathbf{u}) \succeq \beta I$ for all $\mathbf{u} \in \mathbb{R}^N$.*

3.1.6 Minimizers

We recall that the problems of our interest are convex minimization problems. We recall the definition of the minimizer of a function below.

Definition 25. *Let $f : \mathbb{R}^N \to [-\infty, +\infty]$ be proper and let $\mathbf{u}^* \in \mathbb{R}^N$. Then*

(i) *$\mathbf{u}^*$ is a (global) minimizer of f if $f(\mathbf{u}^*) \leq f(\mathbf{u})$ $\forall \mathbf{u} \in \mathbb{R}^N$;*

(ii) *$\mathbf{u}^*$ is a local minimizer of f if $\exists \varepsilon > 0$ such that $\forall \mathbf{u} \in B[\mathbf{u}^*; \varepsilon]$, $f(\mathbf{u}^*) \leq f(\mathbf{u})$.*

The (possibly empty) set of global minimizers is $\operatorname{Argmin} f$*. If* $\operatorname{Argmin} f$ *is a singleton, we may identify* $\operatorname{Argmin} f$ *with its unique element* $\arg\min f$*.*

The following definition allows us to establish a key theorem on the existence of minimizers.

Definition 26 (Coercive functions)**.** *Let $f : \mathbb{R}^N \to [-\infty, +\infty]$. We say that f is coercive if*

$$\lim_{\|\mathbf{u}\| \to +\infty} f(\mathbf{u}) = +\infty, \tag{3.27}$$

and say that f is supercoercive if

$$\lim_{\|\mathbf{u}\| \to +\infty} \frac{f(\mathbf{u})}{\|\mathbf{u}\|} = +\infty. \tag{3.28}$$

Fact 13 (Key Existence Theorem)**.** *[7, Theorem 5.4] Let $f : \mathbb{R}^N \to (-\infty, +\infty]$ be coercive, lsc, and proper. Then f has a (global) minimizer, i.e., there exists $\mathbf{u}^* \in \mathbb{R}^N$ such that*

$$f(\mathbf{u}^*) = \min_{\mathbf{u} \in \mathbb{R}^N} f(\mathbf{u}). \tag{3.29}$$

The following fact, known as *Fermat's rule*, is the key fact we use for finding and verifying minimizers of proper functions.

Fact 14 (Fermat's rule)**.** *[7, Lemma 9.2] Let $f : \mathbb{R}^N \to (-\infty, +\infty]$ be convex, proper, and let $\mathbf{u} \in \mathbb{R}^N$. Then $0 \in \partial f(\mathbf{u}) \iff \mathbf{u}$ is a global minimizer of f.*

In the following, we address facts about minimizers of convex and proper functions. As the definition of convexity is strengthened from convex to strictly convex and strongly convex, the stronger the characterization of their minimizers becomes.

Fact 15. *[7, Proposition 5.9] Let $f : \mathbb{R}^N \to (-\infty, +\infty]$ be convex and proper. Then every local minimizer is also a global minimizer.*

Fact 16. *[7, Proposition 3.13] Let $f : \mathbb{R}^N \to \mathbb{R}^N$ be strictly convex and proper. Then f has either no or a unique minimizer.*

Fact 17. *[7, Theorem 24.7] Let f be β-strongly convex and lsc, where $\beta > 0$. Then the following hold:*

(i) f is supercoercive and hence coercive.

(ii) f has a unique minimizer $\mathbf{u}^*$.

(iii) $\forall \mathbf{u} \in \mathbb{R}^N$, $f(\mathbf{u}) - f(\mathbf{u}^*) \geq \frac{\beta}{2}\|\mathbf{u} - \mathbf{u}^*\|^2$, where $\mathbf{u}^*$ is as in (ii).

3.1.7 Nonexpansive mappings and monotone operator theory

We recall from definition 22 that an operator (or mapping) is called nonexpansive (NE) if it is L-Lipschitz with constant $L \leq 1$. We now introduce the stronger notion of *firm nonexpansivity*.

Definition 27 (firmly nonexpansive (FNE) operator)**.** *Let $T : D \to \mathbb{R}^N$ with $D \subseteq \mathbb{R}^N$. Then T is FNE if*

$$\|T\mathbf{u} - T\mathbf{v}\|^2 + \|(I-T)\mathbf{u} - (I-T)\mathbf{v}\|^2 \leq \|\mathbf{u} - \mathbf{v}\|^2 \quad \forall \mathbf{u}, \mathbf{v} \in D. \tag{3.30}$$

We provide useful characterizations of FNE operators below.

Fact 18. *[7, Theorem 22.3] Let $T : D \to \mathbb{R}^N$. Then the following are equivalent:*

(i) *T is FNE.*

(ii) *$I - T$ is FNE.*

(iii) *$\|T\mathbf{u} - T\mathbf{v}\|^2 \leq \langle \mathbf{u} - \mathbf{v}, T\mathbf{u} - T\mathbf{v} \rangle \quad \forall \mathbf{u}, \mathbf{v} \in D.$*

(iv) *$2T - I$ is NE.*

The following facts discuss the (firm) nonexpansivity of the some important gradient-based operators.

Fact 19 (Baillon-Haddad Theorem)**.** *[7, Corollary 23.9] Let $f : \mathbb{R}^N \to \mathbb{R}$ be convex and differentiable. Then ∇f is NE if and only if ∇f is FNE.*

Fact 20 (Baillon-Haddad Corollary)**.** *[7, Corollary 23.10] Let f be convex and L-smooth on $\mathbb{R}^N$ with respect to the Euclidean norm. Then both $\frac{1}{L}\nabla f$ and $I - \frac{1}{L}\nabla f$ are FNE.*

Related to NE and FNE operators are *averaged operators*, which are mappings that can be written as an average of the identity operator I and one NE operator.

Definition 28 (Averaged mappings)**.** *Let $T : \mathbb{R}^N \to \mathbb{R}^N$, and let $\alpha \in (0, 1)$. Then T is α-averaged if there exists $N : \mathbb{R}^N \to \mathbb{R}^N$ such that N is NE and $T = (1-\alpha)I + \alpha N$.*

From the above definition, we can interpret α as a parameter measuring how close the operator T is to the identity. The following characterization allows to verify if an operator is averaged without having to explicitly determine a NE operator N, as in definition 28.

Fact 21. *[7, Theorem 22.11] Let $T : \mathbb{R}^N \to \mathbb{R}^N$ be NE, and let $\alpha \in (0, 1)$. Then T is α-averaged if and only if for all $\mathbf{u}, \mathbf{v} \in \mathbb{R}^N$, we have*

$$\|T\mathbf{u} - T\mathbf{v}\|^2 \leq \|\mathbf{u} - \mathbf{v}\|^2 - \frac{1-\alpha}{\alpha}\|(I - T)\mathbf{u} - (I - T)\mathbf{v}\|^2. \tag{3.31}$$

In particular, by comparing (3.31) to (3.30), we have that an operator is $\frac{1}{2}$-averaged if and only if it is FNE. Furthermore, if T is α-averaged, it also β-averaged for all $\alpha < \beta < 1$. While the composition of FNE operators is generally not FNE, it is averaged, as per the following fact.

Fact 22. *[7, Corollary 22.13] Let T_1 and T_2 be FNE on $\mathbb{R}^N$. Then T_2T_1 is $\frac{2}{3}$-averaged.*

Now, we consider sequences generated by an operator T, of the form $\mathbf{u}_{k+1} = T\mathbf{u}_k$ for $k = 0, 1, \ldots$. We introduce a notion of monotonicity for the sequence $(\mathbf{u}_k)_{k=0,1,\ldots}$ rather than the operator T, known as *Fejér monotonicity*.

Definition 29 (Fejér monotonicity)**.** *A sequence $(\mathbf{u}_k)_{k=0,1,\ldots}$ in $\mathbb{R}^N$ is Fejér-monotone with respect to a nonempty subset C of $\mathbb{R}^N$ if*

$$\|\mathbf{u}_{k+1} - \mathbf{c}\| \leq \|\mathbf{u}_k - \mathbf{c}\| \quad \forall \mathbf{c} \in C,\ \forall k = 0, 1, \ldots. \tag{3.32}$$

Hence, if a Fejér-monotone sequence converges to a set C, then each iteration brings the sequence closer (or at least as close) to all points in C, such that the distance between latest iterate $\mathbf{u}_k$ and the set C never increases. In particular, for an averaged operator T with nonempty set of fixed points $\operatorname{Fix} T$, we have that the sequences generated by T converges to a fixed point of T.

Fact 23. *[7, Corollary 22.20] Let $T : \mathbb{R}^N \to \mathbb{R}^N$ be α-averaged with $\operatorname{Fix} T \neq \varnothing$. Then $\left(T^k\mathbf{u}_0\right)_{k\in\mathbb{N}}$ is Fejér monotone with respect to $\operatorname{Fix} T$, and it converges to some point in $\operatorname{Fix} T$.*

Also related to NE and FNE operators are *(maximal) monotone operators*, which are typically discussed in the context of minimizing sums of functions [25, 63]. First, we define what we mean by the *resolvent* of an operator.

Definition 30. *Let $T : D \to \mathbb{R}^N$, $D \subseteq \mathbb{R}^N$. The resolvent of an operator T, with parameter $c > 0$, is given by $(I + cT)^{-1}$.*

Note that even if T is possibly multi-valued (i.e., it maps a point to a set), the resolvent $(I + cT)^{-1}$ is always single-valued. We now define monotone operators and maximal monotone operators, the latter being related to a stronger notion of monotonicity.

Definition 31 ((Maximal) monotone operator)**.** *Let $T : D \to \mathbb{R}^N$, where T is possibly multi-valued. Then T is monotone if*

$$(\mathbf{v} - \mathbf{z}, \mathbf{u} - \mathbf{w}) \geq 0, \quad \forall \mathbf{u} \in D, \mathbf{v} \in T(\mathbf{u}), \mathbf{w} \in D, \mathbf{z} \in T(\mathbf{w}). \tag{3.33}$$

If T is monotone, it is said to be maximal monotone if the resolvent $(I + cT)^{-1}$ is a contraction (i.e., L-Lipschitz with $L < 1$) defined on $\mathbb{R}^N$ for any $c > 0$.

The following fact characterizes (maximal) monotonicity in relation to the firm nonexpansiveness of the resolvent.

Fact 24. *[25, Theorem 2] T is monotone if, and only if, its resolvent $(I + cT)^{-1}$ is FNE for any $c > 0$. Furthermore, T is maximal monotone if, and only if, $(I + cT)^{-1}$ is FNE and $\operatorname{dom}\left((I + cT)^{-1}\right) = \mathbb{R}^N$ for any $c > 0$.*

3.1.8 Proximal operators

Related to minimizers of a function is the *proximal operator*, which maps a vector $\mathbf{v} \in \mathbb{R}^N$ to a set of vectors that balance having a small function value with being close to $\mathbf{v}$. We define the proximal operator (or proximal mapping, or simply proximal) in the following.

Definition 32 (Proximal operator)**.** *Let $f : \mathbb{R}^N \to (-\infty, +\infty]$ and let $\mathbf{v} \in \mathbb{R}^N$. Then, the proximal operator of f, $\mathcal{P}_f : \mathbb{R}^N \to \mathbb{R}^N$, at $\mathbf{v}$ is given by*

$$\mathcal{P}_f(\mathbf{v}) = \left\{ \mathbf{u} \in \mathbb{R}^N : \mathbf{u} = \operatorname*{arg\,min}_{\mathbf{z} \in \mathbb{R}^N} \left(f(\mathbf{z}) + \frac{1}{2}\|\mathbf{z} - \mathbf{v}\|^2 \right) \right\}. \tag{3.34}$$

If $\mathbf{u} \in \mathcal{P}_f(\mathbf{v})$, we say that $\mathbf{u}$ is a proximal point of $\mathbf{v}$.

Note that if f is convex, lsc and proper, $\left(f(\cdot) + \frac{1}{2}\|\cdot - \mathbf{v}\|^2\right)(\mathbf{z})$ is strongly convex by fact 11, such that $\mathcal{P}_f(\mathbf{v})$ is single-valued, such that (3.34) becomes

$$\mathcal{P}_f(\mathbf{v}) = \operatorname*{arg\,min}_{\mathbf{z} \in \mathbb{R}^N} \left(f(\mathbf{z}) + \frac{1}{2}\|\mathbf{z} - \mathbf{v}\|^2 \right). \tag{3.35}$$

Then, instead of saying that $\mathbf{u} \in \mathcal{P}_f(\mathbf{v})$, we simply say that $\mathbf{u} = \mathcal{P}_f(\mathbf{v})$. For such functions, we also have the following useful characterizations:

Fact 25. *[7, Theorem 25.3] Let f be convex, lsc and proper on $\mathbb{R}^N$. Let $\mathbf{u} \in \operatorname{dom} f$ and $\mathbf{v} \in \mathbb{R}^N$. Then, the following are equivalent:*

(i) $\mathbf{u} = \mathcal{P}_f(\mathbf{v})$;

(ii) $\mathbf{v} - \mathbf{u} \in \partial f(\mathbf{u})$;

(iii) *for all* $\mathbf{z} \in \mathbb{R}^N$, $\langle \mathbf{z} - \mathbf{u}, \mathbf{v} - \mathbf{u} \rangle \leq f(\mathbf{z}) - f(\mathbf{u})$.

In particular, by fact 25(ii) we have that

$$\mathbf{u} = \mathcal{P}_f(\mathbf{v}) \iff \mathbf{v} = (I + \partial f)\mathbf{u} \iff (I + \partial f)^{-1}\mathbf{v} = \mathbf{u}. \tag{3.36}$$

Therefore, the proximal operator of convex, lsc and proper functions can be characterized as the resolvent of the subdifferential operator:

$$\mathcal{P}_f = (I + \partial f)^{-1}. \tag{3.37}$$

Hence, we use definitions of the unique proximal operator in (3.35) and (3.37) interchangeably, depending on which perspective best suits our analysis at a given moment. The following fact characterizes the connection between the proximal and the minimizer of a function,

Fact 26. *[7, Theorem 25.9] Let f be convex, lsc, and proper on $\mathbb{R}^N$, and let $\mathbf{v} \in \mathbb{R}^N$. Then*

$$f\left(\mathcal{P}_f(\mathbf{v})\right) \leq f(\mathbf{v}) - \frac{1}{2}\left\|\mathbf{v} - \mathcal{P}_f(\mathbf{v})\right\|^2 \tag{3.38}$$

and

$$\mathbf{v} = \mathcal{P}_f(\mathbf{v}) \iff \mathbf{v} \in \operatorname{Arg\,min} f. \tag{3.39}$$

Furthermore, the proximal operator is FNE by fact 27, which we precede with a necessary definition.

Definition 33 (Fenchel conjugate)**.** *Let $f : \mathbb{R}^N \to [-\infty, +\infty]$. Then, the Fenchel conjugate of f is given by*

$$f^*(\mathbf{v}) = \arg\sup_{\mathbf{u} \in \mathbb{R}^N} \left(\langle \mathbf{u}, \mathbf{v} \rangle - f(\mathbf{u})\right). \tag{3.40}$$

Fact 27. *[7, Theorem 26.1] Let f be convex, lsc, and proper on X. Then both $\mathcal{P}_f$ and $I - \mathcal{P}_f$ are FNE, and $\mathcal{P}_f + \mathcal{P}_{f^*} = I$, where f^* denotes the Fenchel conjugate of f (3.40).*

In many cases, we evaluate $\mathcal{P}_{sf}$, where $s > 0$ is some scalar. Note that for any $s > 0$, if $f(\mathbf{u}) \leq f(\mathbf{v})$, then $sf(\mathbf{u}) \leq sf(\mathbf{v})$. Therefore,

$$\mathbf{u}^* = \arg\min_{\mathbf{u} \in \mathbb{R}^N} sf(\mathbf{u}) \iff \mathbf{u}^* = \arg\min_{\mathbf{u} \in \mathbb{R}^N} f(\mathbf{u}), \tag{3.41}$$

so $\operatorname{Arg\,min} f = \operatorname{Arg\,min} sf$. We discuss how multiplying f by a scalar s affects the intepretation of the proximal operator in the following remark.

Remark 2 (Proximal of sf)**.** Let $s > 0$ be some scalar in $\mathbb{R}$. Then, by (3.35) we have that

$$\begin{aligned} \mathbf{u} = \mathcal{P}_{sf}(\mathbf{v}) &\iff \mathbf{u} = \underset{\mathbf{z}\in\mathbb{R}^N}{\operatorname{argmin}} \left(sf(\mathbf{z}) + \frac{1}{2}\left\|\mathbf{z}-\mathbf{v}\right\|^2 \right) \\ &\iff \mathbf{u} = \underset{\mathbf{z}\in\mathbb{R}^N}{\operatorname{argmin}} \left(f(\mathbf{z}) + \frac{1}{2s}\left\|\mathbf{z}-\mathbf{v}\right\|^2 \right), \end{aligned} \tag{3.42}$$

such that s can be interpreted as a relative weight term for the trade-off between minimizing f and remaining close to $\mathbf{v}$ [75]. Similarly, (3.37) and fact 4 we have that

$$\mathcal{P}_{sf} = (I + \partial(sf))^{-1} = (I + s\partial f)^{-1}, \tag{3.43}$$

such that s can be interpreted as a weight term for the subdifferential of f. Finally, by (3.39), we have that

$$\mathbf{v} = \mathcal{P}_f(\mathbf{v}) \iff \mathbf{v} \in \operatorname{Arg\,min} f \iff \mathbf{v} \in \operatorname{Arg\,min} sf \iff \mathbf{v} = \mathcal{P}_{sf}(\mathbf{v}). \tag{3.44}$$

However, $\mathcal{P}_f(\mathbf{v}) \neq \mathcal{P}_{sf}(\mathbf{v})$ in general.

Related to the proximal operator is the *(Moreau) envelope* of a function, which we now define.

Definition 34 (Moreau envelope)**.** *Let f be convex, lsc, and proper on $\mathbb{R}^N$. Then the (Moreau) envelope, with smoothing parameter $\mu > 0$, is the full-domain continuous convex function $\operatorname{env}_\mu f$ defined by*

$$\operatorname{env}_\mu(f(\mathbf{v})) := \min_{\mathbf{u}\in\mathbb{R}^N} \left(f(\mathbf{u}) + \frac{1}{2\mu}\|\mathbf{u}-\mathbf{v}\|^2 \right); \tag{3.45}$$

if $\mu = 1$, we simply write $\operatorname{env}(f)$ instead of $\operatorname{env}_1(f)$.

The function minimized in the definition of the envelope is the same as that in the definition of $\mathcal{P}_{\mu f}$, by remark 2. We establish this connection in the following fact.

Fact 28. *[7, Theorem 27.2] Let f be convex, lsc, and proper on $\mathbb{R}^N$. Then,*

$$\operatorname{env}_\mu(f(\mathbf{u})) := \min_{\mathbf{u}\in\mathbb{R}^N} \left(f\left(\mathcal{P}_{\mu f}(\mathbf{u})\right) + \frac{1}{2\mu}\left\|\mathbf{u} - \mathcal{P}_{\mu f}(\mathbf{u})\right\|^2 \right) \quad \forall \mathbf{u} \in \mathbb{R}^N. \tag{3.46}$$

Furthermore, the envelope of f is L-smooth even if f is not.

Fact 29. *[7, Theorem 27.2] The envelope $\operatorname{env}_\mu f$ is $(1/\mu)$-smooth and its gradient*

$$\nabla \operatorname{env}_\mu f(\mathbf{v}) = \frac{1}{\mu}\left(\mathbf{v} - \mathcal{P}_{\mu f}(\mathbf{v})\right) \tag{3.47}$$

is therefore $(1/\mu)$-Lipschitz.

3.2 Optimization algorithms for minimizing smooth functions

Consider the problem of minimizing a real-valued (hence, proper) convex and L-smooth function f,

$$\min_{\mathbf{u}\in\mathbb{R}^N} f(\mathbf{u}). \tag{3.48}$$

While there are multiple methods that can be considered for solving (3.48), we focus on two simple, classical methods: Gradient Descent (GD) and Proximal Point (PP).

3.2.1 Gradient descent (GD)

Let $f : \mathbb{R}^N \to \mathbb{R}$ be a convex and L-smooth function. Given a starting point $\mathbf{u}_0 \in \mathbb{R}^N$ and a fixed stepsize $s > 0$, the GD iteration is given by

$$\mathbf{u}_{j+1} = \mathbf{u}_j - s\nabla f(\mathbf{u}_j). \tag{3.49}$$

Equivalently, defining the GD operator $\mathcal{G}_{sf}(\mathbf{u}_k) \coloneqq (I - s\nabla f)(\cdot)$, we can write the iteration above as

$$\mathbf{u}_{j+1} = (I - s\nabla f)(\mathbf{u}_j) =: \mathcal{G}_{sf}(\mathbf{u}_j), \quad j = 0, 1, \dots. \tag{3.50}$$

Note that if f is differentiable at $\mathbf{u}$ and $\nabla f(\mathbf{u}) \neq 0$, then $-\nabla f(\mathbf{u})$ is a descent direction (3.19):

$$f'(\mathbf{u}; -\nabla f(\mathbf{u})) = \langle -\nabla f(\mathbf{u}), \nabla f(\mathbf{u})\rangle = -\|\nabla f(\mathbf{u})\|^2 < 0. \tag{3.51}$$

Hence, by fact 8, we have that $f(\mathbf{u} - s\nabla f(\mathbf{u})) < f(\mathbf{u})$ for some sufficiently small s. Therefore, by (3.49), $f(\mathbf{u}_{j+1}) < f(\mathbf{u}_j)$ for all $j = 0, 1, \dots$ if s is sufficiently small. To further justify why we may only take small steps of size s, note that GD is an explicit scheme since $\mathbf{u}_{j+1}$ only depends on information available at the previous iteration j. Therefore, it must have a Courant-Friedrichs-Lewy (CFL) limit on the stepsize s, which in advance we state to be $s < 2/L$.

In order to provide more intuition on how and why the GD method works and prove basic convergence properties, we now consider the *majorization-minimization* interpretation of GD for L-smooth functions; later, we introduce the additional hypothesis of (strong) convexity of f.

Gradient descent as a majorization-minimization method for smooth functions

Assume that f is an L-smooth function and consider a Taylor approximation of $f(\mathbf{u})$ in the neighbourhood of $\mathbf{u} \in \mathbb{R}^N$. That is, for some $\mathbf{u}_j \in \mathbb{R}^N$, we have

$$f(\mathbf{u}) = f(\mathbf{u}_j) + \nabla f(\mathbf{u}_j)^T(\mathbf{u} - \mathbf{u}_j) + \mathcal{O}\left((\mathbf{u} - \mathbf{u}_j)^2\right). \tag{3.52}$$

Suppose $\mathbf{u}_j \approx \mathbf{u}$, where $\mathbf{u}_j$ is the current approximation of $\mathbf{u}$. Then, we can approximate $f(\mathbf{u})$ by the truncated first-order Taylor approximation

$$f(\mathbf{u}) \approx f(\mathbf{u}_j) + \nabla f(\mathbf{u}_j)^T(\mathbf{u} - \mathbf{u}_j). \tag{3.53}$$

By minimizing the right-hand side approximation above with respect to $\mathbf{u}$, one could obtain an improved approximation $\mathbf{u}_{j+1}$. However, $f(\mathbf{u}_j) + \nabla f(\mathbf{u}_j)^T(\mathbf{u} - \mathbf{u}_j)$ does not generally possess a minimum since $\mathbf{u} - \mathbf{u}_j$ is unbounded. Therefore, in order to minimize (3.53), we need to enforce some penalty on the distance between the current approximation $\mathbf{u}_j$ and the approximation target $\mathbf{u}$; moreover, the accuracy of the approximation in (3.53) itself depends on $\mathbf{u}_j$ being sufficiently close to $\mathbf{u}$. Thus, we consider replacing the right-hand side of (3.53) by a L_2-regularized approximation function:

$$\widehat{f}(\mathbf{u}, \mathbf{u}_j) \coloneqq f(\mathbf{u}_j) + \nabla f(\mathbf{u}_j)^T(\mathbf{u} - \mathbf{u}_j) + \frac{1}{2s}\|\mathbf{u} - \mathbf{u}_j\|_2, \tag{3.54}$$

where $s > 0$ is a pre-defined penalty term on the Euclidean distance between $\mathbf{u}$ and $\mathbf{u}_j$.

The operator $\widehat{f}(\mathbf{u}, \mathbf{u}_j)$ in (3.54) is related to an iterative method known as Majorization-Minimization (MM) algorithm for minimizing smooth functions. The MM iteration for minimizing an L-smooth function $f : \mathbb{R}^N \to \mathbb{R}$, starting from some $\mathbf{u}_0 \in \mathbb{R}^N$, is given by

$$\mathbf{u}_{j+1} = \arg\min_{\mathbf{u}} \widehat{f}(\mathbf{u}, \mathbf{u}_j), \quad j = 0, 1, \ldots \tag{3.55}$$

where $\widehat{f}(\mathbf{u}, \mathbf{u}_j)$ is defined in (3.54). Taking the derivative of (3.54) with respect to $\mathbf{u}$ yields

$$\frac{\partial}{\partial \mathbf{u}} \widehat{f}(\mathbf{u}, \mathbf{u}_j) = \nabla f(\mathbf{u}_j) + \frac{1}{s}(\mathbf{u} - \mathbf{u}_j). \tag{3.56}$$

Taking a second derivative with respect to $\mathbf{u}$ yields

$$\frac{\partial^2}{\partial \mathbf{u}^2} \widehat{f}(\mathbf{u}, \mathbf{u}_j) = \frac{1}{s}, \tag{3.57}$$

such that $\widehat{f}(\mathbf{u}, \mathbf{u}_j)$ is convex by fact 3. Hence, to obtain a minimizer, by fact 14 we set the derivative equal to zero:

$$\frac{\partial}{\partial \mathbf{u}} \widehat{f}(\mathbf{u}, \mathbf{u}_j) = \nabla f(\mathbf{u}_j) + \frac{1}{s}(\mathbf{u} - \mathbf{u}_j) = 0. \tag{3.58}$$

Solving this for $\mathbf{u}$ yields

$$\mathbf{u} = \mathbf{u}_j - s\nabla f(\mathbf{u}_j). \tag{3.59}$$

Interpreting $\mathbf{u}$ in this update as an improved approximation $\mathbf{u}_{j+1}$ for the exact minimizer $\mathbf{u}^*$ of $\widehat{f}(\mathbf{u}, \mathbf{u}_j)$, we recover the GD update from (3.49).

Having established the equivalence between GD and MM algorithms, we now state that convergence to a minimizer of an L-smooth function f is attained when $s < 2/L$.

Theorem 1. *[71, Theorem 2.1.14] Let f be an L-smooth function, and let $0 < s < 2/L$. Then, the GD sequence generated by (3.49) converges monotonically in the function value sequence $\{f(\mathbf{u}_j)\}_{t\in\mathbb{N}}$ to a minimizer of (3.48).*

In the following remark, we prove a part of theorem 1 to clearly establish the previously mentioned CFL (or stability) limit of $s < 2/L$ for the (explicit) GD scheme.

Remark 3 (CFL limit of GD)**.** By (3.49), the GD update implies $\mathbf{u}_{j+1} - \mathbf{u}_j = s\nabla f(\mathbf{u}_j)$. Substituting this into the Descent Lemma, fact 10, yields

$$f(\mathbf{u}_{j+1}) \leq f(\mathbf{u}_j) + \langle -s\nabla f(\mathbf{u}_j), \nabla f(\mathbf{u}_j)\rangle + \frac{L}{2}\left\|-s\nabla f(\mathbf{u}_j)\right\|_2^2. \tag{3.60}$$

By the definition of the Euclidean norm, $\langle \nabla f(\mathbf{u}_j), \nabla f(\mathbf{u}_j)\rangle = \|\nabla f(\mathbf{u}_j)\|^2$, such that

$$\begin{aligned} f(\mathbf{u}_{j+1}) &\leq f(\mathbf{u}_j) - s\|\nabla f(\mathbf{u}_j)\|^2 + \frac{L}{2}s^2\left\|\nabla f(\mathbf{u}_j)\right\|^2 \\ &= f(\mathbf{u}_j) - s\left(1 - s\frac{L}{2}\right)\left\|\nabla f(\mathbf{u}_j)\right\|^2. \end{aligned} \tag{3.61}$$

It follows that the difference of the function f evaluated at consecutive iterates $\mathbf{u}_j$ and $\mathbf{u}_{j+1}$ is given by $f(\mathbf{u}_{j+1}) - f(\mathbf{u}_j) = s\left(1 - \frac{L}{2}\right)\left\|\nabla f(\mathbf{u}_j)\right\|^2$, where $\left\|\nabla f(\mathbf{u}_j)\right\|^2$ is non-negative by the positive-definedness of the norm. Hence, the $\{f(\mathbf{u}_j)\}_{t\in\mathbb{N}}$ sequence converges monotonically, i.e., $f(\mathbf{u}_{j+1}) < f(\mathbf{u}_j)$ for all t, if

$$s\left(1 - s\frac{L}{2}\right) > 0 \implies 1 > s\frac{L}{2} \implies s < \frac{2}{L}, \tag{3.62}$$

which proves the CFL limit of GD.

While convergence holds for $0 < s < 2/L$, $s = 1/L$ is the optimal fixed stepsize for gradient descent, at least in one particular sense. We paraphrase the following lemma from [71, Section 1.2.3], where the result is proven as an unlabeled remark.

Lemma 1 (Optimal GD stepsize)**.** *The optimal GD stepsize for (3.49), in the sense of minimizing the upper bound for the decrease of the objective function, is given by $s^* = 1/L$, which satisfies*

$$f(\mathbf{u}_{j+1}) \leq f(\mathbf{u}_j) - \frac{1}{2L} \|\nabla f(\mathbf{u}_j)\|^2. \tag{3.63}$$

Proof. Recall (3.61):

$$f(\mathbf{u}_{j+1}) \leq f(\mathbf{u}_j) - s\left(1 - s\frac{L}{2}\right) \|\nabla f(\mathbf{u}_j)\|^2 \tag{3.64}$$

Note that in order to minimize this upper bound, we must find

$$s^* \coloneqq \arg\min_s h(s), \qquad \text{where} \qquad h(s) \coloneqq -s\left(1 - s\frac{L}{2}\right). \tag{3.65}$$

Taking the derivative and setting it equal to zero yields

$$sL - 1 = 0 \implies s = \frac{1}{L}. \tag{3.66}$$

Taking a second derivative with respect to s yields $L > 0$, such that the $-s(1 - s\frac{L}{2})$ is convex by fact 3 and thus $s^* = 1/L$ is indeed the minimizer of $h(s)$. Finally, (3.63) follows directly from plugging $s = 1/L$ into (3.61). □

Gradient descent for convex, smooth functions

Now, assume that $f : \mathbb{R}^N \to \mathbb{R}$ is convex and L-smooth, thus fully returning to the setting of (3.48). From now on, we restrict our analysis to $s = 1/L$, for which many special convergence results can be proven; note that many of the proofs for the following facts hold for any $0 < s \leq 1/L$, although there is usually no reason to consider $s < 1/L$. We begin by characterizing this $s = 1/L$ monotone convergence as specifically being of the Fejér-monotone type (definition 29).

Fact 30 (GD Fejér-monotone convergence)**.** *[7, Theorem 29.4] Let $f : \mathbb{R}^N \to \mathbb{R}$ be a convex L-smooth function. Let $s = 1/L$. Then, the GD sequence defined by (3.49) converges Fejér-monotonically (see definition 29) to a minimizer of f.*

We are also able to find an upper bound for the GD convergence in the function value with respect to the distance between the initial guess, $\mathbf{u}_0$, and a minimizer of f, $\mathbf{u}^*$.

Fact 31 (GD convergence in the function value)**.** *[7, Theorem 29.4] Let $f : \mathbb{R}^N \to \mathbb{R}$ be a convex and L-smooth function and let $s = 1/L$. Then the function evaluated at the kth point generated by (3.49) satisfies*

$$f(\mathbf{u}_k) - f(\mathbf{u}^*) \leq \frac{L}{2k}\|\mathbf{u}_0 - \mathbf{u}^*\|^2, \tag{3.67}$$

where $\mathbf{u}^*$ is a minimizer of f.

While $s = 1/L$ does not ensure monotone convergence in the gradient norm, we can provide an upper bound for the smallest gradient attained after a fixed number of iterations (although in principle we do not have any information as to which iterate has the smallest gradient).

Fact 32 (GD convergence in gradient norm)**.** *[9, Theorem 10.26] Let $f : \mathbb{R}^N \to \mathbb{R}$ be L-smooth and let $s = 1/L$. Then*

$$\min_{0 \leq j \leq k-1} \|\nabla f(\mathbf{u}_j)\|^2 \leq \sqrt{\frac{2}{sk}\left(f(\mathbf{u}_0) - f(\mathbf{u}^*)\right)}. \tag{3.68}$$

By fact 31, we say that the GD sequence with $s = 1/L$ converges to a minimizer of (3.48) with a sublinear convergence rate of $\mathcal{O}(1/k)$. Unfortunately, convergence in the iterates can be arbitrarily slow, as the following fact establishes.

Fact 33 (GD convergence in the iterates)**.** *[71, Theorem 2.1.7] For any k, $1 \leq k \leq \frac{1}{2}(N-1)$ and any $\mathbf{u}_0 \in \mathbb{R}^N$, the GD sequence generated by (3.49) satisfies*

$$\|\mathbf{u}_k - \mathbf{u}_*\| \geq \frac{1}{8}\|\mathbf{u}_0 - \mathbf{u}^*\|. \tag{3.69}$$

Gradient descent for strongly convex, smooth functions

In many practical situations, our objective function is L-smooth and strongly convex (definition 24). if an L-smooth function f satisfies the (3.70) (e.g., if f is strongly convex), the GD sequence (3.49) converges linearly in the function value; unlike standard convexity, strong convexity ensures fast convergence in the iterates as well.

Fact 34. *[9, Theorem 10.29] The GD sequence (3.49) for a L-smooth and μ-strongly convex f satisfies*

(a) $\|\mathbf{u}_{k+1} - \mathbf{u}^*\|^2 \leq \left(1 - \frac{\mu}{L}\right) \|\mathbf{u}_k - \mathbf{u}^*\|^2$;

(b) $\|\mathbf{u}_k - \mathbf{u}^*\|^2 \leq \left(1 - \frac{\mu}{L}\right)^k \|\mathbf{u}_0 - \mathbf{u}^*\|^2$;

(c) $F(\mathbf{u}_{k+1}) - F^* \leq \frac{L}{2} \left(1 - \frac{\mu}{L}\right)^{k+1} \|\mathbf{u}_0 - \mathbf{u}^*\|^2$.

By fact 34, we say that GD converges to the minimizer of f with a linear convergence rate $\mathcal{O}(1 - \mu/L)$. Note that many of the aforementioned results hold for a more general assumption than strong convexity, which is the Polyak-Lojasiewicz (PL) condition that we now define.

Definition 35. *A lower-bounded function $f : \mathbb{R}^N \to \mathbb{R}$ satisfies the PL condition [77] if there exists $\beta > 0$ such that*

$$\frac{1}{2}\|\nabla f(\mathbf{u})\|^2 \geq \beta\,(f(\mathbf{u}) - f_\star) \quad \forall \mathbf{u} \in \mathbb{R}^N, \tag{3.70}$$

where $f_\star := \inf f$.

In particular, strong convexity implies the PL condition is satisfied [56, Theorem 2]. We refer to [56] for proofs on the linear convergence of Proximal Gradient Descent (PG) for L-smooth functions satisfying the PL condition.

3.2.2 Proximal point (PP)

We recall that GD is an explicit method with CFL condition $s < 2/L$. Therefore, in order to take steps of size $s \geq 2/L$, we must consider an implicit method instead. The method we consider is the PP algorithm, which we now define. Given some starting point $\mathbf{u}_0$ and a positive scalar $s > 0$, the PP algorithm update for a continuous and convex f is given by

$$\mathbf{u}_{j+1} = \mathbf{u}_j - s\nabla f(\mathbf{u}_{j+1}), \quad j = 0, 1, \ldots. \tag{3.71}$$

We note that the iteration (3.71) is very similar to (3.49); instead of evaluating the right-hand side gradient at the previous iteration j, we do so at the current iteration $j+1$. From this, we also notice that PP is an implicit method, such that we can use any step size s. Note that (3.71) implies

$$(I + s\nabla f)(\mathbf{u}_{j+1}) = \mathbf{u}_j, \tag{3.72}$$

such that

$$\mathbf{u}_{j+1} = (I - s\nabla f)^{-1}(\mathbf{u}_j). \tag{3.73}$$

By (3.37), noting that $\partial f = \{\nabla f\}$ for a differentiable function f and that $s\nabla f = \nabla(sf)$ by standard differentiation rules, we have that

$$\mathbf{u}_{j+1} = \mathcal{P}_{sf}(\mathbf{u}_j). \tag{3.74}$$

where we have that $(I - s\nabla f)^{-1} = \mathcal{P}_{sf}$ for a differentiable function f; hence, the name *proximal point* method. More generally, we can define the PP algorithm for a convex but possibly nondifferentiable function f, such that the proximal is given in terms of the subdifferential of f, ∂f ((3.14)):

$$\mathbf{u}_{j+1} = (I + s\partial f)^{-1}(\mathbf{u}_j) := \mathcal{P}_{sf}(\mathbf{u}_j) \quad k = 0, 1, \ldots; \tag{3.75}$$

note that since $s > 0$, we have $s\partial f(\cdot) = \partial(sf)(\cdot)$ by the positive multiple rule for subdifferentials fact 4. Note that if f is not convex, then $\mathcal{P}_{sf}$ can be multi-valued.

The proximal point iteration (3.75) converges to a minimizer of a convex and lsc (e.g., continuous) function f; the L-smoothness of f is not required for this convergence, unlike in the case of GD.

Fact 35. *[7, Corollary 26.2] Let $f : \mathbb{R}^N \to \mathbb{R}$ be convex and lsc. Suppose that the set of minimizers of f, $\operatorname{Arg\,min} f$, is nonempty. Then, the PP sequence generated by (3.75), starting from some $\mathbf{u}_0 \in \mathbb{R}^N$, converges to a minimizer of f.*

Like GD, as shown in fact 31, the PP method converges sublinearly with order $\mathcal{O}(1/k)$.

Fact 36. *[7, Corollary 29.5] Assume that $\operatorname{Arg\,min} g \neq \emptyset$. Let $s > 0$. Then, the PP sequence generated by (3.75), starting from some $\mathbf{u}_0 \in \mathbb{R}^N$, converges to a minimizer $\mathbf{u}^*$ of g, i.e., $(\mathbf{u}_k)_{k\in\mathbb{N}} = (\mathcal{P}_{sg}(\mathbf{u}_{k-1}))_{k\in\mathbb{N}} \to \mathbf{u}^*$, where $\mathbf{u}^* \in \operatorname{Arg\,min} g$. Furthermore, letting $g^* := g(\mathbf{u}^*)$, we have that*

$$g(\mathbf{u}_k) - g^* \leq \frac{\|\mathbf{u}_k - \mathbf{u}^*\|}{2sk} \tag{3.76}$$

.

Here, we do not investigate the PP method in greater detail since our interest is not in the PP method itself, but in using PP to approximate a fixed number of GD steps. We discuss this approximation in subsection 5.5.1, but here, as discussed in [75], we remark one way in which PP approximates GD, with their connection being made by the Moreau envelope (see definition 34).

Remark 4 (PP, GD and the Moreau envelope)**.** By (3.47), for some $s > 0$ we have that

$$\mathcal{P}_{sf}(\mathbf{v}) = \mathbf{v} - s\nabla \operatorname{env}_s f(\mathbf{v}). \tag{3.77}$$

Therefore, one PP step can be interpreted as one GD step for minimizing $\operatorname{env}_s f$ isntead of f. In particular, if f is twice differentiable, for a sufficiently small s we have that [75]

$$\mathcal{P}_{sf}(\mathbf{v}) = (I + s\nabla f)^{-1} = \mathbf{v} - s\nabla f(\mathbf{v}) + \mathcal{O}(s) \approx \mathbf{v} - s\nabla f(\mathbf{v}), \tag{3.78}$$

such that for small s we may interpret PP as an approximation of GD.

We note that more convergence results for the PP method can be obtained as corollaries of theorems presented in subsection 3.3.1 for PG.

3.3 Splitting Algorithms for Minimizing Nonsmooth Sums of Functions

Now, we consider the possibly nonsmooth problem of minimizing the sum of functions

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad F(\mathbf{u}), \quad \text{where } F(\mathbf{u}) = f(\mathbf{u}) + g(\mathbf{u}), \tag{3.79}$$

where f is convex and L-smooth, and g is continuous, convex and possibly nondifferentiable; we assume that the set of minimizers of F is nonempty; i.e., $\operatorname{Arg\,min} f \neq \emptyset$. Note that F can in principle be minimized using the PP method, for which the (3.75) update is defined as

$$\mathbf{u}_{k+1} = (I + s\partial F)^{-1}(\mathbf{u}_k) = (I + s\partial(f+g))^{-1}(\mathbf{u}_k), \quad k = 0, 1 \dots. \tag{3.80}$$

However, $I + s\partial F$ may be hard and expensive to compute. Then, we consider the following fact, which relates the zeros of the subdifferentials of f and g to the existence of minimizers.

Fact 37. *[7, Theorem 13.13] Let f, g be convex and lsc on $\mathbb{R}^N$ such that $\operatorname{dom} f \cap \operatorname{dom} g \neq \varnothing$. Let $S := \operatorname{Argmin}(f+g)$ and*

$$Z := \operatorname{zer}(\partial f + \partial g) = \{\mathbf{u} \in \mathbb{R}^N \mid 0 \in \partial f(\mathbf{u}) + \partial g(\mathbf{u})\}. \tag{3.81}$$

Then, $Z \subseteq S$ and one of the following holds:

(i) $Z = S = \varnothing$.

(ii) $Z = \varnothing$ *and* $S \neq \varnothing$.

(iii) $Z = S \neq \varnothing$.

By fact 5, the minimizer of the sum $F \coloneqq f + g$ is characterized by

$$Z \coloneqq \operatorname{zer}(\nabla f + \partial g) = \{\mathbf{u} \in \mathbb{R}^N \mid -\nabla f(\mathbf{u}) \in \partial g(\mathbf{u})\}. \tag{3.82}$$

This formulation allows us to utilize *splitting algorithms* [63, 25], in which the differentiable part f and nondifferentiable part g are handled separately. More generally, these algorithms are constructed for finding zeros of sums of maximally monotone operators (see definition 31), such as ∇f and ∂g in (3.82).

As in section 3.2, we consider two basic iterative methods for solving (3.79). These methods, known as PG and Alternating Proximals (AP), the former explicit and the latter implicit, can be interpreted as generalization of the GD and PP methods, respectively, for (3.79). For those methods, have the following remark in mind.

Remark 5 (Zeros of the subdifferential of positive multiples of functions)**.** We note that by the positive multiple rule for subdifferentials, fact 4, from (3.81) we get

$$Z = \operatorname{zer}(\partial f + \partial g) = (s\partial f + s\partial(s)) = (\partial(sf) + \partial(sg)). \tag{3.83}$$

Therefore, minimizing $f + g$ is equivalent to minimizing $sf + sg$ for some $s > 0$.

3.3.1 Proximal gradient descent (PG)

From (3.82), we have that in order to solve $F \coloneqq f + g$ as in (3.79), we may minimize the differentiable part f using GD (3.49) and the nondifferentiable part g using PP (3.75); this is achieved by the PG algorithm. Starting from some $\mathbf{u}_0 \in \mathbb{R}^N$, given a stepsize $0 < s < \frac{2}{L}$, the PG method iteration takes the form

$$\mathbf{u}_{k+1} \coloneqq \mathcal{P}_{sg} \circ \mathcal{G}_{sf}(\mathbf{u}_k) = (I + s\partial g)^{-1} \left((I - s\nabla f)(\mathbf{u}_k)\right), \quad k = 0, 1, \ldots, \tag{3.84}$$

where $\mathcal{P}_{sg}$ is the proximal of sg (3.34) and $\mathcal{G}_{sf}$ is the gradient descent step for f of size s (3.49). Thus, the PG method is a composition of a GD step over f and a PP step over g. We note that PG inherits the stability constraint of GD, $s < 2/L$; while in principle $\mathcal{P}_{sg}$ may accept larger step sizes $s \geq 2/L$, we require that s be the same for both $\mathcal{G}_{sf}$ and $\mathcal{P}_{sg}$.

Note that if $f = 0$, such that the composite function F in (3.79) simply becomes $F = g$, we have $\mathcal{G}_{sf} = I$ such that $T = \mathcal{P}_{sg}$. That is, the PG update (3.84) reduces to the PP update, (3.75). Similarly, for $g = 0$, such that $F = f$, we have $\mathcal{P}_{sg} = I$, such that (3.84) reduces to the GD update (3.49). Thus, PG generalizes both PP and GD.

Proximal gradient descent for convex, L-smooth f

We now briefly present converge results for PG, in the setting of (3.79) for aconvex and L-smooth f and a convex g. Then, for $s = 1/L$ in particular, we have that the PG sequence to a solution of (3.79) as detailed in the following facts.

Fact 38. *Let $f : \mathbb{R}^N \to \mathbb{R}$ be a convex, L-smooth function for some Lipschitz constant $L > 0$, and let $g : \mathbb{R}^N \to [-\infty, +\infty]$ be convex and lsc. Let $F := f + g$, as in (3.79), and let $\operatorname{Arg\,min} F \neq \emptyset$. Let $\mathcal{P}_{sg} \circ \mathcal{G}_{sf}$ be the PG operator (3.84) with stepsize $s = 1/L$. Then, the PG sequence generated by (3.84) converges Féjer-monotonically to a minimizer of F.*

Fact 39. *Let $F^* := f(\mathbf{u}^*)$, where $\mathbf{u}^* \in \operatorname{Arg\,min} F$. The PG sequence generated by (3.84), with $s = 1/L$, converges monotonically in the function value with convergence rate $\mathcal{O}(1/k)$; i.e., for $k \geq 1$,*

$$0 \leq F(\mathbf{u}_{k+1}) - F^* \leq F(\mathbf{u}_k) - F^* \leq \frac{L\|\mathbf{u}_0 - \mathbf{u}^*\|^2}{2k}. \tag{3.85}$$

Furthermore, the PG sequence is asymptotically regular, by which we mean distance between successive iterates is bounded (here, by $\mathcal{O}(1/\sqrt{k})$):

$$\|\mathbf{u}_{k-1} - \mathbf{u}_k\| \leq \frac{\sqrt{2}\|\mathbf{u}_0 - \mathbf{u}^*\|}{\sqrt{k}}. \tag{3.86}$$

From fact 39, we can recover fact 31 as a corollary by setting $g = 0$, such that $F = f$ and $\mathcal{P}_{sg}\mathcal{G}_{sf} = \mathcal{G}_{sf}$. Similarly, by setting $f = 0$, we can prove the convergence rate of PP (3.75) in function values, as stated in fact 36, as a corollary of fact 39.

Proximal gradient descent for strongly convex, L-smooth f

Now, assuming f tk be μ-strongly convex, we note that PG has the same linear convergence rate for (3.79) as GD's for (3.48), as stated in fact 34. In fact, the result for GD is proven as a corollary of the theorem for PG, taking $g = 0$ such that $F = f$.

Fact 40. *[9, Theorem 10.29] The PG sequence (3.84) for a L-smooth and μ-strongly convex f, with $s = 1/L$, satisfies*

(a) $\|\mathbf{u}_{k+1} - \mathbf{u}^*\|^2 \leq \left(1 - \frac{\mu}{L}\right)\|\mathbf{u}_k - \mathbf{u}^*\|^2$;

(b) $\|\mathbf{u}_k - \mathbf{u}^*\|^2 \leq \left(1 - \frac{\mu}{L}\right)^k \|\mathbf{u}_0 - \mathbf{u}^*\|^2$;

(c) $F(\mathbf{u}_{k+1}) - F^* \leq \frac{L}{2}\left(1 - \frac{\mu}{L}\right)^{k+1} \|\mathbf{u}_0 - \mathbf{u}^*\|^2$.

Thus, we note that PG and GD have the same convergence rates of $\mathcal{O}(1/k)$ (sublinear) for convex problems and $\mathcal{O}(1-\mu/L)^k$ (linear) for strongly convex problems, but we have yet to interpret the meaning of these convergence rates; we do so in the following.

Finite-time convergence and complexity of proximal gradient descent

In practice, when minimizing some function F using an iterative method, we do not wait until the minimum is exactly attained, i.e., $F(\mathbf{u}_k) = F^*$ for some k. In fact, for methods such as GD and PG, for general problems such as (3.48) and (3.79) we only converge asymptotically to $\mathbf{u}^* \in \operatorname{Arg\,min} f$. Therefore, we must establish a finite-time convergence criteria. Thus, instead of computing an exact minimizer $\mathbf{u}^*$, we consider approximating the minimum function value F^* within some accuracy threshold. That is, given some accuracy parameter $0 < \epsilon \ll 1$, we iterate until the kth iterate satisfies

$$F(\mathbf{u}_k) - F^* \leq \epsilon. \tag{3.87}$$

Note that the convergence rates from fact 39 for convex problems and fact 40 for strongly convex problems bound the $F(\mathbf{u}_k) - F^*$ term from above. Thus, given ϵ, we are able to determine the maximum number of iterations, k, require to satisfy the ϵ-accuracy threshold from (3.87). We refer to this as the *complexity* of the method.

In the following, we establish the complexity of the more general PG method, noting that by proxy this applies to the special cases GD (3.49) (when $g = 0$) and PP (3.75) (when $f = 0$). Before we do so, we note that the μ/L ratio that appears in the strongly convex convergence rates for GD (fact 34) and PG (fact 40) has a special meaning: it is the inverse of the *condition number* of f, which we now define.

Definition 36 (Condition number)**.** *Let $f : \mathbb{R}^N \to \mathbb{R}$ be L-smooth and μ-strongly convex function. Then, the condition number κ of f is given by*

$$\kappa := \frac{L}{\mu} \geq 1. \tag{3.88}$$

When $\kappa = 1$, we say that the problem of minimizing f is perfectly-conditioned; when $\kappa \geq 1$ is small, we say that it is well-conditioned and, when $\kappa \gg 1$, we say that it is ill-conditioned.

By definition 36, note that we can replace the $1 - \mu/L$ terms in the strongly convex bounds from fact 34 for GD and fact 40 for PG by $1 - 1/\kappa$.

Fact 41 (Complexity of the PG method)**.** *[9, Theorems 10.25 and 10.30]*

Let $F \coloneqq f+g$ be defined as in (3.79), where f is convex and L-smooth, and g is convex and possibly nondifferentiable, and consider minimizing F using PG. Let $\mathbf{u}^* \in \operatorname{Arg\,min} F$ be a minimizer of F, and define $F^* \coloneqq F(\mathbf{u}^*)$. Suppose that given some $0 < \varepsilon \ll 1$, we stop iterating when $F(\mathbf{u}_k) - F^* \leq \varepsilon$ is satisfied for some k. Let R be an upper bound for $\|\mathbf{u}_0 - \mathbf{u}^*\|$, and recall that $\kappa = L/\mu$ (3.88). Then,

(a) If f is convex, $F(\mathbf{u}_k) - F^* \leq \varepsilon$ is attained for any $k \geq 1$ satisfying

$$k \geq \left\lceil \frac{LR^2}{2\varepsilon} \right\rceil \sim \mathcal{O}\left(\frac{L}{\epsilon}\right). \tag{3.89}$$

(b) If f is strongly convex, $F(\mathbf{u}_k) - F^* \leq \varepsilon$ is attained for any $k \geq 1$ satisfying

$$k \geq \kappa \log\left(\frac{1}{\varepsilon}\right) + \kappa \log\left(\frac{LR^2}{2}\right) \sim \mathcal{O}\left(\kappa \log\left(\frac{1}{\varepsilon}\right)\right). \tag{3.90}$$

For example, if we want the function value to be within 10^{-6} of the exact minimizer, we need around 1 million iterations of PG! Even for strongly convex functions, in many practical cases we require tens or hundreds of thousands of iterations for many relevant optimization algorithms, some with even faster convergence rates than GD and PG [10, 16, 74].

3.3.2 Alternating proximals (AP)

When it is necessary to take steps of size $s \geq 2/L$, we must consider an implicit variant of PG, which is limited by the $s < 2/L$ stability limit of GD. Hence, for the same reasoning that led us to consider replacing GD by PP, we now consider replacing the gradient descent (3.49) step $\mathcal{G}_{sf}$ of PG by a proximal point (3.75) step $\mathcal{P}_{sf}$; this yields the AP method. Recalling (3.82), this amounts to minimizing both the differentiable part f and the nondifferentiable part g of $F \coloneqq f + g$ (3.79) with PP (3.75); hence, as the name of this method suggests, we alternate between applying proximals on f and g. Starting from some $\mathbf{u}_0 \in \mathbb{R}^N$ and given $s > 0$, the AP iteration is given by

$$\mathbf{u}_{k+1} \coloneqq \mathcal{P}_{sg} \circ \mathcal{P}_{sf}(\mathbf{u}_k) = (I + s\partial g)^{-1}\left((I + s\nabla f)^{-1}(\mathbf{u}_k)\right), \tag{3.91}$$

where $\mathcal{P}_{sg}$ and $\mathcal{P}_{sf}$ are proximals (3.34) of sg and sf, respectively, and we note that $\partial f = \{\nabla f\}$ since f is differentiable by assumption. This method (3.91) is a composition of two proximal steps over f and g. Unfortunately, (3.91) does not generally converge to a minimizer of (3.79).

Fact 42. *[7, Proposition 31.1] Suppose that the function* $\operatorname{env}(f)+g$ *has a minimizer, and let* $\mathbf{u}_0 \in \mathbb{R}^N$*. Then the sequence* $\left(\left(\mathcal{P}_g \circ \mathcal{P}_f\right)^k(\mathbf{u}_0)\right)_{k\in\mathbb{N}}$ *converges to a minimizer of* $\operatorname{env}(f)+g$*, where* $\operatorname{env}(f)$ *is the Moreau envelope of* f *(definition 34).*

Corollary 1. *The AP sequence generated by (3.91),* $\left(\left(\mathcal{P}_{sg} \circ \mathcal{P}_{sf}\right)^k(\mathbf{u}_0)\right)_{k\in\mathbb{N}}$*, converges to a minimizer of* $\operatorname{env}_s f+g$*.*

Proof. By fact 42, it follows that (3.91) converges to a minimizer of $\operatorname{env}(sf)+sg$. Note that

$$\begin{aligned}
\operatorname{env}(sf)(\mathbf{v}) &= \inf_{\mathbf{u}}\left\{sf(\mathbf{u})+\frac{1}{2}\|y-\mathbf{v}\|^2\right\} \\
&= \inf_{\mathbf{u}}\left[s\left(f(\mathbf{u})+\frac{1}{2s}\|\mathbf{u}-\mathbf{v}\|^2\right)\right] \\
&= s\inf_{y}\left\{f(\mathbf{u})+\frac{1}{2s}\|\mathbf{u}-\mathbf{v}\|^2\right\} \\
&= s\operatorname{env}_s f(\mathbf{v}),
\end{aligned} \tag{3.92}$$

such that $\operatorname{env}(sf)(\mathbf{v})+sg = s\operatorname{env}_s f(\mathbf{v})+sg$. Finally, since $s>0$, it follows from (3.83) that

$$\operatorname{Arg\,min}(s\operatorname{env}_s f(\mathbf{v})+sg) = \operatorname{Arg\,min}(\operatorname{env}_s f(\mathbf{v})+g), \tag{3.93}$$

such that (3.91) converges to a minimizer of $\operatorname{env}_s f(\mathbf{v})+g$ for all $s>0$. □

Similarly to how we study the relation between PP and GD, we are not interested in AP insomuch as we are in approximating a fixed number of PG steps with one AP step, which we later explore in subsection 5.5.2.

Chapter 4

Parallel-in-time multigrid methods

A defining feature of time-dependent problems is what is known as the *causality principle* [73], which states that solutions at a given time point are determined by the solutions at earlier time points. In time integration problems of parabolic and hyperbolic Ordinary Differential Equation (ODE)s, such as the heat and wave equations described in subsection 2.6.1 and subsection 2.6.2, respectively, since solutions of the form $\mathbf{u}(\mathbf{x}, t_i)$ must be computed for all $(t_1, \ldots, t_n)$ time points that discretize a continuous time interval $(0, T]$, the number of time points of the discretized temporal mesh yields a sequential bottleneck in time. Beginning in [72], many parallel-in-time (P-in-T) methods have been developed in the past 50 years [34, 42, 73] for time integration problems, which in different ways allow for the solutions at different time points to be computed simultaneously in parallel. In particular, a subset of these methods are based on multigrid methods, such as the parareal method [62], the Parallel Full Approximation Scheme in Space-Time (PFASST) method [26] and the Multigrid Reduction-in-Time (MGRIT) method [27]. While our parallel-in-iteration framework, to be introduced in chapter 5, generally accepts any suitable P-in-T method, the analysis in this thesis is restricted to the MGRIT method, which we explore in this chapter.

4.1 Multigrid Reduction-In-Time (MGRIT)

The MGRIT method [27] was developed as a multilevel generalization of the 2-level parareal method [62] for solving time-dependent systems of differential equations in parallel. Conversely, we may describe parareal as a special 2-level case of MGRIT [27, 39, 40], which is

what we do here. To outline the MGRIT method, let us consider a general Initial Boundary Value Problem (IBVP) (such as (2.133) and (2.142)) of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = \mathcal{L}(t, \mathbf{u}(t)), \quad \mathbf{u}(0) = \mathbf{u}_0, \quad t \in (0, T]. \tag{4.1}$$

Discretizing this system of ODEs using a 1-step method (e.g., any Runge-Kutta method, such as forward-Euler (2.134) and backward-Euler (2.135)) results in a system of equations of the form

$$\begin{aligned} \mathbf{u}_0 &= \mathbf{w}_0, \\ \mathbf{u}_i &= \Phi_i(\mathbf{u}_{i-1}) + \mathbf{w}_i, \quad i = 1, 2, \ldots, N_t, \end{aligned} \tag{4.2}$$

where $\mathbf{w}_0 \equiv \mathbf{u}_0$, and $t_i = i\Delta t$, $i = 0, 1, \ldots, N_t$ is a temporal mesh with spacing $\Delta t = \frac{T}{N_t}$, which we assume is constant for simplicity. Furthermore, $\mathbf{u}_i \approx \mathbf{u}(t_i)$ is the discrete approximation to $\mathbf{u}(t)$ at time t_i, and Φ_i is the *time-stepping operator*, responsible for advancing the solution at t_{i-1} to t_i; for simplicity, we further assume Φ_i is a time-independent operator, such that $\Phi_i(\cdot) = \Phi(\cdot)$ for all $i = 1, 2, \ldots, N_t$. The vector $\mathbf{w}_i$ represents solution-independent forcing terms in the discretized problem. We can then rewrite (4.2) as the following "all-at-once" system:

$$\mathcal{A}(\mathbf{u}) = \begin{bmatrix} I & & & \\ -\Phi(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N_t} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{N_t} \end{bmatrix} = \mathbf{w}. \tag{4.3}$$

For our purposes, it is also pertinent to introduce the associated residual vector, which, given some $\mathbf{v} \approx \mathbf{u}$, is defined as

$$\mathbf{r}(\mathbf{v}) = \mathbf{w} - \mathcal{A}(\mathbf{v}) = \begin{bmatrix} \mathbf{w}_0 - \mathbf{v}_0 \\ \mathbf{w}_1 - [\mathbf{v}_1 - \Phi(\mathbf{v}_0)] \\ \vdots \\ \mathbf{w}_{N_t} - [\mathbf{v}_{N_t} - \Phi(\mathbf{v}_{N_t-1})] \end{bmatrix} = \begin{bmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \\ \vdots \\ \mathbf{r}_{N_t} \end{bmatrix}. \tag{4.4}$$

Instead of solving (4.2) sequentially with time-stepping, MGRIT aims to solve it iteratively and in parallel via the introduction of a coarser approximation of the problem. To this end, given an integer coarsening factor $m > 1$, we introduce a coarse temporal mesh $T_j = j\Delta T$, $j = 0, 1, \ldots, N_T$, where $N_T = \frac{N_t}{m}$, with constant spacing $\Delta T = m\Delta t$. This partitions the $N_t + 1$ points of the fine grid into two sets: 1. *C-points*, which are the $N_T + 1$ points appearing exclusively on the coarse grid, and 2. *F-points* which are the remaining $N_t - N_T$

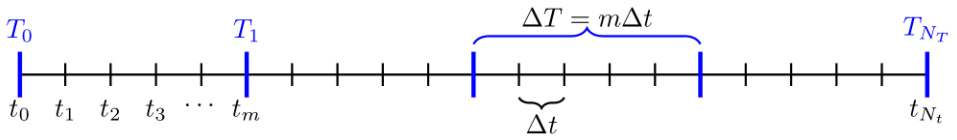


Figure 4.1: Fine- and coarse-grid discretization meshes with coarsening factor m. F-points (in black) are points t_i for $i = 1, \ldots, N_t$ such that $i \neq jm$ for $j = 1, \ldots, N_T = \frac{N_t}{m}$, and C-points (in blue) are points $T_j = t_{jm}$ for $j = 1, \ldots, N_T = \frac{N_t}{m}$.

points that do not appear on the coarse grid. figure 4.1 illustrates the fine- and coarse-grid temporal meshes.

In remark 1, we first noted how multigrid methods can be typically applied for time-dependent problems. In particular, the discretization approach remains very similar, albeit applied to a temporal domain rather than a spatial domain, the latter discussed in most of chapter 2. The following remark further establishes the parallel between the temporal discretization approach of MGRIT to the discussion on spatially-discretized methods in chapter 2.

Remark 6 (MGRIT temporal discretization)**.** For MGRIT's temporal discretization approach, we note that Δt plays a role similar to that of h in the spatial discretizations discussed in chapter 2. Similarly, the continuous-time domain $(0, T]$ and the fine- and coarse-grids from figure 4.1 play analogous roles to the continuous spatial domain Ω and discretized fine- and coarse-grid equispaced meshes Ω_h and Ω_{mh}, with m being the coarsening factor of the (spatial) spacing h. For example, in (4.3), we note that $\mathcal{A}$, $\mathbf{u}$ and $\mathbf{w}$ play similar roles to those of L_h, $\mathbf{u}_h$ and $\mathbf{f}_h$ in (2.104). When $\mathcal{A}(\cdot)$ is assumed to be linear, such as in (4.7), then the resulting all-at-once system is analogous to (2.85), with $\mathcal{A}$ this time playing an equivalent role to L_h. Hence, most of the discussion from chapter 2 applies analogously here, only replacing spatial discretizations by temporal discretizations.

As is standard for a multigrid method (see chapter 2), an MGRIT iteration consists of a coarse-grid correction step sandwiched between pre- and post-relaxation steps. We detail these steps in the following subsections.

4.1.1 The MGRIT F- and C-relaxation schemes

MGRIT employs simple relaxation schemes, consisting of fine-grid steps with the operator Φ, which we call the F- and C-relaxation schemes [27, 24].

F-relaxation

An F-relaxation is the process where, starting from a C-point, the time-stepping operator is used to update the values of the $m-1$ F-points that follow it (that is, until the F-point that immediately precedes the next C-point). This update is given by

$$\left.\begin{aligned} \mathbf{u}_{jm+1} &\leftarrow \Phi\mathbf{u}_{jm} + \mathbf{w}_{jm+1} \\ \mathbf{u}_{jm+2} &\leftarrow \Phi\mathbf{u}_{jm+1} + \mathbf{w}_{jm+2} \\ &\vdots \\ \mathbf{u}_{(j+1)m-1} &\leftarrow \Phi\mathbf{u}_{(j+1)m-2} + \mathbf{w}_{(j+1)m-1} \end{aligned}\right\} \quad \text{for } j = 0, 1, \ldots, N_T - 1. \tag{4.5}$$

C-relaxation

Similarly, C-relaxation is the update of each C-point by stepping from the preceding F-point using Φ. Because there is no F-point preceding the first time point with index $i = 0$, $\mathbf{u}_0$ is left unchanged, such that it satisfies the initial condition $\mathbf{u}_0 = \mathbf{w}_0$ across all iterations. Thus, the C-relaxation update is given by

$$\begin{aligned} \mathbf{u}_0 &\leftarrow \mathbf{w}_0 \\ \mathbf{u}_{km} &\leftarrow \Phi\mathbf{u}_{jm-1} + \mathbf{w}_{jm} \quad \text{for } j = 1, \ldots, N_T. \end{aligned} \tag{4.6}$$

We illustrate both F- and C-relaxation schemes in figure 4.2; note that both F- and C-relaxations can be performed in parallel.



(a) F-relaxation **(b)** C-relaxation

Figure 4.2: Illustration of F- and C-relaxation schemes. Here, the blue squares represent C-points and the white circles represent F-points. The Φ symbol followed by an arrow denotes a forward time-step from one time point to the following one using the operator Φ.

The standard MGRIT pre-relaxation, as considered here, is known as FCF-relaxation [27], and combines an F-, a C-, and an F-relaxation in this order. For parareal, an F-relaxation is used instead. For both MGRIT and parareal, the post-relaxation scheme is taken to be an F-relaxation.

4.1.2 The MGRIT coarse-grid correction problem

In between the pre- and post-relaxations, a coarse-grid correction problem is solved to provide a correction at C-points, as we now describe. MGRIT has both linear and Full Approximation Scheme (FAS) variants [55], suitable for linear and nonlinear Φ, respectively, and these variants differ in the way that coarse-grid correction is performed.

Linear MGRIT

Let us first explain how this works in the linear case, and to this end let us assume that Φ_i is a time-independent, linear operator, such that $\Phi_i(\mathbf{u}_{i-1}) = \Phi\mathbf{u}_{i-1}$ corresponds to a matrix-vector product with matrix Φ. We also write the action of the space-time operator $\mathcal{A}(\mathbf{u}) = \mathcal{A}\mathbf{u}$ as a matrix-vector product, yielding the linear all-at-once system

$$\mathcal{A}\mathbf{u} = \begin{bmatrix} I & & & \\ -\Phi & I & & \\ & \ddots & \ddots & \\ & & -\Phi & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N_t} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{N_t} \end{bmatrix} = \mathbf{w}. \tag{4.7}$$

From (4.2), we can then write, for coarse indices $j = 1, 2, \ldots, N_T$,

$$\mathbf{u}_{jm} = \Phi\mathbf{u}_{jm-1} + \mathbf{w}_{jm} = \cdots = \Phi^m\mathbf{u}_{(j-1)m} + \widetilde{\mathbf{w}}_{jm}, \tag{4.8}$$

where $\widetilde{\mathbf{w}}_{jm} = \mathbf{w}_{jm} + \Phi\mathbf{w}_{jm-1} + \cdots + \Phi^{m-1}\mathbf{w}_{(j-1)m+1}$. Therefore, the fine-grid system (4.3) is equivalent to the following system written only in terms of the C-points:

$$\begin{aligned} \mathbf{u}_0 &= \mathbf{w}_0, \\ \mathbf{u}_{jm} &= \Phi^m\mathbf{u}_{(k-1)m} + \widetilde{\mathbf{w}}_{jm}, \quad j = 1, 2, \ldots, N_T, \end{aligned} \tag{4.9}$$

where Φ^m plays the role of time-stepping operator for this coarse system. Or, in block-matrix form

$$\mathcal{A}_\Delta\mathbf{u}_\Delta = \begin{bmatrix} I & & & \\ -\Phi^m & I & & \\ & \ddots & \ddots & \\ & & -\Phi^m & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_m \\ \vdots \\ \mathbf{u}_{N_Tm} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \widetilde{\mathbf{w}}_m \\ \vdots \\ \widetilde{\mathbf{w}}_{N_Tm} \end{bmatrix} = R_\Phi\mathbf{w} \coloneqq \mathbf{w}_\Delta, \tag{4.10}$$

where R_Φ is the rectangular restriction operator

$$R = \begin{bmatrix} I & & & & & & & & \\ & \Phi^{m-1} & \dots & \Phi & I & & & & \\ & & & & & \ddots & & & \\ & & & & & & \Phi^{m-1} & \dots & \Phi & I \end{bmatrix}. \tag{4.11}$$

While system (4.10) has a factor of m fewer equations than (4.3), it is no less expensive to solve because (naively) computing the action of Φ^m is m times as expensive as computing that of Φ. So, instead, MGRIT solves an approximation to the *ideal* coarse-grid system (4.10) arising from replacing the *ideal* coarse-grid time-stepping operator Φ^m with an approximation,

$$\Psi \approx \Phi^m; \tag{4.12}$$

we note that Φ_Δ is often used instead of Φ in the MGRIT literature, such as [27, 24]. That is, rather than inverting $\mathcal{A}_\Delta$ in (4.10), one inverts the approximation $\mathcal{B}_\Delta$ given by

$$\mathcal{B}_\Delta = \begin{bmatrix} I & & & \\ -\Psi & I & & \\ & \ddots & \ddots & \\ & & -\Psi & I \end{bmatrix}. \tag{4.13}$$

Let us now explain in more detail how MGRIT uses the approximation $\mathcal{B}_\Delta \approx \mathcal{A}_\Delta$. Consider the coarse system $\mathcal{A}_\Delta \mathbf{u}_\Delta = \mathbf{w}_\Delta$ (4.10). Let $\mathbf{v}_\Delta \approx \mathbf{u}_\Delta$ be an approximate solution of (4.10) with error $\mathbf{e}_\Delta = \mathbf{u}_\Delta - \mathbf{v}_\Delta$. Then, note that

$$\mathcal{A}_\Delta \mathbf{e}_\Delta = \mathcal{A}_\Delta \mathbf{u}_\Delta - \mathcal{A}_\Delta \mathbf{v}_\Delta = \mathbf{w}_\Delta - \mathcal{A}\mathbf{v}_\Delta = \mathbf{r}_\Delta, \tag{4.14}$$

where $\mathbf{r}_\Delta$ denotes the coarse-grid residual. Since $\mathcal{A}_\Delta$ is invertible, it follows that $\mathbf{e}_\Delta = \mathcal{A}_\Delta^{-1}\mathbf{r}_\Delta$, such that

$$\mathbf{u}_\Delta = \mathbf{v}_\Delta + \mathbf{e}_\Delta = \mathbf{v}_\Delta + \mathcal{A}_\Delta^{-1}\mathbf{r}_\Delta. \tag{4.15}$$

Assuming that $\Psi \approx \Phi^m$ such that $\mathcal{B}_\Delta^{-1} \approx \mathcal{A}_\Delta^{-1}$, then

$$\mathbf{u}_\Delta \approx \mathbf{v}_\Delta + \mathcal{B}_\Delta^{-1}\mathbf{r}_\Delta. \tag{4.16}$$

Let $\left\{\mathbf{u}_\Delta^{(k)}\right\}_{k=0,1,\dots}$ be a sequence of approximate solutions generated by the $k = 0, 1, \dots$ MGRIT iterations. Letting $\mathbf{u}_\Delta^{(k+1)}$ and $\mathbf{u}_\Delta^{(k)}$ play the roles of $\mathbf{u}_\Delta$ and $\mathbf{v}_\Delta$, respectively,

where we think of $\mathbf{u}_\Delta^{(k+1)}$ as a better approximation of $\mathbf{u}_\Delta$ than $\mathbf{u}_\Delta^{(k)}$, we obtain the MGRIT residual correction scheme:

$$\mathbf{u}_\Delta^{(k+1)} = \mathbf{u}_\Delta^{(k)} + \mathcal{B}_\Delta^{-1}\left(\mathbf{w}_\Delta - \mathcal{A}_\Delta \mathbf{u}_\Delta^{(k)}\right), \quad k = 0, 1, \ldots \tag{4.17}$$

Component-wise, this may be written with Φ and Ψ as

$$\mathbf{u}_{(j+1)m}^{(k+1)} = \Psi \mathbf{u}_{jm}^{(k+1)} + \Phi^m \mathbf{u}_{jm}^{(k)} - \Psi \mathbf{u}_{jm}^{(k)} + \widetilde{\mathbf{w}}_{(j+1)m}, \tag{4.18}$$

where $\mathbf{u}_0^{(k)} = \mathbf{w}_0$.

FAS-MGRIT

Now let us consider the MGRIT coarse-grid correction in the more general FAS case (see subsection 2.5.4), where $\Phi(\cdot)$ in (4.3) is not (necessarily) a linear operator. Let $\mathbf{v} \approx \mathbf{u}$ be an approximate solution of $\mathcal{A}(\mathbf{u}) = \mathbf{w}$ in (4.3). After performing the FCF-relaxation step, the fine-grid approximation $\mathbf{v}$ and nonlinear residual $\mathbf{w} - \mathcal{A}(\mathbf{v})$ are injected to the coarse-grid:

$$\mathbf{v}_\Delta = R_I(\mathbf{v}), \quad \mathbf{r}_\Delta = R_I(\mathbf{w} - \mathcal{A}(\mathbf{v})), \tag{4.19}$$

where R_I stands for the injection operator. The *ideal* FAS coarse-grid problem is then to solve the system

$$\mathcal{A}_\Delta(\mathbf{u}_\Delta) = \mathcal{A}_\Delta(\mathbf{v}_\Delta) + \mathbf{r}_\Delta \tag{4.20}$$

for the coarse-grid solution $\mathbf{u}_\Delta$, where similarly to (4.10),

$$\mathcal{A}_\Delta(\cdot) = \begin{bmatrix} I & & & \\ -\Phi^m(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi^m(\cdot) & I \end{bmatrix}. \tag{4.21}$$

The coarse-grid system (4.20) results from the elimination of F-points variables from the residual equation $\mathcal{A}(\mathbf{u}) - \mathcal{A}(\mathbf{v}) = \mathbf{r}(\mathbf{v})$. Similarly to the linear case, the MGRIT iteration solves an approximation of (4.20),

$$\mathcal{B}_\Delta(\mathbf{u}_\Delta) = \mathcal{B}_\Delta(\mathbf{v}_\Delta) + \mathbf{r}_\Delta, \tag{4.22}$$

where $\mathcal{B}_\Delta(\cdot) \approx \mathcal{A}_\Delta(\cdot)$ replaces the $\Phi^m(\cdot)$ terms in $\mathcal{A}_\Delta(\cdot)$ by a computationally cheaper $\Psi(\cdot) \approx \Phi^m(\cdot)$. Letting $\mathbf{u}_\Delta^{(k+1)}$ and $\mathbf{u}_\Delta^{(k)}$ play the roles of $\mathbf{u}_\Delta$ and $\mathbf{v}_\Delta$, respectively, the component-wise FAS residual correction scheme is then given by

$$\mathbf{u}_{(j+1)m}^{(k+1)} = \Psi\left(\mathbf{u}_{jm}^{(k+1)}\right) + \Phi\left(\mathbf{u}_{(j+1)m-1}^{(k)}\right) - \Psi\left(\mathbf{u}_{jm}^{(k)}\right) + \mathbf{w}_{(j+1)m}, \quad j = 1, 2, \ldots, N_T. \tag{4.23}$$

We note that when $\Phi(\cdot)$ and $\Psi(\cdot)$ are linear operators Φ and Ψ, the FAS variant simplifies to the linear variant of MGRIT, as generally discussed in subsection 2.5.4.

4.2 MGRIT from an algorithmic perspective

We now formally define the linear MGRIT and FAS-MGRIT algorithms. As in section 2.4, we begin by defining 2-level schemes, which are analogous to algorithm 2 albeit with temporal discretization instead of spatial discretization.

4.2.1 Linear MGRIT algorithm

We define the 2-level linear MGRIT algorithm below in algorithm 6.

Algorithm 6 2-level MGRIT($\Phi, \Psi, \mathbf{u}, \mathbf{w}, \varepsilon$).

1: **repeat**
2: Apply FCF-relaxation to $\mathcal{A}\mathbf{u} = \mathbf{w}$ using Φ.
3: Compute the residual on the coarse grid: $\mathbf{r}_\Delta = \mathbf{w}_\Delta - \mathcal{A}\mathbf{u}_\Delta$.
4: Solve the coarse-grid correction problem using Ψ: $\mathcal{B}_\Delta \mathbf{u}_\Delta = \mathbf{w}_\Delta$.
5: Correct the approximate solution at the C-points.
6: Apply F-relaxation to $\mathcal{A}\mathbf{u} = \mathbf{w}$ using Φ.
7: **until** $\|\mathbf{r}\|/\|\mathbf{r}^{(0)}\| \leq \varepsilon$.

Based on [24], we detail each step of the MGRIT algorithm, which helps illustrate how each component of the current approximate solution $\mathbf{u}$ is updated at each stage of the MGRIT algorithm.

Pre-relaxation

As discussed in subsection 4.1.1, for a typical MGRIT solve we perform an FCF-relaxation, which is composed of an F-relaxation, defined in (4.5), a C-relaxation, defined (4.6), followed by another F-relaxation. Each F-relaxation has $\mathcal{O}(N_t - N_T)$ operations, while each C-relaxation has $\mathcal{O}(N_T)$ operations; however, recall that both can be performed in parallel. In particular, note from (4.5) that when initializing the MGRIT algorithm with some initial approximate solution vector $\mathbf{u}$, only the C-point vectors (i.e., $\mathbf{u}_{jm}$ for $j = 0, 1, \ldots, N_T$) need to be initialized, as the values at the F-points are immediately overwritten by the first F-relaxation.

Coarse-grid residual

After the FCF-relaxation, the coarse-grid residual is computed based on the relaxed approximate solution. This residual computation takes the form

$$\begin{aligned} \mathbf{r}_0 &\leftarrow \mathbf{w}_0 - \mathbf{u}_0 \\ \mathbf{r}_{jm} &\leftarrow \mathbf{w}_{jm} - \mathbf{u}_{km} + \Phi \mathbf{u}_{jm-1}, \quad j = 1, 2, \ldots, N_T. \end{aligned} \tag{4.24}$$

Note that this residual computation can be performed in parallel.

The standard Euclidean norm $\|\cdot\|_2$ of the MGRIT residual, $\mathbf{r}$ (4.4), is used in the halting criteria. We note that since the residual is measured after performing the FCF-relaxations, by (4.5) the residual at the F-points will be zero, such that it is sufficient to measure the residual at the C-points, which is described component-wise by (4.24). That is, letting $\mathbf{r}_c$ denote the C-point residual, we have that

$$\|\mathbf{r}\|_2 = \sqrt{\sum_{i=1}^{N_t} \|\mathbf{r}_i\|_2^2} = \sqrt{\sum_{j=1}^{N_T} \|\mathbf{r}_{jm}\|_2^2} = \|\mathbf{r}_c\|_2 \tag{4.25}$$

holds immediately after the (4.24) step is computed.

Coarse-grid correction

MGRIT then solves an approximate coarse-grid correction problem given by

$$\begin{aligned} \mathbf{c}_0 &= \mathbf{r}_0 \\ \mathbf{c}_{jm} &= \Psi \mathbf{c}_{(j-1)m} + \mathbf{r}_{km}, \quad j = 1, 2, \ldots, N_T, \end{aligned} \tag{4.26}$$

where $\mathbf{c}_{jm}$ approximates the error at the C-points jm, $j = 0, 1 \ldots, N_T$. Note that this problem is solved sequentially with $\mathcal{O}(N_T)$ operations. Given the solutions $\mathbf{c}_{jm}$, $j = 0, 1 \ldots, N_T$, of (4.27), the C-points of the approximate solution are updated by

$$\mathbf{u}_{jm} \leftarrow \mathbf{u}_{jm} + \mathbf{c}_{jm}, \quad j = 0, 1, \ldots, N_T. \tag{4.27}$$

Note that this correction step can be done in parallel.

Post-relaxation

We note that while in a rigorous description of the MGRIT algorithm we describe a post F-relaxation step as the final step of each MGRIT iteration, this post-relaxation is typically omitted for 2-level due to being made redundant by the first F-relaxation of the next MGRIT iteration. Then, in principle we would only necessary to perform this F-relaxation on the final MGRIT iteration (i.e., when the MGRIT algorithm halts), since there is no subsequent iteration to perform a pre-relaxation step on. However, since the MGRIT algorithm will halt when the residual of the approximate solution, and this residual is computed immediately after the pre-relaxation step, the post-relaxation step in this final iteration is also superfluous. We note that for MGRIT with 3 or more levels, a post-relaxation step must be performed in all levels except for the finest one.

4.2.2 FAS-MGRIT algorithm

Similarly, the FAS-MGRIT algorithm is defined in algorithm 7. We do not detail this algorithm step-by-step for brevity, but note that algorithm 7 simply an implementation of the FAS multigrid scheme, algorithm 5, for the MGRIT method.

Algorithm 7 2-level FAS-MGRIT($\Phi, \Psi, \mathbf{u}, \mathbf{w}, \varepsilon$).

1: **repeat**
2: Apply FCF-relaxation to $\mathcal{A}(\mathbf{u}) = \mathbf{w}$ using Φ.
3: Inject the fine-grid approximate solution and residual to the coarse grid: $\mathbf{u}_\Delta = R_I(\mathbf{u}), \quad \mathbf{r}_\Delta = R_I(\mathbf{w} - \mathcal{A}(\mathbf{u}))$.
4: Solve the coarse-grid correction problem: $\mathcal{B}_\Delta(\mathbf{v}_\Delta) = \mathcal{B}_\Delta(\mathbf{u}_\Delta) + \mathbf{r}_\Delta$.
5: Compute the coarse-grid error approximation: $\mathbf{e}_\Delta = \mathbf{v}_\Delta - \mathbf{u}_\Delta$.
6: Correct the approximate solution at the C-points using ideal interpolation: $\mathbf{u} = \mathbf{u} + P(\mathbf{e}_\Delta)$.
7: Apply F-relaxation to $\mathcal{A}(\mathbf{u}) = \mathbf{w}$ using Φ.
8: **until** $\|\mathbf{r}\|/\|\mathbf{r}^{(0)}\| \leq \varepsilon$.

Remark 7 (Multilevel MGRIT schemes)**.** While we have only described 2-level MGRIT methods akin to algorithm 2, multilevel MGRIT schemes can be obtained by recursively applying the 2-level schemes, from which we obtain (FAS)-MGRIT V-cycles and FMG implementations analogous to algorithm 3 and algorithm 4, respectively. In particular, for multilevel MGRIT solves, we consider $V(1,1)$-cycles for all the numerical experiments in this thesis.

4.3 MGRIT from a multigrid operator perspective

In this section, we consider the linear MGRIT, algorithm 6, and analyze it from an algebraic perspective. In particular, we define several MGRIT operators, such as the interpolation, restriction and relaxation operators from (2.101), as well as error propagation operators, such as in subsection 2.3.2, associated with the linear MGRIT (parareal) variant described by algorithm 6. Before we do so, we begin by considering an exact two-grid correction scheme, then introducing the parareal and MGRIT methods as two-grid approximations of this exact scheme. Note that the following algebraic theory heavily relies on the existence of Schur complements; see chapter A for a summary.

We recall that the fine grid can be partitioned into C-points, with indices $i = jm$, $j = 0, \dots, N_T$, where $N_T = N_t/m$, and F-points, which are the remaining points with indices $i \neq jm$; note that there are $N_C := N_T + 1$ C-points and $N_F := N_t - N_C$ F-points. Now, we consider reordering the all-at-once system (4.3) first by F-points, then by C-points; this partitioning of time points is similar to red-black partitioning discussed in subsection 2.1.3 for spatial discretizations. Let $\mathbf{u}_i \in \mathbb{R}^N$ for all $i = 0, 1, \dots, N_t$. Then, using subscripts c and f to indicate the sets of C- and F- points, we reorder the solution vector $\mathbf{u} \in \mathbb{R}^{N(N_t+1)}$ from (4.7) into F- and C-point partitions $\mathbf{u}_f$ and $\mathbf{u}_c$, respectively, as

$$\begin{bmatrix} \mathbf{u}_f \\ \mathbf{u}_c \end{bmatrix} \in \mathbb{R}^{N(N_t+1)}, \quad \text{where} \quad \mathbf{u}_f := \begin{bmatrix} \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{m-1} \\ \mathbf{u}_{m+1} \\ \vdots \\ \mathbf{u}_{N_t-1} \end{bmatrix} \in \mathbb{R}^{NN_F}, \quad \mathbf{u}_c := \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_m \\ \vdots \\ \mathbf{u}_{N_T m} \end{bmatrix} \in \mathbb{R}^{NN_C}. \tag{4.28}$$

Similarly, we reorder $\mathcal{A} \in \mathbb{R}^{N(N_t+1)\times N(N_t+1)}$ from (4.7) as

$$\mathcal{A} = \begin{bmatrix} \mathcal{A}_{ff} & \mathcal{A}_{fc} \\ \mathcal{A}_{cf} & \mathcal{A}_{cc} \end{bmatrix} \in \mathbb{R}^{N(N_t+1)\times N(N_t+1)}, \tag{4.29}$$

where $\mathcal{A}_{ff} \in \mathbb{R}^{NN_F \times NN_F}$, $\mathcal{A}_{fc} \in \mathbb{R}^{NNc \times NNf}$, $\mathcal{A}_{fc} \in \mathbb{R}^{NNf \times NNc}$ and $\mathcal{A}_{cc} \in \mathbb{R}^{NNc \times NNc}$ are defined in (4.56) to (4.59). In lieu of this rigorous definition of each block, which we leave for subsection 4.3.1, we consider an illustrative example of this partitioning into sets of F- and C-points.

Example 1 (F- and C-point partitioning)**.** Let $N_t = 6$ and $m = 3$. Then, we have that

$$\mathbf{u}_c = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_3 & \mathbf{u}_6 \end{bmatrix}^T, \quad \mathbf{u}_f = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_4 & \mathbf{u}_5 \end{bmatrix}^T \tag{4.30}$$

are the F- and C-point partitions of $\mathbf{u}$ as in (4.28); similarly, we reorder $\mathbf{w}$, the right-hand side from (4.7), as

$$\mathbf{w}_c = \begin{bmatrix} \mathbf{w}_0 & \mathbf{w}_3 & \mathbf{w}_6 \end{bmatrix}^T, \quad \mathbf{w}_f = \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \mathbf{w}_4 & \mathbf{w}_5 \end{bmatrix}^T \tag{4.31}$$

Reordering $\mathcal{A}$ as in (4.29) and letting I denote the identity operator in $\mathbb{R}^{N\times N}$, we obtain

$$\begin{bmatrix} \mathcal{A}_{ff} & \mathcal{A}_{fc} \\ \mathcal{A}_{cf} & \mathcal{A}_{cc} \end{bmatrix} := \left[\begin{array}{cccc:cccc} I & & & & -\Phi & & & \\ -\Phi & I & & & & & & \\ & & I & & & & -\Phi & \\ & & -\Phi & I & & & & \\ \hdashline & & & & I & & & \\ & -\Phi & & & & I & & \\ & & & -\Phi & & & I & \end{array}\right], \tag{4.32}$$

where the dash lines represent the separation of each block. Note then that

$$\begin{bmatrix} \mathcal{A}_{ff} & \mathcal{A}_{fc} \\ \mathcal{A}_{cf} & \mathcal{A}_{cc} \end{bmatrix} \begin{bmatrix} \mathbf{u}_f \\ \mathbf{u}_c \end{bmatrix} = \begin{bmatrix} \mathbf{w}_f \\ \mathbf{w}_c \end{bmatrix} \tag{4.33}$$

describes exactly the same linear system of equation as $\mathcal{A}\mathbf{u} = \mathbf{w}$ in standard ordering (4.7),

$$\begin{bmatrix} I & & & & & & & \\ -\Phi & I & & & & & & \\ & -\Phi & I & & & & & \\ & & -\Phi & I & & & & \\ & & & -\Phi & I & & & \\ & & & & -\Phi & I & & \\ & & & & & -\Phi & I & \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \\ \mathbf{u}_4 \\ \mathbf{u}_5 \\ \mathbf{u}_6 \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \mathbf{w}_2 \\ \mathbf{w}_3 \\ \mathbf{w}_4 \\ \mathbf{w}_5 \\ \mathbf{w}_6 \end{bmatrix}. \tag{4.34}$$

By performing a block Gaussian elimination on $\mathcal{A}$ (see remark 48), we obtain the following decomposition:

$$\begin{bmatrix} \mathcal{A}_{ff} & \mathcal{A}_{fc} \\ \mathcal{A}_{cf} & \mathcal{A}_{cc} \end{bmatrix} = \begin{bmatrix} I_f & \mathbf{0} \\ \mathcal{A}_{cf}\mathcal{A}_{ff}^{-1} & I_c \end{bmatrix} \begin{bmatrix} \mathcal{A}_{ff} & \mathbf{0} \\ \mathbf{0} & \mathcal{A}_{cc} - \mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc} \end{bmatrix} \begin{bmatrix} I_f & \mathcal{A}_{ff}^{-1}\mathcal{A}_{fc} \\ \mathbf{0} & I_c \end{bmatrix}, \tag{4.35}$$

where $I_c \in \mathbb{R}^{NNc\times NNc}$ and $I_f \in \mathbb{R}^{NN_F\times NN_F}$ are identity operators. We let

$$S = \begin{bmatrix} I_f \\ \mathbf{0} \end{bmatrix}, \tag{4.36}$$

such that

$$\mathcal{A}_{ff} = S^T \mathcal{A} S, \tag{4.37}$$

and define the the "ideal" restriction and interpolation operators, R and P, respectively, as

$$R = \begin{bmatrix} -\mathcal{A}_{cf}\mathcal{A}_{ff}^{-1} & I_c \end{bmatrix}, \tag{4.38}$$

and

$$P = \begin{bmatrix} -\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc} \\ I_c \end{bmatrix}. \tag{4.39}$$

We say that R and P are ideal in the sense that they define the Schur complement of $\mathcal{A}_{ff}$ (see definition 39),

$$\mathcal{A}_{cc} - \mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc} = R\mathcal{A}P \coloneqq \mathcal{A}_\Delta, \tag{4.40}$$

where $\mathcal{A}_\Delta$ is as in (4.10). From (4.35), finding the inverse of $\mathcal{A}$ in terms of the Schur complement $\mathcal{A}_\Delta$ (see remark 47 and/or remark 48) yields the following decomposition of $\mathcal{A}^{-1}$:

$$\mathcal{A}^{-1} = P(R\mathcal{A}P)^{-1}R + S\left(S^T\mathcal{A}S\right)^{-1}S^T. \tag{4.41}$$

Then, since $\mathcal{I} - \mathcal{A}^{-1}\mathcal{A} = \mathbf{0}$, where $\mathcal{I}$ denotes the identity operator in $\mathbb{R}^{N(N_t+1)\times N(N_t+1)}$, we get the multiplicative identity [27]

$$\left(\mathcal{I} - P(R\mathcal{A}P)^{-1}R\mathcal{A}\right)\left(\mathcal{I} - S\left(S^T\mathcal{A}S\right)^{-1}S^T\mathcal{A}\right) = \mathbf{0}, \tag{4.42}$$

where the the first term denotes the error propagation of the ideal coarse-grid correction (i.e., with the Schur complement $\mathcal{A}_\Delta = R\mathcal{A}P$ as the coarse-grid operator) and the second term denotes the error propagation of the F-relaxation. Defining the coarse-grid injection operator

$$R_I = \begin{bmatrix} \mathbf{0} & I_c \end{bmatrix}, \tag{4.43}$$

we note that

$$\left(\mathcal{I} - S\left(S^T\mathcal{A}S\right)^{-1}S^T\mathcal{A}\right) = PR_I. \tag{4.44}$$

The exact coarse-grid correction error propagator can then be written as

$$\left(\mathcal{I} - P(R\mathcal{A}P)^{-1}R\mathcal{A}\right)PR_I = P(I_c - (\mathcal{A}_\Delta)^{-1}\mathcal{A}_\Delta)R_I. \tag{4.45}$$

Parareal as a multigrid reduction method

Instead of solving the coarse-grid problem exactly using $\mathcal{A}_\Delta$, the parareal method considers solving it approximately using $\mathcal{B}_\Delta$, where $\mathcal{B}_\Delta^{-1} \approx \mathcal{A}_\Delta^{-1}$ (4.13). Hence, we replace (4.45) by

$$\begin{aligned}\mathcal{E}_F &\coloneqq P\left(I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta\right) R_I \\ &= \left(\mathcal{I} - P\mathcal{B}_\Delta^{-1}R\mathcal{A}\right) PR_I \\ &= \left(\mathcal{I} - P\mathcal{B}_\Delta^{-1}R\mathcal{A}\right)\left(\mathcal{I} - S\left(S^T\mathcal{A}S\right)^{-1}S^T\mathcal{A}\right),\end{aligned} \tag{4.46}$$

where $\mathcal{E}_F$ denotes the parareal error propagation (or, equivalently, the MGRIT error propagation with F-relaxation). This strategy of replacing the exact Schur complement by approximations, possibly adding relaxations, is a typical feature of Multigrid Reduction (MGR) methods [30, 80], a class of highly efficient multigrid methods for elliptic differential equation problems, such as those discussed in chapter 2. Hence, we may interpret parareal as a 2-level MGR method.

Since $R\mathcal{A}P = R_I\mathcal{A}P$, we can save computational work by instead working with the cheaper R_I instead of R. Then, letting

$$M_F^{-1} \coloneqq S\left(S^T\mathcal{A}S\right)^{-1}S^T = \begin{bmatrix} \mathcal{A}_{ff}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{4.47}$$

and

$$M_{cgc}^{-1} \coloneqq P\mathcal{B}_\Delta^{-1}R_I = \begin{bmatrix} -\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc} \\ I_c \end{bmatrix} \mathcal{B}_\Delta^{-1} \begin{bmatrix} \mathbf{0} & I_c \end{bmatrix} = \begin{bmatrix} \mathbf{0} & -\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\mathcal{B}_\Delta^{-1} \\ \mathbf{0} & \mathcal{B}_\Delta^{-1} \end{bmatrix}, \tag{4.48}$$

we have that

$$\mathcal{I} - P\mathcal{B}_\Delta^{-1}R_I\mathcal{A} = \mathcal{I} - M_{cgc}^{-1}\mathcal{A} \tag{4.49}$$

and

$$\mathcal{I} - S\left(S^T\mathcal{A}S\right)^{-1}S^T\mathcal{A} = \mathcal{I} - M_F^{-1}\mathcal{A}, \tag{4.50}$$

such that the approximate coarse-grid correction and the error propagation of the F-relaxation both recover the classical iterative fixed-point method form $I - M^{-1}\mathcal{A}$. Then, we can write $\mathcal{E}_F$ (4.46) as

$$\begin{aligned}\mathcal{E}_F &= (\mathcal{I} - M_{cgc}^{-1}\mathcal{A})(\mathcal{I} - M_F^{-1}\mathcal{A}) \\ &= P(I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)R_I \\ &= P\begin{bmatrix} \mathbf{0} & I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{0} & -\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\left(I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta\right) \\ \mathbf{0} & I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta \end{bmatrix}.\end{aligned} \tag{4.51}$$

MGRIT as a multigrid reduction method

MGRIT follows from parareal by replacing the F-relaxation with an FCF-relaxation. Therefore, we replace the F-relaxation error propagator, $\mathcal{I} - M_F^{-1}\mathcal{A} = \mathcal{I} - S\left(S^T\mathcal{A}S\right)^{-1}S^T\mathcal{A}$, by the FCF-relaxation error propagator, which is the product of the F-, C- and F-relaxation error propagators in this order:

$$\begin{aligned}&\left(\mathcal{I} - S\left(S^T\mathcal{A}S\right)^{-1}S^T\mathcal{A}\right)\left(\mathcal{I} - R_I^T\left(R_I\mathcal{A}R_I^T\right)^{-1}R_I\mathcal{A}\right)\left(\mathcal{I} - S\left(S^T\mathcal{A}S\right)^{-1}S^T\mathcal{A}\right)\\ &= (\mathcal{I} - M_F^{-1}\mathcal{A})(\mathcal{I} - M_C^{-1}\mathcal{A})(\mathcal{I} - M_F^{-1}\mathcal{A}),\end{aligned} \tag{4.52}$$

where we define

$$M_C^{-1} \coloneqq R_I^T\left(R_I\mathcal{A}R_I^T\right)^{-1}R_I = \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathcal{A}_{cc}^{-1}\end{bmatrix}. \tag{4.53}$$

Since $\left(\mathcal{I} - S\left(S^T\mathcal{A}S\right)^{-1}S^T\mathcal{A}\right) = PR_I$ and $\mathcal{A}_{cc} = I_c$ (4.59), we can write the error propagation of the FCF-relaxation as

$$P\left(I - \mathcal{A}_{cc}^{-1}\left(\mathcal{A}_{cc} - \mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\right)\right)R_I = P\left(I - \mathcal{A}_\Delta\right)R_I, \tag{4.54}$$

from which we observe that FCF-relaxation corresponds to Jacobi smoothing on the coarse time grid with the exact (Schur complement) coarse-grid operator, $\mathcal{A}_\Delta = R\mathcal{A}P$. Then, the MGRIT error propagator (with FCF-relaxation) is given by

$$\begin{aligned}\mathcal{E}_{FCF} &= (\mathcal{I} - M_{cgc}^{-1}\mathcal{A})(\mathcal{I} - M_F^{-1}\mathcal{A})(I - M_C^{-1}\mathcal{A})(\mathcal{I} - M_F^{-1}\mathcal{A})\\ &= \mathcal{E}_F(\mathcal{I} - M_C^{-1}\mathcal{A})(\mathcal{I} - M_F^{-1}\mathcal{A})\\ &= \begin{bmatrix}0 & -\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\left(I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta\right)\mathcal{A}_{cc}^{-1}\mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\\ 0 & \left(I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta\right)\mathcal{A}_{cc}^{-1}\mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\end{bmatrix}\\ &= P\begin{bmatrix}0 & \left(I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta\right)\mathcal{A}_{cc}^{-1}\mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\end{bmatrix}\\ &= P\left(\left(I_c - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta\right)\mathcal{A}_{cc}^{-1}\mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\right)R_I.\end{aligned} \tag{4.55}$$

Note that this also describes a two-level MGR method.

4.3.1 The MGRIT matrices

In the following, we formally define the matrices that appeared in the previous section as derived in [89], in terms of which we define the MGRIT error propagation operators: $\mathcal{E}_F$

(4.51) for MGRIT with F-relaxation (i.e., parareal) and $\mathcal{E}_{FCF}$ (4.55) for MGRIT with FCF-relaxation. We begin by defining $\mathcal{A}_{ff}$, $\mathcal{A}_{fc}$, $\mathcal{A}_{cf}$ and $\mathcal{A}_{cc}$ matrices from (4.29), with the dashed lines outlining the matrix blocks that operate either on single C-points or groups of $m-1$ F-points, in terms of the operators $\mathbf{0}, \Phi, I \in \mathbb{R}^{N\times N}$:

$$\mathcal{A}_{ff} = \left[\begin{array}{cccc:c:cccc} I & & & & & & & & \\ -\Phi & I & & & & & & & \\ & \ddots & \ddots & & & & & & \\ & & -\Phi & I & & & & & \\ \hdashline & & & & \ddots & & & & \\ \hdashline & & & & & I & & & \\ & & & & & -\Phi & I & & \\ & & & & & & \ddots & \ddots & \\ & & & & & & & -\Phi & I \end{array}\right], \tag{4.56}$$

$$\mathcal{A}_{fc} = \left[\begin{array}{ccc:c} -\Phi & & & \mathbf{0} \\ \mathbf{0} & & & \vdots \\ \vdots & & & \vdots \\ \mathbf{0} & & & \mathbf{0} \\ \hdashline & \ddots & & \vdots \\ \hdashline & & -\Phi & \mathbf{0} \\ & & \mathbf{0} & \vdots \\ & & \vdots & \vdots \\ & & \mathbf{0} & \mathbf{0} \end{array}\right], \tag{4.57}$$

$$\mathcal{A}_{cf} = \left[\begin{array}{ccccc:c:cccc} \mathbf{0} & \dots & & & & & & & \\ \mathbf{0} & \dots & \mathbf{0} & -\Phi & & & & & \\ \hdashline & & & & \ddots & & & & \\ \hdashline & & & & & \mathbf{0} & \dots & \mathbf{0} & -\Phi \end{array}\right], \tag{4.58}$$

$$\mathcal{A}_{cc} = \left[\begin{array}{c:c:c:c} I & & & \\ \hdashline & I & & \\ \hdashline & & \ddots & \\ \hdashline & & & I \end{array}\right]. \tag{4.59}$$

By computing simple matrix inverses and products, we obtain the remaining matrices from (4.35):

$$(\mathcal{A}_{ff})^{-1} = \left[\begin{array}{cccc:c:cccc} I & & & & & & & & \\ \Phi & I & & & & & & & \\ \vdots & \ddots & \ddots & & & & & & \\ \Phi^{m-2} & \dots & \Phi & I & & & & & \\ \hdashline & & & & \ddots & & & & \\ \hdashline & & & & & I & & & \\ & & & & & \Phi & I & & \\ & & & & & \vdots & \ddots & \ddots & \\ & & & & & \Phi^{m-2} & \dots & \Phi & I \end{array}\right], \tag{4.60}$$

$$-\mathcal{A}_{cf}(\mathcal{A}_{ff})^{-1} = \left[\begin{array}{cccc:c:cccc} \mathbf{0} & \dots & & & & & & & \\ \hdashline \Phi^{m-1} & \Phi^{m-2} & \dots & \Phi & & & & & \\ \hdashline & & & & \ddots & & & & \\ \hdashline & & & & & \Phi^{m-1} & \Phi^{m-2} & \dots & \Phi \end{array}\right], \tag{4.61}$$

$$-(\mathcal{A}_{ff})^{-1}\mathcal{A}_{fc} = \left[\begin{array}{ccc:c:ccc:c} \Phi & & & \mathbf{0} & & & & \\ \Phi^2 & & & \vdots & & & & \\ \vdots & & & \vdots & & & & \\ \Phi^{m-1} & & & \mathbf{0} & & & & \\ \hdashline & \ddots & & \vdots & & & & \\ \hdashline & & \Phi & \mathbf{0} & & & & \\ & & \Phi^2 & \vdots & & & & \\ & & \vdots & \vdots & & & & \\ & & \Phi^{m-1} & \mathbf{0} & & & & \end{array}\right], \tag{4.62}$$

$$\mathcal{A}_{cf}(\mathcal{A}_{ff})^{-1}\mathcal{A}_{fc} = \left[\begin{array}{c:c:c:c} \mathbf{0} & & & \\ \hdashline \Phi^m & \mathbf{0} & & \\ \hdashline & \ddots & \ddots & \\ \hdashline & & \Phi^m & \mathbf{0} \end{array}\right]. \tag{4.63}$$

The Schur complement of $\mathcal{A}_{ff}$ (see definition 39), $\mathcal{A}_\Delta = R\mathcal{A}P$, is then given by

$$\mathcal{A}_\Delta = \mathcal{A}_{cc} - \mathcal{A}_{cf}\left(\mathcal{A}_{ff}\right)^{-1}\mathcal{A}_{fc} = \begin{bmatrix} I & & & \\ -\Phi^m & I & & \\ & \ddots & \ddots & \\ & & -\Phi^m & I \end{bmatrix}, \tag{4.64}$$

which matches its definition in (4.10).

Recall that we approximate of $\mathcal{A}_\Delta^{-1}$ in $\mathcal{I} - \mathcal{A}_\Delta^{-1}\mathcal{A}_\Delta = \mathbf{0}$ by some cheaper approximation $\mathcal{B}_\Delta^{-1}$ (4.13), in terms of which we obtain the parareal error propagator $\mathcal{E}_F = P\left(I - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta\right)R_I$ (4.46). By (4.10) and (4.13), we obtain

$$\begin{aligned} \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta &= \begin{bmatrix} I & & & & \\ \Psi & I & & & \\ \Psi^2 & \Psi & I & & \\ \vdots & \vdots & \ddots & I & \\ \Psi^{N_T} & \Psi^{N_T-1} & \dots & \Psi & I \end{bmatrix} \begin{bmatrix} I & & & \\ -\Phi^m & I & & \\ & \ddots & \ddots & \\ & & -\Phi^m & I \end{bmatrix} \\ &= \begin{bmatrix} I & I & & & \\ \Psi - \Phi^m & \Psi - \Phi^m & I & & \\ \Psi\left(\Psi - \Phi^m\right) & \vdots & \ddots & \ddots & \\ \vdots & \Psi^{N_T-2}\left(\Psi - \Phi^m\right) & \dots & \Psi - \Phi^m & I \end{bmatrix}, \end{aligned} \tag{4.65}$$

from which it follows that

$$I - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta = \begin{bmatrix} \mathbf{0} & & & & \\ I & \mathbf{0} & & & \\ \Psi & I & \mathbf{0} & & \\ \vdots & \vdots & \ddots & \ddots & \\ \Psi^{N_T-1} & \Psi^{N_T-2} & \dots & I & \mathbf{0} \end{bmatrix} \operatorname{diag}\left(\Phi^m - \Psi\right). \tag{4.66}$$

Similarly, we find

$$I - \mathcal{A}_\Delta\mathcal{B}_\Delta^{-1} = \operatorname{diag}\left(\Phi^m - \Psi\right) \begin{bmatrix} \mathbf{0} & & & & \\ I & \mathbf{0} & & & \\ \Psi & I & \mathbf{0} & & \\ \vdots & \vdots & \ddots & \ddots & \\ \Psi^{N_T-1} & \Psi^{N_T-2} & \dots & I & \mathbf{0} \end{bmatrix}, \tag{4.67}$$

where $I - \mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta = I - \mathcal{A}_\Delta\mathcal{B}_\Delta^{-1}$ if Φ and Ψ commute.

4.4 MGRIT convergence theory under a diagonalizability assumption

4.4.1 The MGRIT error propagation operators in block-matrix form

Recall from subsection 2.3.2 that for an iterative method with error propagation matrix $\mathcal{E}$, we have that the error after k iterations, $\mathbf{e}^{(k)}$, is given by

$$\mathbf{e}^{(k)} = \mathcal{E}^k \mathbf{e}^{(0)}; \tag{4.68}$$

note that unlike in (2.63), we take this to be error propagation of the entire MGRIT algorithm, not just the relaxation scheme. Let $\mathbf{v}_i^{(k)}$ denote the approximation obtained after the kth consecutive MGRIT iteration for the exact solution vector $\mathbf{u}_i$, where $i = 0, 1, \ldots, N_t$ denotes the fine-grid time point. Then, we define the kth MGRIT iteration error vector as

$$\mathbf{e}^{(k)} \coloneqq \begin{bmatrix} \mathbf{e}_0^{(k)} \\ \mathbf{e}_1^{(k)} \\ \vdots \\ \mathbf{e}_{N_t}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{u}_0 - \mathbf{v}_0^{(k)} \\ \mathbf{u}_1 - \mathbf{v}_1^{(k)} \\ \vdots \\ \mathbf{u}_{N_t} - \mathbf{v}_{N_t}^{(k)} \end{bmatrix}. \tag{4.69}$$

Following the same F- and C-point partition as in (4.28), we reorder the error vector $\mathbf{e}^{(k)}$ for $k = 0, 1, \ldots$ as

$$\begin{bmatrix} \mathbf{e}_f^{(k)} \\ \mathbf{e}_c^{(k)} \end{bmatrix}, \quad \text{where} \quad \mathbf{e}_f^{(k)} \coloneqq \begin{bmatrix} \mathbf{e}_1^{(k)} \\ \vdots \\ \mathbf{e}_{m-1}^{(k)} \\ \mathbf{e}_{m+1}^{(k)} \\ \vdots \\ \mathbf{e}_{N_t-1}^{(k)} \end{bmatrix}, \quad \mathbf{e}_c \coloneqq \begin{bmatrix} \mathbf{e}_0^{(k)} \\ \mathbf{e}_m^{(k)} \\ \vdots \\ \mathbf{e}_{N_t}^{(k)} \end{bmatrix}. \tag{4.70}$$

Note that applying the injection restriction operator R_I (4.43) to a the reordered error vector as in (4.70) yields

$$R_I \mathbf{e}^{(k)} = \begin{bmatrix} \mathbf{0} & I_c \end{bmatrix} \begin{bmatrix} \mathbf{e}_f^{(k)} \\ \mathbf{e}_c^{(k)} \end{bmatrix} = \mathbf{e}_c^{(k)} \tag{4.71}$$

such that the error after one MGRIT iteration with F-relaxation (i.e., parareal) is given by

$$\begin{aligned}\begin{bmatrix}\mathbf{e}_f^{(1)}\\ \mathbf{e}_c^{(1)}\end{bmatrix} &= \mathcal{E}_F \begin{bmatrix}\mathbf{e}_f^{(0)}\\ \mathbf{e}_c^{(0)}\end{bmatrix}\\ &= P(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)R_I\mathbf{e}^{(0)}\\ &= P(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)\mathbf{e}_c^{(0)}\\ &= \begin{bmatrix} -\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)\mathbf{e}_c^{(0)}\\ (I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)\mathbf{e}_c^{(0)}\end{bmatrix}.\end{aligned} \tag{4.72}$$

Then, note that the error after a second iteration is simply

$$\begin{aligned}\begin{bmatrix}\mathbf{e}_f^{(2)}\\ \mathbf{e}_c^{(2)}\end{bmatrix} &= \mathcal{E}_F \begin{bmatrix}\mathbf{e}_f^{(1)}\\ \mathbf{e}_c^{(1)}\end{bmatrix}\\ &= P(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)R_I\mathbf{e}^{(1)}\\ &= P(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)R_I\begin{bmatrix} -\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)\mathbf{e}_c^{(0)}\\ (I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)\mathbf{e}_c^{(0)}\end{bmatrix}\\ &= P(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)^2\mathbf{e}_c^{(0)}.\end{aligned} \tag{4.73}$$

Therefore, by repeatedly applying $\mathcal{E}_F$ to $\mathbf{e}_0$, we obtain

$$\begin{bmatrix}\mathbf{e}_f^{(k)}\\ \mathbf{e}_c^{(k)}\end{bmatrix} = \mathcal{E}_F^k \begin{bmatrix}\mathbf{e}_f^{(0)}\\ \mathbf{e}_c^{(0)}\end{bmatrix} = P(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta)^k\mathbf{e}_c^{(0)}. \tag{4.74}$$

Similarly, for MGRIT with FCF-relaxation, we get

$$\begin{bmatrix}\mathbf{e}_f^{(k)}\\ \mathbf{e}_c^{(k)}\end{bmatrix} = \mathcal{E}_{FCF}^k \begin{bmatrix}\mathbf{e}_f^{(0)}\\ \mathbf{e}_c^{(0)}\end{bmatrix} = P\left(\left(I-\mathcal{B}_\Delta^{-1}\mathcal{A}_\Delta\right)\mathcal{A}_{cc}^{-1}\mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}\right)^k\mathbf{e}_c^{(0)}. \tag{4.75}$$

Note that for both relaxation schemes, the error after k iterations depends only on a propagation of the initial error at C-points, $\mathbf{e}_c^{(0)}$. Therefore, we can restrict the MGRIT error propagation analysis to the C-points only, knowing that the C-point error is propagated to the F-points by P. Then, it is convenient to redefine the error propagation operators $\mathcal{E}_F$ and $\mathcal{E}_{FCF}$ in terms of their action with respect to the error at C-points:

$$\mathcal{E}_F^{[c]} \coloneqq I - B_\Delta^{-1}\mathcal{A}_\Delta \tag{4.76}$$

and

$$\mathcal{E}_{FCF}^{[c]} \coloneqq \left(\left(I - \mathcal{B}_{\Delta}^{-1}\mathcal{A}_{\Delta}\right)\mathcal{A}_{cc}^{-1}\mathcal{A}_{cf}\mathcal{A}_{ff}^{-1}\mathcal{A}_{fc}, \tag{4.77}$$

such that for F-relaxation we have

$$\mathbf{e}_c^{(k)} = \left(\mathcal{E}_F^{[c]}\right)^k \mathbf{e}_c^{(0)} \tag{4.78}$$

and, for FCF-relaxation,

$$\mathbf{e}_c^{(k)} = \left(\mathcal{E}_{FCF}^{[c]}\right)^k \mathbf{e}_c^{(0)} \tag{4.79}$$

with respect to the definitions of $\mathcal{E}_F$ as in (4.76) for the former and of $\mathcal{E}_{FCF}$ as in (4.77) for the latter. For our following analysis, it also useful to write the error propagation operators explicitly in block-matrix form. From (4.76) and (4.66), we have that

$$\mathcal{E}_F^{[c]} = \begin{bmatrix} \mathbf{0} & & & & \\ I & \mathbf{0} & & & \\ \Psi & I & \mathbf{0} & & \\ \vdots & \vdots & \ddots & \ddots & \\ \Psi^{N_T-1} & \Psi^{N_T-2} & \cdots & I & \mathbf{0} \end{bmatrix} \operatorname{diag}\left(\Phi^m - \Psi\right). \tag{4.80}$$

Then, by (4.76) and (4.63), after some basic computations we find

$$\mathcal{E}_{FCF}^{[c]} = \begin{bmatrix} \mathbf{0} & & & & & \\ \mathbf{0} & \mathbf{0} & & & & \\ I & \mathbf{0} & \mathbf{0} & & & \\ \Psi & I & \mathbf{0} & \mathbf{0} & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ \Psi^{N_T-2} & \cdots & \cdots & I & \mathbf{0} & \mathbf{0} \end{bmatrix} \operatorname{diag}((\Phi^m - \Psi)\Phi^m). \tag{4.81}$$

In particular, we note that $\mathcal{E}_F^{[c]}$ and $\mathcal{E}_{FCF}^{[c]}$ are lower-triangular Toeplitz block-matrix operators. Many convergence properties of MGRIT rely on Toeplitz operator theory; see [89]. We define Toeplitz (block) matrices below:

Definition 37 (Toeplitz (block) matrix)**.** *Let $M \in \mathbb{R}^{n\times n}$. We say that A is a Toeplitz*

matrix if

$$A = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-(n-1)} \\ a_1 & a_0 & a_{-1} & \ddots & & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & & \ddots & a_1 & a_0 & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}. \tag{4.82}$$

That is, A is Toeplitz if the i, j entries of A, denoted by $A_{i,j}$, satisfy

$$A_{i,j} = A_{i+1,j-1} = a_{i-j}. \tag{4.83}$$

More compactly, we can say that A is the Toeplitz matrix defined by

$$\{a_{-(n-1)}, a_{-(n-2)}, \ldots, a_{-1}, a_0, a_1, \ldots, a_{n-2}, a_{n-1}\}. \tag{4.84}$$

Similarly, we say that $A \in \mathbb{R}^{nm \times nm}$ is a Toeplitz block matrix (or block-Toeplitz) if

$$A = \begin{bmatrix} A_0 & A_{-1} & A_{-2} & \cdots & \cdots & A_{-(n-1)} \\ A_1 & A_0 & A_{-1} & \ddots & & \vdots \\ A_2 & A_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & A_{-1} & A_{-2} \\ \vdots & & \ddots & A_1 & A_0 & A_{-1} \\ A_{n-1} & \cdots & \cdots & A_2 & A_1 & A_0 \end{bmatrix}, \tag{4.85}$$

where $A_{i,j} \in \mathbb{R}^{m \times m}$ satisfy

$$A_{i,j} = A_{i+1,j-1} = A_{i-j}. \tag{4.86}$$

Then, we can say that A is the Toeplitz block-matrix defined by

$$\{A_{-(n-1)}, A_{-(n-2)}, \ldots, A_{-1}, A_0, A_1, \ldots, A_{n-2}, A_{n-1}\}. \tag{4.87}$$

4.4.2 Error propagation of one MGRIT iteration

In the following, we present the MGRIT convergence analysis MGRIT, which follows from analyzing the error propagation of one MGRIT iteration. This analysis, first presented in [24], originally follows from the algorithmic description of MGRIT as in section 4.2. Here,

we present it more directly from the previously defined error propagation operators. From this point onwards, for brevity we restrict our analysis to MGRIT with FCF-relaxation, which is the typical MGRIT configuration and what we consider for our numerical experiments. However, we note that a very similar analysis applies to MGRIT with F-relaxation (i.e., parareal), only replacing the work of $\mathcal{E}_{FCF}$ by $\mathcal{E}_F$. For a more economic notation, henceforth we denote $\mathcal{E} \coloneqq \mathcal{E}_{FCF}^{[c]}$ as in (4.81):

$$\mathcal{E} \coloneqq \begin{bmatrix} \mathbf{0} & & & & & \\ \mathbf{0} & \mathbf{0} & & & & \\ I & \mathbf{0} & \mathbf{0} & & & \\ \Psi & I & \mathbf{0} & \mathbf{0} & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ \Psi^{N_T-2} & \cdots & \cdots & I & \mathbf{0} & \mathbf{0} \end{bmatrix} \operatorname{diag}((\Phi^m - \Psi)\Phi^m). \tag{4.88}$$

Given the observation that $\mathcal{E}$ is a block-Toeplitz operator, we can compactly describe $\mathcal{E}$ by its $(\mathcal{E})_{i,j}$ block entries, where $i = 0, \ldots, N_t$ and $j = 0, \ldots, N_t$ denote the row and column indices, respectively. Then, the entries of $\mathcal{E}$ (4.77) are given by

$$(\mathcal{E})_{i,j} = \begin{cases} \Psi^{i-j-2}(\Phi^m - \Psi)\Phi^m & \text{if } i - j \geq 2, \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{4.89}$$

Note that $\ell \coloneqq i - j$ ranges from 0 to N_T, by which (4.89) implies that there are only $N_T + 1$ reoccurring blocks in $\mathcal{E}$, which we call the Toeplitz blocks of $\mathcal{E}$ and denote by $\mathcal{E}_l$ for $\ell = 0, \ldots, N_T$. In particular, we define the $\mathcal{E}_\ell$ Toeplitz blocks in terms of the last row of $\mathcal{E}$,

$$\{\mathcal{E}_0, \mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_{N_T}\} \coloneqq \{(\mathcal{E})_{N_T,N_T}, (\mathcal{E})_{N_T,N_T-1}, (\mathcal{E})_{N_T,N_T-2}, \ldots, (\mathcal{E})_{N_T,0}, \}, \tag{4.90}$$

such that we represent $\mathcal{E}$ in (lower-triangular) Toeplitz form as

$$\mathcal{E} = \begin{bmatrix} \mathcal{E}_0 & & & & & \\ \mathcal{E}_1 & \mathcal{E}_0 & & & & \\ \mathcal{E}_2 & \mathcal{E}_1 & \mathcal{E}_0 & & & \\ \mathcal{E}_3 & \mathcal{E}_2 & \mathcal{E}_1 & \mathcal{E}_0 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ \mathcal{E}_{N_T} & \cdots & \cdots & \mathcal{E}_2 & \mathcal{E}_1 & \mathcal{E}_0 \end{bmatrix}. \tag{4.91}$$

Recall from (4.79) that

$$\mathbf{e}_c^{(1)} = \mathcal{E}\mathbf{e}_c^{(0)}, \tag{4.92}$$

or, for any pair of subsequent MGRIT iterations $k-1$ and k, for $k \geq 1$,

$$\mathbf{e}_c^{(k)} = \mathcal{E}\mathbf{e}_c^{(k-1)}, \tag{4.93}$$

where $\mathcal{E}$ is defined in (4.89). Expanding the error block vectors, we have that

$$\begin{bmatrix} \mathbf{e}_0^{(k)} \\ \mathbf{e}_m^{(k)} \\ \mathbf{e}_{2m}^{(k)} \\ \mathbf{e}_{3m}^{(k)} \\ \vdots \\ \mathbf{e}_{N_T m}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathcal{E}_0 & & & & & \\ \mathcal{E}_1 & \mathcal{E}_0 & & & & \\ \mathcal{E}_2 & \mathcal{E}_1 & \mathcal{E}_0 & & & \\ \mathcal{E}_3 & \mathcal{E}_2 & \mathcal{E}_1 & \mathcal{E}_0 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ \mathcal{E}_{N_T} & \cdots & \cdots & \mathcal{E}_2 & \mathcal{E}_1 & \mathcal{E}_0 \end{bmatrix} \begin{bmatrix} \mathbf{e}_0^{(k-1)} \\ \mathbf{e}_m^{(k-1)} \\ \mathbf{e}_{2m}^{(k-1)} \\ \mathbf{e}_{3m}^{(k-1)} \\ \vdots \\ \mathbf{e}_{N_T m}^{(k-1)} \end{bmatrix}, \tag{4.94}$$

Hence, noting that the above together with (4.90) implies

$$\mathbf{e}_{im}^{(k)} = \sum_{j=0}^{N_T} (\mathcal{E})_{i,j} \mathbf{e}_{jm}^{(k-1)}, \quad i = 2, \ldots, N_T, \tag{4.95}$$

and $\mathbf{e}_0^{(k)} = \mathbf{e}_m^{(k)} = \mathbf{0}$, by (4.89) we obtain the following lemma.

Lemma 2 (C-point errors after one MGRIT iteration)**.** *[24, Lemma 3.2] After one iteration of MGRIT with FCF-relaxation, the approximate solution at the ith C-point satisfies* $\mathbf{v}_{im}^{(k)} = \mathbf{u}_{im} - \mathbf{e}_{im}^{(k)}$*, where the error at the C-points satisfies*

$$\mathbf{e}_0^{(k)} = \mathbf{e}_m^{(k)} = \mathbf{0}, \tag{4.96}$$

$$\mathbf{e}_{im}^{(k)} = \sum_{j=0}^{i-2k} \Psi^{i-j-2}(\Phi^m - \Psi)\Phi^m \mathbf{e}_{jm}^{(k-1)}, \quad i = 2k, \ldots, N_T. \tag{4.97}$$

4.4.3 MGRIT convergence for diagonalizable operators

Now, we assume that the fine- and coarse-grid operators, $\Phi \in \mathbb{R}^{N \times N}$ and $\Psi \in \mathbb{R}^{N \times N}$, respectively, can be diagonalized by the same unitary transformation

$$Q = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_N \end{bmatrix}, \quad Q^*Q = I, \tag{4.98}$$

where the column vectors $\mathbf{q}_n$, $n = 1, \ldots, N$ form an orthonormal basis of eigenvectors for the eigenvalues of Φ and Ψ. That is, we can unitarily diagonalize Φ and Ψ by the same Q, such that

$$\Phi = Q\,\mathrm{diag}(\phi_1, \ldots, \phi_N)Q^*, \quad \Psi = Q\,\mathrm{diag}(\psi_1, \ldots, \psi_N)Q^*, \tag{4.99}$$

with ϕ_n and ψ_n denoting the n-th eigenvalues of Φ and Ψ, respectively. We further assume that Φ and Ψ are stable operators, by which we mean that $\rho(\Psi) < 1$ and $\rho(\Phi) < 1$. Then, we can expand the errors in terms of these eigenvectors:

$$\mathbf{e}_i^{(k)} = \sum_{n=1}^{N} e_{i,n}^{(k)} \mathbf{q}_n, \quad i = 0, 1, \ldots, N_t, \quad k = 0, 1, \ldots. \tag{4.100}$$

In particular, we consider expanding the errors at the C-points, $i = jm$ for $j = 0, 1, \ldots, N_T$. Then, for a given mode $n = 1, \ldots, N$, we denote the scalar C-point error vector (i.e., the nth coordinates of the error at the C-points with respect to the eigenbasis given by the columns of Q)

$$\widehat{\mathbf{e}}_n^{(k)} = \begin{pmatrix} e_{0,n}^{(k)} \\ e_{m,n}^{(k)} \\ \vdots \\ e_{N_T m,n}^{(k)} \end{pmatrix} = \begin{pmatrix} \mathbf{q}_n^* \mathbf{e}_0^{(k)} \\ \mathbf{q}_n^* \mathbf{e}_m^{(k)} \\ \vdots \\ \mathbf{q}_n^* \mathbf{e}_{N_T m}^{(k)} \end{pmatrix}, \quad n = 1, \ldots, N. \tag{4.101}$$

Then, from lemma 2 we get

$$e_{0,n}^{(k)} = e_{m,n}^{(k)} = 0, \tag{4.102}$$

$$e_{im,n}^{(k)} = \sum_{j=0}^{i-2k} \psi_n^{i-j-2} (\phi_n^m - \psi_n) \phi_n^m e_{jm,n}^{(k-1)}, \quad i = 2k, \ldots, N_T, \quad n = 1, \ldots, N. \tag{4.103}$$

We can describe this relation in matrix form as

$$\widehat{\mathbf{e}}_n^{(k)} = \mathcal{E}_n \widehat{\mathbf{e}}_n^{(k-1)}, \quad n = 1, \ldots, N \tag{4.104}$$

where $\mathcal{E}_n$, the error propagation matrix for the mode n, is given by

$$\mathcal{E}_n \coloneqq \begin{bmatrix} 0 & & & & & \\ 0 & 0 & & & & \\ 1 & 0 & 0 & & & \\ \psi_n & 1 & 0 & 0 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ \psi_n^{N_T-2} & \cdots & \cdots & 1 & 0 & 0 \end{bmatrix} \operatorname{diag}((\phi_n^m - \psi_n)\phi_n^m), \quad n = 1, \ldots, N. \tag{4.105}$$

Note that $\mathcal{E}_n$ is a lower-triangular Toeplitz matrix (see definition 37) for all $n = 1, \ldots, N_T$. In particular, note that

$$\|\mathcal{E}_n\|_1 = \|\mathcal{E}_n\|_\infty = |\phi_n^m - \psi_n| \, |\phi_n^m| \sum_{j=0}^{N_T-2} |\psi_n|^j = |\phi_n^m - \psi_n| \, |\phi_n^m| \frac{\left(1 - |\psi_n|^{N_T-1}\right)}{(1 - |\psi_n|)}, \tag{4.106}$$

where the maximum absolute row sum (which gives $\|\mathcal{E}\|_\infty$) is attained by the last row, corresponding to the row index $i = N_t$, while the maximum absolute column sum (which gives $\|\mathcal{E}\|_1$) is attained by the first column, corresponding to the column index $j = 0$. Then, it follows that

$$\|\mathcal{E}_n\|_2 \leq \sqrt{\|\mathcal{E}_n\|_1 \|\mathcal{E}_n\|_\infty} = \|\mathcal{E}_n\|_\infty = |\phi_n^m - \psi_n|\,|\phi_n^m| \frac{\left(1 - |\psi_n|^{N_T - 1}\right)}{(1 - |\psi_n|)}. \tag{4.107}$$

This yields the following fundamental theorem for analyzing MGRIT convergence.

Fact 43. *[24, Theorem 3.3] Let Φ and Ψ be simultaneously diagonalizable by a unitary transformation Q, with eigenvalues ϕ_n and ψ_n, respectively. Furthermore, assume that both time-stepping operators are stable, i.e., $|\phi_n| < 1$ and $|\psi_n| < 1$. Then the global space-time error vector at the C-points $\mathbf{e}_c = \left[\mathbf{e}_0^T, \mathbf{e}_m^T, \mathbf{e}_{2m}^T, \ldots, \mathbf{e}_{N_T m}^T\right]^T$ of MGRIT with FCF-relaxation satisfies*

$$\|\mathcal{E}\mathbf{e}_c\|_2 \leq \max_{1 \leq n \leq N} \|\mathcal{E}_n\|_\infty \|\mathbf{e}_c\|_2 = \max_{1 \leq n \leq N} \left\{ |\phi_n^m - \psi_n|\,|\phi_n^m| \frac{\left(1 - |\psi_n|^{N_T - 1}\right)}{(1 - |\psi_n|)} \right\} \|\mathbf{e}_c\|_2 \tag{4.108}$$

Note that fact 43 implies that MGRIT convergence analysis can be simplified to estimating the error convergence for the worst-case eigenmode, where the error reduction factor of each eigenmode can be computed independently. We note that a similar result to fact 43 for 2-level MGRIT with F-relaxation (parareal) can be similarly obtained; as shown in [24], the corresponding bound is the same as (4.108) except for lacking the $|\phi_n^m|$ in the term inside the curly brackets. We also note that in practice, since N_T is typically sufficiently large such that $|\psi_n|^{N_T - 1} \approx 0$ for all $n = 1, \ldots, N$, we often replace $\|\mathcal{E}_n\|_\infty$ by its upper bound,

$$\|\mathcal{E}_n\|_\infty = |\phi_n^m - \psi_n|\,|\phi_n^m| \frac{\left(1 - |\psi_n|^{N_T - 1}\right)}{(1 - |\psi_n|)} \leq \frac{|\phi_n^m - \psi_n|\,|\phi_n^m|}{(1 - |\psi_n|)}, \tag{4.109}$$

where

$$\lim_{N_T \to +\infty} \|\mathcal{E}_n\|_\infty = \frac{|\phi_n^m - \psi_n|\,|\phi_n^m|}{(1 - |\psi_n|)}. \tag{4.110}$$

Thus, (4.110) provides a characterization of the sense in which the $\Psi \approx \Phi^m$ approximation must behave in terms of the asymptotic error propagation infinity norm. Intuitively, it suggests that we need $\phi_n^m \approx \psi_n$ to be a sufficiently accurate for all N eigenvalues of Φ and Ψ for MGRIT to converge satisfactorily; in particular, Φ^m must approximate Ψ very accurately for eigenvectors of Ψ corresponding to larger eigenvalues, and less accurately for smaller eigenvalues. We refer to [89, 54] for more general MGRIT convergence results.

Chapter 5

Parallel-in-iteration optimization

In this chapter we outline our parallel-in-iteration (P-in-I) strategy for the solution of discrete optimization problems mentioned in chapter 1 and first presented in [3]. In general, the problems we consider here are the two optimization problems discussed in chapter 3. We recall that the first problem, discussed in section 3.2, is that of minimizing a real-valued (hence, proper) convex and L-smooth function f:

$$\min_{\mathbf{u}\in\mathbb{R}^N} f(\mathbf{u}). \tag{5.1}$$

The second problem, discussed in section 3.3, is that of minimizing the sum of functions $f+g$, where f is convex and L-smooth, and g is continuous, convex and possibly nondifferentiable:

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad F(\mathbf{u}), \quad \text{where } F(\mathbf{u}) = f(\mathbf{u}) + g(\mathbf{u}). \tag{5.2}$$

As outlined in chapter 1 and discussed in chapter 3, the motivation for parallelizing over iteration is that convergence rates of the classical methods for these problems, namelyGradient Descent (GD) (3.49) for (5.1) and Proximal Gradient Descent (PG) (3.84) for (5.2), tend to be slow. Thus, in practice, they typically require many iterations to reach convergence.

In the context of discrete optimization methods, the *causality principle* [73] discussed in chapter 4 means that the solution at iteration k, $\mathbf{u}_k$, is needed to compute the updated solution at iteration $k+1$, $\mathbf{u}_{k+1}$, as is evident from the update formulas of the optimization methods described in chapter 3. We recall from the discussion in chapter 4 that many parallel-in-time (P-in-T) methods were developed for time integration problems to circumvent this causality principle and allow for solutions at different time points to be computed concurrently and in parallel. Hence, by re-purposing techniques for the P-in-T solution

of Ordinary Differential Equation (ODE)s and Partial Differential Equation (PDE)s to parallelize over iterations of optimization algorithms, we aim to reduce the wall-clock time required to solve discrete optimization problems. Specifically in this work we focus on the parallel-in-time method known as Multigrid Reduction-in-Time (MGRIT) (multigrid reduction-in-time) [27], which we discussed in great detail in chapter 4. However, we once again note that other suitable parallel-in-time methods, such as the parareal method [62] or the Parallel Full Approximation Scheme in Space-Time (PFASST) method [69, 26] can also be considered.

5.1 Other parallel optimization strategies

Before we discuss our parallel-in-iteration strategy, we briefly discuss two other parallel optimization approaches in section 5.1. We exemplify each of these approaches by one illustrative method, and discuss in which ways our approach is different.

5.1.1 Parallel-in-space optimization

For large scale optimization problems, many parallel optimization methods based on spatial parallelism [15, 84] have been developed for distributing computational work across multiple processors. Among the positive outcomes of parallel-in-space methods are the optimization of memory and storage requirements and the reduction of the wall-clock time required for the convergence of several numerical algorithms. For example, we consider a special case of (5.2), given by

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad g(\mathbf{u}) + \frac{1}{p}\sum_{i=1}^{p} f_i(\mathbf{u}), \tag{5.3}$$

where g is a convex, continuous and possibly nondifferentiable function and $f_1, \ldots, f_p$ are convex, L-smooth functions. Typically, this problem would be solved using PG (5.32), with iteration update given by

$$\mathbf{u}_{k+1} = (I + s\partial g)^{-1}\left(\mathbf{u}_k - \frac{s}{p}\sum_{i=1}^{p}\nabla f_i(\mathbf{u}_k)\right) \tag{5.4}$$

for some $0 < s < 2/L$. In a standard sequential implementation of this PG update, each of the p evaluations of ∇f_i gradients must be performed in sequence, such that the time

required for computing each PG step increases proportionally to p. However, each of these gradient calculations only uses the value of $\mathbf{u}_k$ at the beginning of the iteration, such that in principle each of these gradients may be computed independently of each other. Defining the *consensus set*

$$C \coloneqq \left\{(\mathbf{u}_1, \ldots, \mathbf{u}_p) \mid \mathbf{u}_1 = \cdots = \mathbf{u}_p \in \mathbb{R}^N\right\}, \tag{5.5}$$

the *distributed PG method* [84] instead solves

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_p \in \mathbb{R}^N} \quad g(\mathbf{u}_1) + \frac{1}{p}\sum_{i=1}^{p} f_i(\mathbf{u}_i) \quad \text{s.t. } (\mathbf{u}_1, \ldots, \mathbf{u}_p) \in C \tag{5.6}$$

with iteration update given by

$$\mathbf{u}_{k+1} = (I + s\partial g)^{-1}\left(\mathbf{u}_k - \frac{s}{p}\sum_{i=1}^{p} \nabla f_i((\mathbf{u}_i)_k)\right), \tag{5.7}$$

where $(\mathbf{u}_i)_k$ denotes the value of $\mathbf{u}_i$, $i = 1, \ldots, p$ on the kth iteration. That is, each of the p gradients of ∇f_i is computed by individual agents (i.e., processors) then sent to a central agent who takes their average and performs the PG step. Then, the update $\mathbf{u}_{k+1}$ is broadcast to all individual agents before the following iteration. Hence, the distributed PG method discussed in the example above is able to converge in faster wall-clock time by performing all the p gradient evaluations concurrently in parallel (if there are p processors available, in this case) rather than sequentially, if the time saved by this parallelization outmatches the additional time required to communicate the gradients from each individual agent to the central agent and vice-versa.

While distributed PG is able to reduce the time for computing each PG iteration, it does not reduce the number of PG iterations required for convergence. If this number is very large, then it is the number N_t of updates of the form (5.7) required for convergence that represents the biggest sequential bottleneck, rather than the number p of gradient computations in each update. Our parallel-in-iteration strategy is developed with the intention of computing these N_t iterations in parallel, rather than sequentially, such that we are able to handle this iteration-wise bottleneck that distributed PG and other parallel-in-space methods cannot. However, we note many methods combining parallelization in space and time have been successfully developed [34, 42], such that we consider paralellization-in-space strategies as being complementary to our parallel-in-iteration framework.

5.1.2 Parallel-in-time optimization

Parallel-in-time methods have been previously used in the context of solving PDE-constrained optimization problems either as preconditioners or for solving time-dependent subproblems that arise in the context of time-dependent PDE-constrained optimization, such as in optimal control [95, 97, 38, 64, 59] or for computing the geometry of induction motors [52]. These previous applications differ from our approach, which is to use parallel-in-time methods to parallelize across iterations of some underlying sequential optimization method.

For example, we consider the ParaOpt algorithm [38, 59] for optimal control problems, which belongs to the class of multiple-shooting P-in-T methods [34]. Consider the problem of finding the control $c(t)$ and the state $y(t)$ that minimize the cost function

$$J(c) = \frac{1}{2}\left\|y(T) - y_*\right\|^2 + \frac{\alpha}{2}\int_0^T \|c(t)\|^2 dt, \tag{5.8}$$

where $\alpha > 0$ is a fixed regularization parameter, y_* is a target state. The evolution of the state function $y : [0, T] \to \mathbb{R}^N$ is described by

$$\dot{y}(t) = f(y(t)) + c(t), \tag{5.9}$$

where $c(t)$ is the control term, and the first-order optimality condition is described by

$$\dot{y} = f(y) - \frac{\lambda}{\alpha}, \quad \dot{\lambda} = -\left(f'(y)\right)^T \lambda, \tag{5.10}$$

where λ is the adjoint state; the initial condition for y and the final condition for λ are given by

$$y(0) = y_0, \quad \lambda(T) = y(T) - y_*. \tag{5.11}$$

Note that this system of differential equations is fully coupled; we cannot solve for y without knowing the value of λ and vice-versa.

ParaOpt considers dividing the temporal domain $[0, T]$ into the union of subintervals, $[0, T] = \cup_{\ell=0}^{L-1} [T_\ell, T_{\ell+1}]$. Introducing two sets of intermediate states $(Y_\ell)_{\ell=0,\cdots,L}$ and $(\Lambda_\ell)_{\ell=1,\cdots,L}$ corresponding to approximations of y and λ at times $T_0, \ldots, T_{L-1}$ and $T_1, \ldots, T_L$ respectively, then (5.10) is equivalent to solving

$$\begin{pmatrix} y\left(T_{\ell+1}\right) \\ \lambda\left(T_\ell\right) \end{pmatrix} = \begin{pmatrix} P\left(Y_\ell, \Lambda_{\ell+1}\right) \\ Q\left(Y_\ell, \Lambda_{\ell+1}\right) \end{pmatrix} \tag{5.12}$$

on each of the $[T_\ell, T_{\ell+1}]$ intervals, where P and Q are the nonlinear solution operators for (5.10), with initial condition $y\left(T_l\right) = Y_l$ and final condition $\lambda\left(T_{\ell+1}\right) = \Lambda_{\ell+1}$. Note that P

propagates the state y forward to $T_{\ell+1}$ and Q propagates the adjoint state λ backward to T_ℓ. Defining a nonlinear system of equations in terms of P and Q, (5.12) can be solved by Newton's method (Newton-Raphson (NR)) with the NR iteration update given by

$$\mathcal{F}'\begin{pmatrix} Y^n \\ \Lambda^n \end{pmatrix}\begin{pmatrix} Y^{n+1} - Y^n \\ \Lambda^{n+1} - \Lambda^n \end{pmatrix} = -\mathcal{F}\begin{pmatrix} Y^n \\ \Lambda^n \end{pmatrix}, \qquad n = 0, 1, 2, \ldots, \tag{5.13}$$

where $\mathcal{F}'\left(\begin{smallmatrix} Y^n \\ \Lambda^n \end{smallmatrix}\right)$ denotes the Jacobian matrix of $\mathcal{F}$. The ParaOpt method then solves

$$\mathcal{J}^G\begin{pmatrix} Y^n \\ \Lambda^n \end{pmatrix}\begin{pmatrix} Y^{n+1} - Y^n \\ \Lambda^{n+1} - \Lambda^n \end{pmatrix} = -\mathcal{F}\begin{pmatrix} Y^n \\ \Lambda^n \end{pmatrix}, \quad n = 0, 1, 2, \ldots \tag{5.14}$$

for Y^{n+1} and Λ^{n+1}, where $\mathcal{J}^G\left(\begin{smallmatrix} Y \\ \Lambda \end{smallmatrix}\right) \approx \mathcal{F}'\left(\begin{smallmatrix} Y^n \\ \Lambda^n \end{smallmatrix}\right)$ approximates the true Jacobian; this approximate Jacobian is obtained by replacing the exact (fine-grid) derivatives by computing them on a coarse-grid using parareal [62] (see chapter 4).

The ParaOpt example above illustrates the types of optimization problems that can be naturally solved with parallel-in-time methods, namely optimal control problems with constraints described by time-dependent PDEs on bounded time intervals. Although our approach uses some of the same P-in-T methods that can be used for these problems, such as parareal (which we present in chapter 4 as a special case of MGRIT), the problems we solve are not time-dependent. Instead, we treat the iteration index k of the optimization methods used for solving (5.1) and (5.2) as a discrete time index, such that P-in-T methods allow us to compute distinct iterations of the given optimization method in parallel rather than sequentially.

5.2 Optimization Model Problems

We now introduce the two model problems considered henceforth in this thesis. The first model problem is relatively simple, and serves largely as a stepping stone towards the more challenging second model problem, which presents a more realistic application of our proposed algorithm. The optimization problems we consider have a connection to discretizations of time-dependent heat or diffusion PDEs, and thus help to motivate our use of parallel-in-time methods.

5.2.1 Model Problem 1 (MP1) - Quadratic minimization

For our first optimization model problem, Model Problem 1 (MP1), we consider the straightforward problem of quadratic minimization, formulated as

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad \frac{1}{2}\langle A_1\mathbf{u},\mathbf{u}\rangle - \langle \mathbf{b},\mathbf{u}\rangle, \tag{5.15}$$

where A_1 is the 1D Laplacian matrix (2.5), where $\mathbf{b}\in\mathbb{R}^N$ is some prescribed vector. Since A_1 is symmetric positive definite (SPD), the objective function $f(\mathbf{u}) = \langle A_1\mathbf{u},\mathbf{u}\rangle - \langle \mathbf{b},\mathbf{u}\rangle$ is strongly convex and L-smooth; here, L denotes the smallest Lipschitz constant of ∇f, which, in this case is $L = \|A_1\|$. Since A_1 is SPD, hence invertible, the unique solution of (5.15) is given by $\mathbf{u} = (A_1)^{-1}\mathbf{b}$, which is attained when $\nabla f(\mathbf{u}) = \mathbf{0}$. Note that the objective function f defined by (5.15) satisfies the conditions required for convergence of both (5.30) and (5.31).

5.2.2 Model Problem 2 (MP2) - Elastic obstacle problem (EOP)

The elastic obstacle problem (EOP) [13] consists of finding the equilibrium state of an elastic membrane under a non-penetration constraint. The EOP we consider was initially motivated by mathematical physics [81] such as steady state fluid interaction, thin-plate solid dynamics and elastostatics [93]. Various stategies have been proposed for numerically solving the EOP, including multigrid approaches such as [11, 65, 1, 98]. The formulation we present here follows an L^1 penalty approach [93].

Given an obstacle function $\phi(\mathbf{x}) : \Omega \to \mathbb{R}$, the EOP seeks the location of a membrane $\widehat{u}(\mathbf{x}) : \Omega \to \mathbb{R}$ with the lowest elastic potential energy. Assuming the elastic potential energy of the membrane is proportional to its surface area, the EOP can be written as

$$\min_{\widehat{u}\in\Omega} \quad \int_\Omega \sqrt{1+\|\nabla\widehat{u}\|_2^2}\,\mathrm{d}\mathbf{x} \quad \text{s.t. } \widehat{u}\geq\phi \text{ on } \Omega,\ \widehat{u}=0 \text{ on } \partial\Omega. \tag{5.16}$$

Here, $\nabla\widehat{u} : \Omega \to \mathbb{R}^d$ is the gradient field of $\widehat{u}$, and the norm $\|\cdot\|_2$ induced by the inner product $\langle\cdot,\cdot\rangle$ is the standard 2-norm for functions over Ω. Note that we assume $\phi \leq 0$ on $\partial\Omega$, so the boundary condition $\widehat{u} = 0$ on $\partial\Omega$ is well-posed.

An approximation of (5.16) can be obtained by linearization [98], based on the Maclaurin series $\sqrt{1+x^2} = 1 + \frac{1}{2}x^2 + \mathcal{O}(x^4)$. Assuming that $\|\nabla\widehat{u}\|_2^2$ is sufficiently small such that

we can ignore higher order terms, we have $\sqrt{1+\|\nabla\widehat{u}\|_2^2} \approx 1 + \frac{1}{2}\|\nabla\widehat{u}\|_2^2$; further dropping the constant 1 gives the linearized obstacle problem

$$\min_{\widehat{u}\in\Omega} \quad \int_\Omega \frac{1}{2}\|\nabla\widehat{u}\|_2^2 \, \mathrm{d}\mathbf{x} \quad \text{s.t. } \widehat{u} \geq \phi, \ \widehat{u} = 0 \text{ on } \partial\Omega. \tag{5.17}$$

In order to solve the linearized problem (5.17) we discretize it with finite differences on the aforementioned equispaced meshes.

Consider first the 1D case. Let $x_i, i \in \{1, \ldots, N\}$, denote an interior mesh point, and let $\widehat{u}_i \approx \widehat{u}(x_i)$ be an approximation of $\widehat{u}$ at this mesh point. Then the integral is discretized as

$$\int_\Omega \frac{1}{2}\|\nabla\widehat{u}(x)\|_2^2 \mathrm{d}x \approx \sum_{i=1}^{N} \frac{1}{2}\|\nabla\widehat{u}(x_i)\|_2^2 \, h \approx \frac{h}{2}\langle A_1\widehat{\mathbf{u}}, \widehat{\mathbf{u}}\rangle, \tag{5.18}$$

where $\widehat{\mathbf{u}} = (\widehat{u}_1, \ldots, \widehat{u}_n) \in \mathbb{R}^N$, and $A_1 \in \mathbb{R}^{N\times N}$ is the 1D Laplacian matrix (2.5), which assumes zero boundary conditions. The approximation proceeds analogously in 2D, where now we let (x_i, y_j), $i, j \in \{1, \ldots, N\}$, denote an interior mesh point, and $\widehat{u}_{i,j} \approx \widehat{u}(x_i, y_j)$. In this case, the integral is approximated as

$$\int_\Omega \frac{1}{2}\|\nabla\widehat{u}(x,y)\|_2^2 \, \mathrm{d}x \, \mathrm{d}y \approx \sum_{i,j=1}^{N} \frac{1}{2}\|\nabla\widehat{u}(x_i,y_j)\|_2^2 \, h^2 \approx \frac{h^2}{2}\langle A_2\widehat{\mathbf{u}}, \widehat{\mathbf{u}}\rangle, \tag{5.19}$$

where $\widehat{\mathbf{u}} = (\widehat{u}_{1,1}, \ldots, \widehat{u}_{N,N}) \in \mathbb{R}^{N^2}$ and $A_2 \in \mathbb{R}^{N^2\times N^2}$ is the 2D Laplacian matrix (2.9).

Ignoring the leading h or h^2 factors, since this does not change the solution of the minimization problem, our discretized versions of (5.17) take the form

$$\min_{\widehat{\mathbf{u}}\in\mathbb{R}^{N^d}} \quad \frac{1}{2}\langle A_d\widehat{\mathbf{u}}, \widehat{\mathbf{u}}\rangle \quad \text{s.t. } \widehat{\mathbf{u}} \geq \boldsymbol{\phi}. \tag{5.20}$$

for $d \in \{1, 2\}$. Moreover, $\boldsymbol{\phi} \in \mathbb{R}^N$ denotes the discretized version of $\phi(\mathbf{x})$ on the underlying mesh, and A_d the Laplacian matrix on this mesh. By making a change of variable $\mathbf{u} := \widehat{\mathbf{u}} - \boldsymbol{\phi} \geq \mathbf{0}$, we can write (5.20) in its shifted form:

$$\min_{\mathbf{u}\in\mathbb{R}^{N^d}} \quad \frac{1}{2}\langle A_d\mathbf{u}, \mathbf{u}\rangle + \langle A_d\boldsymbol{\phi}, \mathbf{u}\rangle + \frac{1}{2}\langle A_d\boldsymbol{\phi}, \boldsymbol{\phi}\rangle \quad \text{s.t. } \mathbf{u} \geq \mathbf{0}. \tag{5.21}$$

Dropping the $\frac{1}{2}\langle A_d\boldsymbol{\phi}, \boldsymbol{\phi}\rangle$ term (since it doesn't influence the minimizer), choosing $\mathbf{p} = -A_d\boldsymbol{\phi}$ and using an indicator function $i_+ : \mathbb{R}^N \to \mathbb{R}$, where $i_+(\mathbf{a}) = 0$ if $(\mathbf{a})_q \geq 0$ for all q

and $i_+(\mathbf{a}) = +\infty$ if $(\mathbf{a})_q < 0$ for any q, allows (5.21) to be written in unconstrained form as

$$\min_{\mathbf{u}\in\mathbb{R}^{N^d}} \quad \frac{1}{2}\langle A_d\mathbf{u},\mathbf{u}\rangle - \langle \mathbf{p},\mathbf{u}\rangle + i_+(\mathbf{u}). \tag{5.22}$$

However, we choose to work with an alternative unconstrained formulation, given by

$$\min_{\mathbf{u}\in\mathbb{R}^{N^d}} \quad \frac{1}{2}\langle A_d\mathbf{u},\mathbf{u}\rangle - \langle \mathbf{p},\mathbf{u}\rangle + \lambda \left\|(-\mathbf{u})_+\right\|_1 . \tag{5.23}$$

where $\lambda > 0$ is a predefined penalty parameter and $\sum_i (-(\mathbf{u})_i)_+ = \|(-\mathbf{u})_+\|_1$, where $(a)_+ = \max\{0, a\}$ and $\|\cdot\|_1$ denotes the L^1 norm. For sufficiently large λ, the solution of (5.23) is the same as that of (5.22) [66, 32], hence approximating the solution of (5.17) after reversing the change of variables $\widehat{\mathbf{u}} = \mathbf{u} + \boldsymbol{\phi}$.

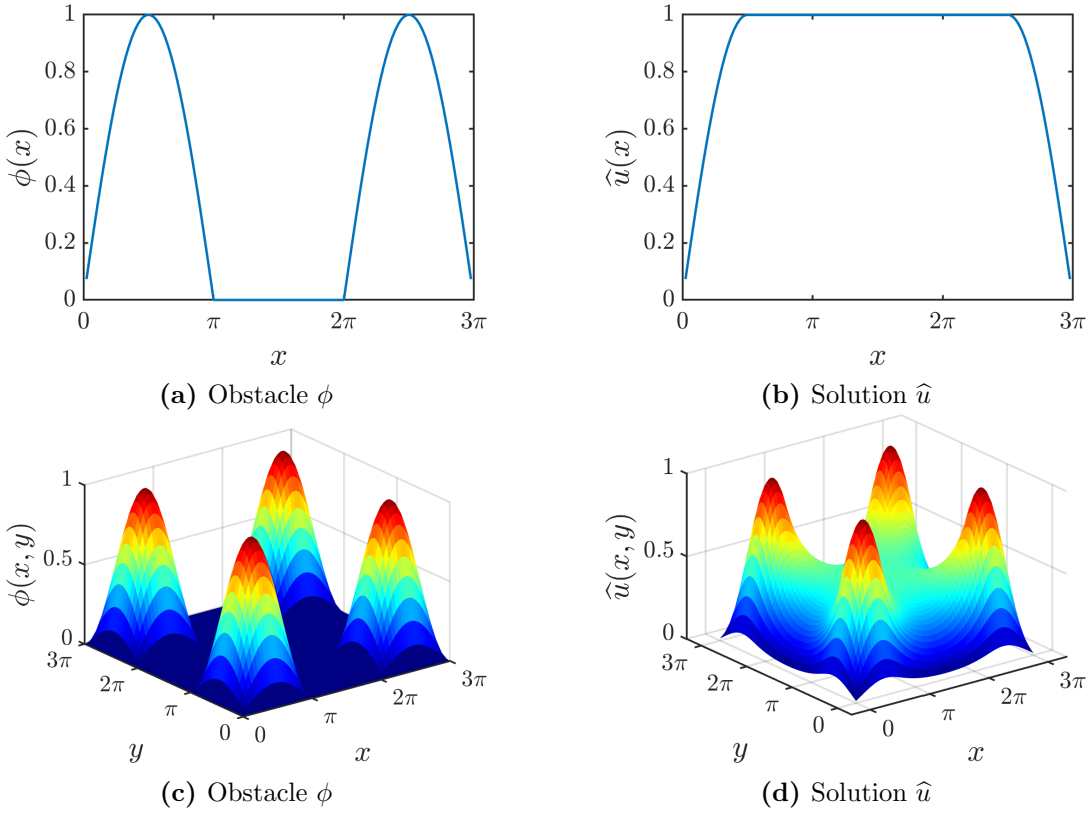
Throughout the rest of this thesis, we refer to the shifted approximate obstacle problem in penalty form (5.23) as Model Problem 2 (MP2)-1D and MP2-2D, in the 1D and 2D cases, respectively. In our MP2-1D numerical experiments, we consider $x \in \Omega := [0, 3\pi]$ and the obstacle $\phi(x) = \max\{0, \sin x\}$, for which (5.17) has the exact solution given by $\widehat{u}(x) = \sin x$ for $x \in \left[0, \frac{1}{2}\pi\right]$ or $x \in \left[\frac{5}{2}\pi, 3\pi\right]$ and $\widehat{u}(x) = 1$ for $x \in \left[\frac{1}{2}\pi, \frac{5}{2}\pi\right]$; see figure 5.1. For MP2-2D, we consider $(x, y) \in \Omega := [0, 3\pi]^2$ and $\phi(x, y) = \max\{0, \sin x\} \max\{0, \sin y\}$; see figure 5.1. We take $\lambda = 900$ in both the 1D and 2D cases, which is large enough such that solving (5.23) is equivalent to solving (5.22). Figure 5.1 presents visualizations of the obstacles and corresponding solutions of MP2-1D and MP2-2D. Note that the solutions pictured in panels (b) and (d) are those on the interior of the domain, and hence with the boundary points $\widehat{u} = 0$ eliminated.

Despite MP1 and MP2 being the only problems we investigate at large, in remark 8 we highlight their relevance as basic model problems for the more general classes of problems (5.1) for the former and (5.2) for the latter.

Remark 8 (On model problems based on minimizing quadratic functions)**.** We note that both MP1, (5.15), and MP2, (5.23), are related to the problem of minimizing quadratic functions of the form

$$f(\mathbf{u}) = \frac{1}{2}\langle A\mathbf{u},\mathbf{u}\rangle - \langle \mathbf{b},\mathbf{u}\rangle, \tag{5.24}$$

for $\mathbf{u} \in \mathbb{R}^N$, where $A \in \mathbb{R}^{N\times N}$ is some SPD matrix and $\mathbf{b} \in \mathbb{R}^N$ is some prescribed vector. In particular, MP1 and MP2 replace the generic positive definite (PD) matrix A in (5.24) by the SPD 1D or 2D Laplacian matrices, (2.5) and (2.9), respectively. In particular, the Laplacian matrices provide a bridge for using P-in-T methods originally developed for Discrete Boundary Value Problem (DBVP)s and Initial Boundary Value Problem (IBVP)s,



(a) Obstacle ϕ **(b)** Solution $\widehat{u}$

(c) Obstacle ϕ **(d)** Solution $\widehat{u}$

Figure 5.1: **(a)** and **(b)**: obstacle ϕ and solution $\widehat{u}$ of MP2-1D (5.23). **(c)** and **(d)**: obstacle ϕ and solution $\widehat{u}$ of MP2-2D (5.23), computed numerically using the PG algorithm (5.32).

which use Laplacian operators to discretize continuous spatial or temporal domains; see chapter 2. However, for the sake of generality, at times we consider MP1 and MP2 with a generic SPD matrix A instead of the A_d Laplacian matrices.

While minimizing a quadratic function (5.24) is a very simple problem, it reflects fundamental behaviours of more general convex smooth problems. To see this, consider a L-smooth, μ-strongly convex function $f : \mathbb{R}^N \to \mathbb{R}$. Then, for $\mathbf{u}, \mathbf{v} \in \mathbb{R}^N$, we have that the L-smoothness of f implies the Descent Lemma (3.23),

$$f(\mathbf{v}) \leq f(\mathbf{u}) + \langle \nabla f(\mathbf{u}), \mathbf{u} - \mathbf{v} \rangle + \frac{L}{2}\|\mathbf{u} - \mathbf{v}\|^2, \tag{5.25}$$

while by fact 11, the μ-strong convexity of f implies that

$$f(\mathbf{v}) \geq f(\mathbf{u}) + \langle \nabla f(\mathbf{u}), \mathbf{u} - \mathbf{v} \rangle + \frac{\mu}{2}\|\mathbf{u} - \mathbf{v}\|^2. \tag{5.26}$$

Thus, every strongly convex and L-smooth function is bounded from above and below by quadratic functions. Furthermore, consider a truncated second-order Taylor expansion of $f(\mathbf{u})$ at $\mathbf{v}$,

$$f(\mathbf{v}) = f(\mathbf{u}) + \langle \nabla f(\mathbf{u}), \mathbf{u} - \mathbf{v} \rangle + \frac{1}{2} \langle H(\mathbf{v})(\mathbf{u} - \mathbf{v}), \mathbf{u} - \mathbf{v} \rangle. \tag{5.27}$$

Then, the upper- and lower-bounds above imply that the Hessian at a point $\mathbf{v}$, $H(\mathbf{v}) \coloneqq \nabla^2 f(\mathbf{v})$, is bounded by

$$\mu I \preceq H(\mathbf{v}) \preceq LI \tag{5.28}$$

for all $\mathbf{v}$. In terms of the eigenvalues $\lambda_1(\mathbf{v}), \ldots, \lambda_n(\mathbf{v})$ of $H(\mathbf{v})$ for some $\mathbf{v}$, we then have that for all $i = 1, \ldots, n$ and any $\mathbf{v}$ we must have

$$\mu \leq \min_i \lambda_i(\mathbf{v}) \leq \lambda_i \leq \max_i \lambda_i(\mathbf{v}) \leq L. \tag{5.29}$$

For a quadratic function of the form (5.24), the Hessian is constant for all $\mathbf{u}$: it is simply the SPD matrix A, with eigenvalues $\lambda_1, \ldots, \lambda_n$, for which we have that $\mu = \min_i \lambda_i$ and $L = \max_i \lambda_i$. Therefore, we have that μ and L perform the same general roles as lower and upper bounds, respectively, for the spectrum of the Hessian, but this time strictly bounding the spectrum in the interval $[\mu, L] \in (0, L]$.

Thus, the analysis we present for solving MP1, (5.15), with a standard method such as GD (5.30) reflects general convergence behaviours of a wide class of more complex convex smooth problems, while remaining analytically tractable. Furthermore, MP2, (5.23), while simply being the minimization of an L_2-regularized quadratic function, is a nonlinear and nonsmooth problem, which is significantly harder to solve than MP1. However, due to their similarities, many of the considerations for the simpler MP1 can be translated to the more difficult MP2.

5.3 Optimization algorithms

We now discuss the optimization algorithms that we apply to the discrete model problems from section 5.2. The main algorithms we consider are gradient descent (GD) for MP1, (5.15), and proximal gradient descent (PG) for MP2, (5.23), whose convergence we attempt to speed up by means of parallelization-in-time In our method, we also make use of two auxiliary implicit algorithms on the coarse grid, namely Proximal Point (PP) for MP1 and Alternating Proximals (AP) for MP2, which are closely related to GD and PG, respectively. We refer the reader to section 3.2 for more on GD and PP, and section 3.3 for more on PG and AP.

5.3.1 Optimization algorithms for MP1

We note that MP1 (5.15) is a case of (5.1), the general convex L-smooth minimization problem, for the strongly convex function $f(\mathbf{u}) = \frac{1}{2}\langle A_1\mathbf{u}, \mathbf{u}\rangle - \langle \mathbf{b}, \mathbf{u}\rangle$, with A_1 the Laplacian (2.5). Moreover, $\partial f(\mathbf{u}) = \{\nabla f(\mathbf{u})\}$, $\nabla f(\mathbf{u}) = A_1\mathbf{u} - \mathbf{b}$ is L-Lipschitz with $L = \|A_1\|_2$, such that f is L-smooth. Therefore, the objective function f defined by (5.15) satisfies the conditions required for convergence of both GD, (5.30), and PP, (5.31). The GD update (5.30) for MP1, (5.15) with stepsize $s = 1/L$ then takes the form

$$\begin{aligned} \mathbf{u}_i = \mathcal{G}_{sf} &:= (I - s\nabla f)(\mathbf{u}_{i-1}) \\ &= (I - sA_1)\mathbf{u}_{i-1} + s\mathbf{b}, \quad i = 1, 2, \ldots. \end{aligned} \tag{5.30}$$

When it is necessary to take steps of size larger than the $2/L$ limit for GD, we take PP steps, which for MP1 take the form

$$\begin{aligned} \mathbf{u}_i := \mathcal{P}_{sf} &= (I + s\nabla f)^{-1}(\mathbf{u}_{i-1}) \\ &= (I + sA_1)^{-1}(\mathbf{u}_{i-1} + s\mathbf{b}), \quad i = 1, 2, \ldots. \end{aligned} \tag{5.31}$$

5.3.2 Optimization algorithms for MP2

We note that MP2 (5.23) is a special case of (3.79), the problem of minimizing the sum of a convex, L-smooth function f and a convex but possibly nondifferentiable function g, with $f = \frac{1}{2}\langle A_d\mathbf{u}, \mathbf{u}\rangle - \langle \mathbf{p}, \mathbf{u}\rangle$ and $g = \lambda \|(-\mathbf{u})_+\|_1$. Here, f is strongly convex and L-smooth, with $L = \|A_d\|_2$, where A_d is given by (2.5) for $d = 1$ and by (2.9) for $d = 2$, and g is convex and continuous. Therefore, f and g satisfy the requirements for convergence of PG, (5.32), which converges to a solution of (3.79) as desired. Hence, we take PG as our basic optimization method for MP2, (5.23), for which the PG update takes the form

$$\begin{aligned} \mathbf{u}_i := \mathcal{P}_{sg} \circ \mathcal{G}_{sf}(\mathbf{u}_{i-1}) &= (I + s\partial g)^{-1}(I - s\nabla f)(\mathbf{u}_{i-1}) \\ &= (I + s\partial g)^{-1}\left((I - sA_d)\mathbf{u}_{i-1} + s\mathbf{b}\right), \quad i = 1, 2, \ldots. \end{aligned} \tag{5.32}$$

It can be shown that the subdifferential of g is given component-wise by

$$(\partial \|(-\mathbf{u})_+\|_1)_i = \begin{cases} \{-1\}, & (\mathbf{u})_i < 0, \\ [-1, 0], & (\mathbf{u})_i = 0, \\ \{0\}, & (\mathbf{u})_i > 0, \end{cases} \tag{5.33}$$

where the i subscript denotes the ith component of the corresponding vector, such that the proximal operator $P_{sg} = (I + s\partial g)^{-1}$ is then given by

$$\left(\mathcal{P}_{s\lambda\|(-\cdot)_+\|_1}(\mathbf{u})\right)_i = \begin{cases} (\mathbf{u})_i + s\lambda, & (\mathbf{u})_i < -s\lambda, \\ 0, & -s\lambda \leq (\mathbf{u})_i \leq 0\,; \\ (\mathbf{u})_i, & (\mathbf{u})_i > 0 \end{cases} \tag{5.34}$$

see [7, Example 25.12] for a similar example.

When it is necessary to take steps of size $s > 2/L$, we substitute PG by AP, (5.35). The AP update for MP2 takes the form

$$\begin{aligned} \mathbf{u}_i \coloneqq \mathcal{P}_{sg} \circ \mathcal{P}_{sf}(\mathbf{u}_{i-1}) &= (I + s\partial g)^{-1} \circ (I + s\nabla f)^{-1}(\mathbf{u}_{i-1}) \\ &= (I + s\partial g)^{-1} \left((I + sA_1 - s\mathbf{b})^{-1}(\mathbf{u}_{i-1})\right), \quad i = 1, 2, \ldots, \end{aligned} \tag{5.35}$$

where P_{sg} is defined in (5.34). We recall from subsection 3.3.2 that (5.35) does not generally converge to a minimizer of (3.79), but to a minimizer of $\mathrm{env}(f) + g$; we address this issue later in subsection 5.5.2.

5.4 Connections between optimization algorithms and differential equations

Here, we describe the connection between the aforementioned optimization algorithms from chapter 3 and discretizations of ODEs, which provides key motivation for our P-in-I framework. The connection between these has been well established in the literature; see [75, 90, 4, 82].

5.4.1 Optimization methods as ODE discretizations

Gradient descent (GD) and Proximal point (PP) First, consider the gradient flow ODE of a convex and L-smooth function f, given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = -\nabla f(\mathbf{u}(t)), \quad \mathbf{u}(0) = \mathbf{u}_0. \tag{5.36}$$

Applying a forward finite-difference approximation to (5.36) yields a forward-Euler discretization (2.134)

$$\mathbf{u}_{k+1} = \mathbf{u}_k - s\nabla f(\mathbf{u}_k), \quad k = 0, 1, \ldots, \tag{5.37}$$

where $\mathbf{u}_{k+1} \approx \mathbf{u}(t_k + \Delta t)$, $\mathbf{u}_k \approx \mathbf{u}(t_k)$, and $s = \Delta t$. Observe that this forward Euler discretization is identical to the GD iteration (5.30). Similarly, applying a backward finite-difference approximation to (5.36) yields the backward Euler discretization (2.135)

$$\mathbf{u}_{k+1} = \mathbf{u}_k - s\nabla f(\mathbf{u}_{k+1}), \quad k = 0, 1, \dots \tag{5.38}$$

Let us recall the PP method (5.31) and note that

$$\begin{aligned} \mathbf{u}_{k+1} = \mathcal{P}_{sf}(\mathbf{u}) &\iff \mathbf{u}_{k+1} = \operatorname*{argmin}_{\mathbf{u}} \left(sf(\mathbf{u}) + \frac{1}{2} \|\mathbf{u} - \mathbf{u}_k\|^2 \right) \\ &\iff \mathbf{u}_{k+1} = \operatorname*{argmin}_{\mathbf{u}} \left(f(\mathbf{u}) + \frac{1}{2s} \|\mathbf{u} - \mathbf{u}_k\|^2 \right) \\ &\iff \mathbf{0} = \nabla f\left(\mathbf{u}_{k+1}\right) + \frac{1}{s}\left(\mathbf{u}_{k+1} - \mathbf{u}_k\right) \\ &\iff \mathbf{u}_{k+1} = \mathbf{u}_k - s\nabla f(\mathbf{u}_{k+1}). \end{aligned} \tag{5.39}$$

Therefore, the PP algorithm (5.31) is equivalent to a backward Euler discretization of (5.36). Furthermore, the PP algorithm can be interpreted as an implicit variant of the GD method (5.30).

As discussed in subsection 2.6.1, in the context of ODEs and PDEs, the forward Euler method as in (5.37) is known to have poor stability properties for stiff equations, putting strict requirements on the step size Δt. On the other hand, the backward Euler discretization as in (5.38) has relatively improved stability properties for stiff equations, typically permitting much larger time step sizes. These stability properties influence the design of our parallel-in-iteration methodology. In particular, the methodology requires us to take large steps on coarse grids, and we chose to use the PP method (5.31) when these steps are larger than $s > 2/L$ so as to avoid instability of GD (5.30).

Proximal gradient (PG) and alternating proximals (AP) Now, consider the gradient flow of $f + g$, assuming, for simplicity, that both f and g are differentiable. The associated gradient flow ODE is

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = -\nabla f(\mathbf{u}(t)) - \nabla g(\mathbf{u}(t)). \tag{5.40}$$

Applying a forward Euler step on the ∇f term, and a backward Euler step on the ∇g term results in the implicit-explicit (IMEX) discretization

$$\mathbf{u}_{k+1} = (I + s\nabla g)^{-1}\left((I - s\nabla f)(\mathbf{u}_k)\right), \quad k = 0, 1, \dots \tag{5.41}$$

Observe that (5.41) is identical to (5.32), such that the PG method can be interpreted as an IMEX discretization of the corresponding gradient flow equation (5.40). Similarly, the AP method (5.35) is equivalent to a backward-backward-Euler discretization of (5.40). Hence, for a similar reasoning as that outlined at the end of subsection 5.4.1, when we need to take steps larger than $s > 2/L$, to avoid the instability of PG, which it inherits from GD, we use the AP method (5.35) instead, which is unconditionally stable due to being a composition of two (uncoditionally stable) PP steps.

5.4.2 Links between solving ODEs using MGRIT and parallel-in-iteration optimization

Having established in section 5.4 that many optimization algorithms, in particular GD (5.30), may be interpreted as discretizations of ODEs, we further strengthen the connection between solving ODEs and optimization problems by showing the equivalence between solving our first model problem, MP1 (5.15), with the GD method (5.30), and solving the 1D heat equation IBVP with the standard spatial and temporal discretization choices discussed in chapter 2. In particular, this equivalence means that we are able to use MGRIT for solving MP1, since MGRIT is known to work well for parabolic PDE problems [27, 24, 55],

Recall section 4.1 and consider using MGRIT to solve the 1D heat equation IBVP on a bounded spatial domain $\Omega = [0, X] \subset \mathbb{R}$ with Dirichlet boundary conditions and constant source term:

$$\begin{aligned} \frac{\partial}{\partial t}u(x,t) - \frac{\partial^2}{\partial x^2}u(x,t) = b, \qquad & (x,t) \in \Omega \times [0,T], \\ u(x,0) = u_0(x), \qquad & x \in \Omega. \end{aligned} \tag{5.42}$$

To discretize this problem, let $x_j = j\Delta x$, $t_i = i\Delta t$, $j = 0, 1, \ldots, N$ and $i = 0, 1, \ldots, N_t$ be a uniform space-time mesh with spacing $\Delta x = X/N$ and $\Delta t = T/N_t$, respectively. We discretize the spatial domain by approximating the second derivative with respect to x by a second-order central finite difference. This yields

$$\frac{\partial^2}{\partial x^2}u\left(x_j, t_i\right) \approx \frac{u^i_{j-1} - 2u^i_j + u^i_{j+1}}{\Delta x^2} \tag{5.43}$$

Leting $\mathbf{u}_i = [u^i_1, u^i_2, \ldots, u^i_N]^T$ be the vector of solutions at all interior spatial points at time step i, and assuming $u^n_0 = 0$ and $u^n_{N+1} = 0$ to eliminate the Dirichlet boundary conditions, we represent the spatial discretization across the entire spatial grid by

$$-\frac{\partial^2}{\partial x^2}\mathbf{u}\left(x, t_n\right) \approx A_1 \mathbf{u}_n, \tag{5.44}$$

where A_1 is the 1D Laplacian matrix (2.5). For the temporal discretization, we approximate the derivative with respect to t with a forward Euler approximation,

$$\frac{\partial}{\partial t}u\left(x_j, t_i\right) \approx \frac{u_j^{i+1} - u_j^i}{\Delta t}, \tag{5.45}$$

which we can describe over the entire spatial grid as

$$\frac{\partial}{\partial t}\mathbf{u}\left(x, t_i\right) \approx \frac{\mathbf{u}_{i+1} - \mathbf{u}_i}{\Delta t} \tag{5.46}$$

Substituting the spatial and temporal discretizations above onto the 1D heat equation,

$$\frac{\partial}{\partial t}u(x,t) - \frac{\partial^2}{\partial x^2}u(x,t) = b, \tag{5.47}$$

letting $\mathbf{b} = [b, b, \ldots, b]^T$ represent the constant source term applied across the spatial grid, yields

$$\frac{\mathbf{u}_{i+1} - \mathbf{u}_i}{\Delta t} + A_1\mathbf{u}_i = \mathbf{b}. \tag{5.48}$$

Letting $s = \Delta t$, we rearrange the discretized 1D heat equation as the explicit update

$$\mathbf{u}_{i+1} = (I - sA_1)\mathbf{u}_i + s\mathbf{b}. \tag{5.49}$$

Therefore, denoting $\Phi := (I - sA_1)$ and $\mathbf{w}_i = s\mathbf{b}$ for $i = 1, \ldots, N_t$, the discretized 1D heat equation is represent by the linear all-at-once system (4.7) given by

$$\mathcal{A}\mathbf{u} = \begin{bmatrix} I & & & \\ -\Phi & I & & \\ & \ddots & \ddots & \\ & & -\Phi & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N_t} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{N_t} \end{bmatrix} = \mathbf{w}. \tag{5.50}$$

where $\mathbf{u}_0 = \mathbf{w}_0$ is the initial condition. Now, we consider solving MP1, (5.15), using the GD method, (5.30). Recall from (5.30) that the ith GD iteration for MP1, (5.15), with stepsize $0 < s < 2/L$, is given by

$$\mathbf{u}_i = (I - sA_1)\mathbf{u}_{i-1} + s\mathbf{b}. \tag{5.51}$$

Then, letting $\Phi := (I - sA_1)$ and $\mathbf{w}_i = s\mathbf{b}$ for $i = 1, \ldots, N_t$, we can represent the act of computing N_t iterations of GD all-at-once for MP1, starting from some initial guess $\mathbf{u}_0 \in \mathbb{R}^N$, by the same all-at-once system (5.50) as the discretized 1D heat equation.

Therefore, solving MP1, (5.15), all-at-once using the GD method, is equivalent to solving the discretized 1D heat equation IBVP, (5.42), with forward Euler discretization on the temporal domain and second-order finite difference (Laplacian) discretization on the spatial domain.

While previous research on solving the 1D heat equation IBVP (5.42) with MGRIT, namely [27, 24], certainly motivates and informs our investigation, differences in objective and approach justify a new exploration of this model problem, regardless of the similarities. We discuss this in the following remark.

Remark 9 (MGRIT for the heat equation vs. MP1)**.** Both [27, 24] are strongly concerned with matching the accuracy of spatial and temporal discretizations of (5.42), with [27] testing different ratios between time step lengths and spatial mesh sizes and [24] employing higher-order time discretizations in order to match the spatial and temporal discretization orders. Secondly, with this accuracy concern in mind, [27, 24] mostly employ implicit discretizations, which are unconditionally stable in regards to time step lengths, on all levels. These objectives differ from ours significantly, as we are concerned with providing speedup over sequential solves for problems to which explicit optimization methods are typically applied, and in which the time step lengths follow a standard choice (such as $s = 1/L$ for GD) for the chosen method. Furthermore, the aforementioned accuracy concerns in both space and time are not present here, as our accuracy requirement is that the MGRIT solution at the final time points approximates the solution of the model problem to a similar tolerance accuracy as the sequential solve which we are trying to speed up. Therefore, it is necessary for us to provide our own numerical experiments and speedup estimates for this model problem, despite the similarities with [27, 24].

5.5 MGRIT for parallel-in-iteration optimization

Having suggested in subsection 5.4.2 that solving certain optimization problems using MGRIT is equivalent to solving a corresponding discretized ODE problem, we borrow from the MGRIT P-in-T framework for time integration differential problems, presented in section 4.1, to establish our P-in-I framework for optimization problems. We then generally consider solving convex minimization problems of the form (5.1) or (5.2) with a 1-step iterative optimization algorithm of the form

$$\mathbf{u}_{i+1} = \Phi(\mathbf{u}_i) + \mathbf{w}_{i+1}, \quad i = 0, 1, \ldots, \tag{5.52}$$

where $\Phi(\cdot)$ is a linear or nonlinear operator and $\mathbf{w}_{i+1}$ is a generic term that does not depend on $\mathbf{u}_k$. Suppose this iterative method converges to an exact minimizer of the optimization

problem, given by

$$\mathbf{u}^* := \lim_{i \to +\infty} \mathbf{u}_i. \tag{5.53}$$

Assume N_t iterations of the given method (5.52) are required in order to approximate $\mathbf{u}^*$ to the desired accuracy; i.e., $\mathbf{u}_{N_t} \approx \mathbf{u}^*$ is satisfactorily accurate approximate solution of the underlying optimization problem. Given an initial guess $\mathbf{u}_0 \in \mathbb{R}^N$, computing N_t iterations of (5.52) is equivalent to solving a system of equations of the form

$$\begin{aligned} \mathbf{u}_0 &= \mathbf{w}_0, \\ \mathbf{u}_i &= \Phi_i(\mathbf{u}_{i-1}) + \mathbf{w}_i, \quad i = 1, 2, \ldots, N_t. \end{aligned} \tag{5.54}$$

Hence, computing the required N_t iterations of (5.52) in parallel is equivalent to solving the below all-at-once system in parallel:

$$\mathcal{A}(\mathbf{u}) = \begin{bmatrix} I & & & \\ -\Phi(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N_t} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{N_t} \end{bmatrix} = \mathbf{w}, \tag{5.55}$$

where $\mathbf{w}_0 = \mathbf{u}_0$ is the initial condition; note that (5.55) is identical to (4.3). By the discussion in section 4.1, the system above can be solved in parallel using the Full Approximation Scheme (FAS)-MGRIT algorithm, algorithm 7, with the iteration index $i = 0, 1, \ldots, N_t$ playing the role of a discrete "time" index. Furthermore, if Φ is a linear operator, then (5.55) is equivalent to the linear all-at-once system (4.7), such that we can solve it using the linear MGRIT algorithm, algorithm 6. Hence, all discussion and considerations made in chapter 4 apply directly here; the key challenge remains finding an appropriate coarse-grid operator, Ψ, such that $\Psi \approx \Phi^m$ is both a cheap and accurate approximation. In particular, Ψ must generally be an operator corresponding to an implicit optimization method, as we discuss in remark 10.

Remark 10 (On the need for implicit coarse-grid operators)**.** We recall that explicit time-discretization schemes such as forward Euler (2.134) have stability issues, which prevents the use of large time steps. As a result, a large number of time steps are usually required for convergence; in turn, this presents a larger potential for parallelization in time when compared to schemes which converge in few iterations. In order to solve a smaller sequential problem, MGRIT coarsens the original problem such that larger time steps must be taken on the coarse grid. To circumvent stability issues with explicit methods which limit the step size lengths, MGRIT can employ implicit time-discretization schemes such as backward Euler (2.135) on coarse grids, as in [27]. Unfortunately, the implicit computations that

must be done to solve the coarse-grid problem are costlier than explicit ones; with a careful choice of the coarse-grid operator and coarsening factor, however, the parallelization on the fine grid can offset this additional cost such that MGRIT can provide speedup over the sequential time-stepping approach.

Analogously, explicit optimization methods such as GD (5.30) and PG (5.32) also require time steps to be bounded in order for them to converge; in particular, both require $s < 2/L$, where L is the Lipschitz constant of the gradient of ∇f. With the connection between optimization methods and time-discretization schemes from section 5.4 in mind, we infer that, in order to solve an optimization in parallel using MGRIT with an explicit method, we must restrict its use to the fine grid and instead employ an implicit method on the coarse grid. In particular, for a coarsening factor m, we need one step of the implicit method to accurately and cheaply approximate m steps of the explicit method.

Given a pair of candidates for fine- and coarse-grid operators, MGRIT convergence analysis theory can provide some information as to what makes a good approximation under some reasonable assumptions; e.g., if both fine- and coarse-grid operators are simultaneously diagonalizable by the same unitary transformation, error reduction factors can be estimated as a function of the corresponding eigenvalues, as established in fact 43 from [24]; a similar two-level reduction analysis can be done if the Fourier symbols of the operators satisfy simultaneous diagonalization (not necessarily unitary) assumption [23]. On the other hand, research on suitable choices of coarse-grid operators is often limited to certain classes of problems, such as recent works on modified semi-Lagrangian coarse-grid operators for advection-dominated and hyperbolic PDEs [23, 21, 19, 22, 58].

For solving MP1, (5.15), and MP2, (5.23), with MGRIT, we recall that the former is a special case of (5.1) and the latter of (5.2). Hence, following the discussion in section 3.2 and section 3.3, respectively, we take the fine-grid method to be GD (5.30) for MP1 and PG (5.32) for MP2. We discuss our choices of the coarse-grid operator Ψ for each of these model problems in the following.

5.5.1 Choice of Ψ for MP1 using gradient descent

Inspired by the similarities between our model problem (5.15) and the 1D heat equation (5.42), in this paper we follow a simple heuristic inspired by [27] of simply replacing forward Euler steps on the finest level with backward Euler steps on the coarse s. Recall from (5.51) that we take

$$\Phi := I - sA_1. \tag{5.56}$$

We further refine this by choosing the step size to be $s = 1/L$, which is the classical choice for GD; see subsection 3.2.1. As discussed in section 5.4, the natural implicit candidate for the coarse-grid operator is the PP operator in (5.31):

$$\Psi := (I + msA_1)^{-1}. \tag{5.57}$$

That is, we use the approximation

$$\Psi := (I + msA_1)^{-1} \approx (I - sA_1)^m = \Phi^m. \tag{5.58}$$

Intuitively, following the discussion in section 5.4, we are approximating m forward Euler time steps (2.134), each with step size s, with a single backward Euler time step (2.135) of size ms. We recall that the PP method (5.31) converges for any $s > 0$, such that $ms = \frac{m}{L}$ is a feasible step size choice for any coarsening factor m.

In a multilevel MGRIT setting of $\widehat{\ell} \geq 2$ levels, we take $\ell = 1$ to refer to the fine grid. For $2 \leq \ell \leq \widehat{\ell}$, the ℓth level employs the operator

$$\Psi^\ell := (I + m^{\ell-1}sA_1)^{-1}, \tag{5.59}$$

an implicit operator similar to (5.57) with the appropriate time step coarsening; when $\ell = 2$, we omit the superscript. We note that the computation of the inverse matrices in (5.57) and (5.59) amounts to solving a linear system, which we do by means of LU-factorization; the L and U factors are stored, such that the factorization is computed only once per level.

Analysis of the $\Psi \approx \Phi^m$ approximation for the MP1 operators

To rigorously analyze the approximation (5.58) in an MGRIT setting using fact 43, we first prove that Φ (5.56) and Ψ (5.57) satisfy (4.99), such that Φ and Ψ are simultaneously diagonalizable by the same unitary transformation Q (4.98), whose columns form an orthonormal basis of eigenvectors $\mathbf{q}_i$, $i = 1, \ldots, N$. For the sake of generality, we consider a generic SPD matrix $A \in \mathbb{R}^{N\times N}$ in place of the 1D Laplacian matrix A_1, (2.5).

Proposition 1. *Let $A \in \mathbb{R}^{N\times N}$ be an SPD matrix, such that A can be diagonalized as $A = Q\Lambda Q^*$, where Q is an unitary matrix (i.e., $Q^*Q = QQ^* = I$) and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ is the diagonal matrix of eigenvalues of A. Let $\Phi = I - sA$ and $\Psi = (I + msA)^{-1}$. Then, Φ and Ψ are simultaneously diagonalizable by Q:*

$$\Phi = Q(I - s\Lambda)Q^*, \quad \Psi = Q(I + ms\Lambda)^{-1}Q^*, \tag{5.60}$$

where $I - s\Lambda$ and $(I + ms\Lambda)^{-1}$ are diagonal matrices.

Proof. Substituting $A = Q\Lambda Q^*$ in the definition of Φ yields

$$\Phi = I - sA = QIQ^* - sQ\Lambda Q^* = Q(I - s\Lambda)Q^*. \tag{5.61}$$

Note that I and Λ are diagonal matrices; since s is a scalar, $s\Lambda$ is also a diagonal matrix. Thus, $I - s\Lambda$ is a diagonal matrix, such that Φ is unitarily diagonalized by Q. Similarly, we substitute $A = Q\Lambda Q^*$ in the definition of Ψ:

$$\begin{aligned} \Psi = (I + msA)^{-1} &= (QIQ^* + msQ\Lambda Q^*)^{-1} \\ &= (Q(I + ms\Lambda)Q^*)^{-1} \end{aligned} \tag{5.62}$$

Using the property of inverse matrices that $(XYZ)^{-1} = Z^{-1}Y^{-1}X^{-1}$ and the fact that Q is unitary, such that $Q^{-1} = Q^*$ and $(Q^*)^{-1} = Q$, we get:

$$\Psi = (Q^*)^{-1}\,(I + ms\Lambda)^{-1}Q^{-1} = Q(I + ms\Lambda)^{-1}Q^* \tag{5.63}$$

Note that $I + ms\Lambda$ is a diagonal matrix, such that its inverse, $(I + ms\Lambda)^{-1}$, is also a diagonal matrix. Hence, Ψ is also unitarily diagonalized by Q. □

We can show that Φ and Ψ commute as a direct consequence of their simultaneous diagonalizability, . However, diagonalizability is not required: we prove the stronger result that Φ and Ψ commute for any square matrix A such that $I + msA$ is invertible.

Corollary 2. *Let $\Phi = I - sA$ and $\Psi = (I + msA)^{-1}$, where $A \in \mathbb{R}^{N\times N}$ is a square matrix such that $I + msA$ is invertible. Then Φ and Ψ commute.*

Proof. Let $M = I + msA$, such that $\Psi = M^{-1}$. First, we compute ΦM,

$$\Phi M = (I - sA)(I + msA) = I + msA - sA - ms^2A^2 \tag{5.64}$$

and $M\Phi$,

$$M\Phi = (I + msA)(I - sA) = I - sA + msA - ms^2A^2. \tag{5.65}$$

Since matrix addition is commutative, both expressions are identical:

$$\Phi M = I + (m - 1)sA - ms^2A^2 = M\Phi. \tag{5.66}$$

Now, multiplying $\Phi M = M\Phi$ from the left and the right by M^{-1} yields

$$M^{-1}(\Phi M)M^{-1} = M^{-1}(M\Phi)M^{-1}. \tag{5.67}$$

By the associativity of matrix multiplication, we regroup the terms as

$$(M^{-1}\Phi)(MM^{-1}) = (M^{-1}M)(\Phi M^{-1}). \tag{5.68}$$

Substituting $MM^{-1} = M^{-1}M = I$ yields

$$M^{-1}\Phi = \Phi M^{-1}. \tag{5.69}$$

Substitute Ψ back in for M^{-1}, we have that

$$\Psi\Phi = \Phi\Psi, \tag{5.70}$$

such that Φ and Ψ commute. □

In particular, assuming commutativity in the absence of diagonalizability may be useful for MGRIT analysis if we are to consider applying the GD and PP operators considered here, for example, for minimizing a quadratic function of the form (5.24) where A is PD, but not SPD. Returning to the SPD setting, with Φ and Ψ simultaneously diagonalizable by proposition 1, we can analyze the approximation $\Phi^m \approx \Psi$ through their eigenvalues,

$$\begin{aligned} \phi_n^m &= (1 - s\lambda_n)^m, \\ \psi_n &= \frac{1}{1 + ms\lambda_n}, \end{aligned} \tag{5.71}$$

where ϕ_n and ψ_n, $N = 1, \ldots, N$ are the eigenvalues of Φ and Ψ as in (4.99), and λ_n denotes the nth eigenvalue of A for $n = 1, \ldots, N$. We now show that Ψ is stable (i.e., $|\psi_n| < 1$ for all n) unconditionally, while Φ is stable (i.e., $|\phi_n| < 1$ for all n) for all $0 < s < 2/L$, where $L = \max_n \lambda_n$. In particular, this provides another proof for the Courant-Friedrichs-Lewy (CFL) limit of GD.

Lemma 3. *Let $\Phi = I - sA$ and $\Psi = (I + msA)^{-1}$, where $A \in \mathbb{R}^{N\times N}$ is an SPD matrix with eigenvalues $\lambda_n > 0$ for $n = 1, \ldots, N$, where $L \coloneqq \max_n \lambda_n$. Let ϕ_n and ψ_n denote the eigenvalues of Φ and Ψ, respectively, for $n = 1, \ldots, N$. Then, $|\psi_n| < 1$ for all n for any $s > 0$, and $|\phi_n| < 1$ for all n if $0 < s < 2/L$. Hence, we say that Ψ is unconditionally stable, while Φ is stable for $0 < s < 2/L$.*

Proof. For the eigenvalues of Ψ, note that

$$0 < \psi_n \coloneqq \frac{1}{1 + ms\lambda_n} < 1 \tag{5.72}$$

unconditionally, since for all $m \geq 1$, $s > 0$ and $\lambda_n > 0$ we have that the denominator $1 + ms\lambda_n > 1$. For the eigenvalues of Φ, note that $|\phi_n| < 1$ for some $n = 1, \ldots, N$ is equivalent to

$$-1 < 1 - s\lambda_n < 1. \tag{5.73}$$

Breaking this inequality into two inequalities, we have that

$$1 - s\lambda_n < 1 \implies s\lambda_n > 0, \tag{5.74}$$

which is trivially satisfied since $s > 0$ and $\lambda_n > 0$ for all n. For the other inequality,

$$-1 < 1 - s\lambda_n \implies s\lambda_n < 2 \implies s < \frac{\lambda_n}{2}. \tag{5.75}$$

Thus, for $|\phi_n| < 1$ hold for all n, we must have

$$s < \max_n \frac{\lambda_n}{2} = \frac{L}{2} \implies s < \frac{2}{L}. \tag{5.76}$$

□

Now, we take $s = 1/L$, as considered in our numerical experiments. To simplify our notation, we denote $z_n = s\lambda_n = \frac{\lambda_n}{L}$ for $n = 1, \ldots, N$; we say that z_n is nth normalized eigenvalue of A, with $x_n \in (0, 1]$ for all n. Then, we denote the nth eigenvalues of Φ and Ψ, defined in (5.71), in terms of x_n by

$$\begin{aligned} \phi_n^m &= (1 - z_n)^m, \\ \psi_n &= \frac{1}{1 + mz_n}. \end{aligned} \tag{5.77}$$

Note that for $s = 1/L$, the assumptions of fact 43 are all satisfied. Therefore, the factor by which the error at C-points is reduced by one iteration of MGRIT with Φ as the fine-grid operator and Ψ as the coarse-grid operator, with coarsening factor m, is bounded from above by

$$B_m = \max_n \frac{|\phi_n^m - \psi_n|\,|\phi_n^m|}{1 - |\psi_n|} \tag{5.78}$$

First, we evaluate the $|\phi_n^m - \psi_n|$, which is the error of the approximation $\Phi^m \approx \Psi$ for the nth mode, $n = 1, \ldots, N$. For this analysis, we assume that the normalized spectrum is approximately continuous in $(0, 1]$ for a sufficiently large N. In the following remark, we justify that this assumption is valid in particular when A is a Laplacian matrix.

Remark 11 (Laplacian matrices have asymptotically dense (normalized) spectrum)**.** Consider the MP1 setting of $s = 1/L$ and $A = A_1$ (2.5), the 1D Laplacian, with the eigenvalues λ_n of A_1 defined in (2.6). In particular, we have that

$$0 < \mu \coloneqq \min_n \lambda_n = \lambda_1 \leq \lambda_n \leq L \coloneqq \max_n \lambda_n = \lambda_N < \frac{4}{h^2}, \tag{5.79}$$

where $h = 1/(N+1)$. Note that as $N \to +\infty$, the normalized spectrum of A_1, $z_n = \lambda_n/L \in (0, 1]$, densely covers the continuous $(0, 1]$ interval, by which we mean that

$$\lim_{N \to +\infty} \lambda_n - \lambda_{n-1} = 0 \quad \forall n = 2, \ldots, N. \tag{5.80}$$

We say that a matrix such as A_1 whose N eigenvalues satisfy (5.80) is a matrix with *asymptotically dense spectrum* on the interval $(0, L]$. Consequently, (5.80) implies that the normalized eigenvalues of the form $z_n = \lambda_n/L$ form a *asymptotically dense normalized spectrum* on the interval $(0, 1]$. In particular, the 2D Laplacian matrix A_2, (2.9), with eigenvalues given in (2.11), also satisfies (5.80).

Proposition 2. *Assume that A is a matrix with dense normalized spectrum (see remark 11), and assume $z \in (0, 1]$ to be continuous. Let $\phi^m(z) = (1-z)^m$ and $\psi = \frac{1}{1+mz}$. Then,*

$$\phi^m(z) - \psi(z) = -\frac{m(m+1)}{2} z^2 + \mathcal{O}(z^3). \tag{5.81}$$

Proof. For z near 0, the binomial expansion of $(1-z)^m$ yields

$$\phi^m(z) = 1 - mz + \frac{m(m-1)}{2} z^2 - \frac{m(m-1)(m-2)}{6} z^3 + \mathcal{O}(z^4). \tag{5.82}$$

Similarly, for z near 0, specifically $mz = m\lambda_n/L < 1$, the geometric series expansion of $(1+mz)^{-1}$ yields:

$$\begin{aligned} \psi(z) &= 1 - mz + (mz)^2 - (mz)^3 + \mathcal{O}(z^4) \\ &= 1 - mz + m^2 z^2 - m^3 z^3 + \mathcal{O}(z^4). \end{aligned} \tag{5.83}$$

Taking the difference and the absolute value yields the desired result. □

Therefore, the approximation $\Phi \approx \Phi^m$ is more accurate at modes corresponding to smaller eigenvalues z near zero, with the error increasing for z closer to 1. In particular, for $z = sL = 1$, we have that the error of approximation is exactly

$$\phi_n^m(1) - \psi_n(1) = (1-1)^n - \frac{1}{1+m} = -\frac{1}{1+m}, \tag{5.84}$$

such that

$$|\phi_n^m(1) - \psi_n(1)| = \frac{1}{1+m} \leq \frac{1}{3}. \tag{5.85}$$

However, note in (5.78) that each $|\phi_n^m - \psi_n|$ term is multiplied by $|\phi_n|^m$. For $z = 1$, we have that

$$|\phi_n^m(1) - \psi_n(1)||\phi_n^m(1)| = \frac{1}{1+m} \cdot 0 = 0; \tag{5.86}$$

for $z \approx 1$, we have

$$\phi_n^m(z) = (1-z)^m \approx 0. \tag{5.87}$$

This result showcases the typical multigrid dynamic where the relaxation, corresponding to $|\phi_n|^m$, is more effective on the upper half of the spectrum (i.e., $z \to 1$), the coarse-grid correction, associated with $|\psi_n - \phi_n^m|$, is more effective on the lower half (i.e., $z \to 0$).

We now prove that for $s = 1/L$, the upper bound (5.78) is stricly smaller than 1, such that MGRIT is convergent for this choice of fine- and coarse-grid operators.

Theorem 2. *Let $s = 1/L$, $\Phi = I - sA$ and $\Psi = (I + msA)^{-1}$. Suppose the simultaneous diagonalization assumption in (4.99) is satisfied, where ϕ_n and ψ_n are the nth eigenvalues of Φ and Ψ. Then, for all $n = 1, \ldots, N$ we have that*

$$B_{m,n} = \frac{|\phi_n^m - \psi_n||\phi_n|^m}{1 - |\psi_n|} < 1. \tag{5.88}$$

Therefore, $\max_n B_{m,n} = B_m < 1$ for all m, where B_m is defined in (5.78), such that MGRIT converges by fact 43.

Proof. Substituting the definitions of ϕ_n and ψ_n from (5.77) into (5.88), we get

$$B_{m,n} = \frac{\left|(1-z_n)^m - \frac{1}{1+mz_n}\right|}{1 - \frac{1}{1+mz_n}} |1 - z_n|^m \tag{5.89}$$

Simplifying the denominator,

$$1 - \frac{1}{1+mz_n} = \frac{1+mz_n-1}{1+mz_n} = \frac{mz_n}{1+mz_n}, \tag{5.90}$$

and the numerator,

$$(1-z_n)^m - \frac{1}{1+mz_n} = \frac{(1-z_n)^m(1+mz_n) - 1}{1+mz_n}, \tag{5.91}$$

simplifies the expression for $B_{m,n}$ to

$$B_{m,n} = \frac{|(1-z_n)^m(1+mz_n)-1|}{mz_n}(1-z_n)^m \tag{5.92}$$

To remove the absolute value, we analyze the function $f(z_n) = (1-z_n)^m(1+mz_n)$. Taking the derivative with respect to z_n yields

$$\begin{aligned} f'(z_n) &= -m(1-z_n)^{m-1}(1+mz_n) + m(1-z_n)^m \\ &= m(1-z_n)^{m-1}\big[-(1+mz_n)+(1-z_n)\big] \\ &= -m(m+1)z_n(1-z_n)^{m-1}; \end{aligned} \tag{5.93}$$

note that for any $z_n \in (0,1]$, $f'(z_n) \leq 0$. Since $f(0) = 1$ and the function is strictly decreasing in $(0,1]$, we have that $f(z_n) \leq 1$ for all z_n, such that we may substitute the absolute value by negative of the terms inside it:

$$B_{m,n} = \frac{1-(1-z_n)^m(1+mz_n)}{mz_n}(1-z_n)^m \tag{5.94}$$

Now, let $g = (1-z_n)^m$; note that $0 \leq g < 1$ for $z_n > 0$. Substituting g into the latest equation for $B_{m,n}$ yields

$$B_{m,n} = \frac{1-g(1+mz_n)}{mz_n}g = \frac{g-g^2(1+mz_n)}{mz_n} \tag{5.95}$$

We want to prove that $B_{m,n} < 1$. Because $mz_n > 0$, we can multiply both sides of the inequality $B_{m,n} < 1$ by mz_n and rearrange the result:

$$\begin{aligned} & g - g^2(1+mz_n) < mz_n \\ \iff & g - g^2 - g^2mz_n < mz_n \\ \iff & g - g^2 < mz_n + g^2mz_n, \end{aligned} \tag{5.96}$$

which yields the inequality

$$g - g^2 \leq mz_n(1+g^2). \tag{5.97}$$

To prove this inequality is universally true, we apply Bernoulli's inequality, which states that for $z_n \in (0,1]$ and integer $m \geq 1$, we must have

$$(1-z_n)^m \geq 1 - mz_n. \tag{5.98}$$

Substituting $g = (1-z_n)^m$, we get

$$g \geq 1 - mz_n \implies 1 - g \leq mz_n. \tag{5.99}$$

Multiplying both sides of this inequality by g yields

$$g(1-g) \leq mz_n g \iff g - g^2 \leq mz_n g. \tag{5.100}$$

Because $g^2 - g + 1 > 0$ for all real numbers, it follows that $g < 1 + g^2$. Hence, substituting this upper bound for g on the right side provides

$$g - g^2 \leq mz_n g < mz_n(1 + g^2), \tag{5.101}$$

which confirms the required inequality and proves that $B_{m,n} < 1$ for all $z_n \in (0, 1]$. □

By theorem 2, MGRIT with this choice of fine-grid operator Φ and coarse-grid operator Ψ converges for any choice of m, in the sense that the MGRIT approximation error eventually approaches 0 for any arbitrary accuracy threshold. However, theorem 2 does not say anything about how *quickly* MGRIT will converge. It is not practical to evaluate

$$B_{m,n} = \frac{|\phi_n^m - \psi_n||\phi_n|^m}{1 - |\psi_n|} = \frac{1 - (1 - z_n)^m(1 + mz_n)}{mz_n}(1 - z_n)^m \tag{5.102}$$

for every possible m and z_n in order to compute $\max_m B_m = \max_m(\max_n B_{m,n})$, which would yield an upper bound for (5.78) over all possible choices of m. However, by once again treating $z \in (0, 1]$ as a continuous variable, in the following we maximize the continuous equivalent of $B_{m,n}$ for extreme choices of $m = 2$ and $m \to +\infty$, assuming that results for finite $m \geq 2$ fall somewhere in between.

Lemma 4 (MGRIT error bounds for the MP1 Φ and Ψ operators)**.** *Let $z \in (0, 1)$ and*

$$B_m(z) \coloneqq \frac{1 - (1 - z)^m(1 + mz)}{mz}(1 - z)^m. \tag{5.103}$$

Let $B_m \coloneqq \max_{z\in(0,1]} B_m(z)$. Then $B_2 \approx 0.177$ and $B_\infty \coloneqq \lim_{m\to+\infty} B_m \approx 0.112$.

Proof. Substitute $m = 2$ into the formula for $B_m(z)$ yields

$$B_2(z) = \frac{1 - (1 - z)^2(1 + 2z)}{2z}(1 - z)^2 \tag{5.104}$$

Expanding $(1 - z)^2(1 + 2z)$ yields

$$(1 - 2z + z^2)(1 + 2z) = 1 + 2z - 2z - 4z^2 + z^2 + 2z^3 = 1 - 3z^2 + 2z^3, \tag{5.105}$$

which we substitute back into the fraction:

$$\frac{1-(1-3z^2+2z^3)}{2z} = \frac{3z^2-2z^3}{2z} = \frac{3z-2z^2}{2}. \tag{5.106}$$

Multiplying by $(1-z)^2$ then yields

$$\begin{aligned} B_2(z) &= \frac{1}{2}(3z-2z^2)(1-2z+z^2) \\ &= \frac{1}{2}(-2z^4+7z^3-8z^2+3z) \end{aligned} \tag{5.107}$$

To find the maximum, we set the derivative of B_2 with respect to z to zero, which yields

$$\frac{1}{2}(-8z^3+21z^2-16z+3) = 0. \tag{5.108}$$

Thus, we must find the roots of the cubic polynomial $-8z^3+21z^2-16z+3=0$. Notice that if we test $z=1$, the polynomial evaluates to $-8+21-16+3=0$. Therefore, $(z-1)$ is a factor, such that we factor the cubic polynomial as

$$(z-1)(-8z^2+13z-3) = 0. \tag{5.109}$$

To find the other 2 roots, we use the quadratic formula to find the roots of $-8z^2+13z-3=0$, which are

$$z = \frac{13 \pm \sqrt{169-4(8)(3)}}{16} = \frac{13 \pm \sqrt{73}}{16}. \tag{5.110}$$

Noticing that $\frac{13+\sqrt{73}}{16} \approx 1.346 \notin (0,1]$, the other valid root is $z^* = \frac{13-\sqrt{73}}{16} \approx 0.2785$. Plugging z^* into the formula for B_2 yields

$$B_2(z^*) \approx 0.177. \tag{5.111}$$

Evaluating the other root $z=1$ yields $B_2(1)=0$, such that z^* is indeed the maximizer. Therefore, $B_2 = \max_z B_2(z) = B_2(z^*) \approx 0.177$.

Now, define a new scaled variable $y = mz$, such that $z = y/m$. Substitute $z = y/m$ into the definition of $B(z)$ yields:

$$B(y) = \frac{1-\left(1-\frac{y}{m}\right)^m(1+y)}{y}\left(1-\frac{y}{m}\right)^m \tag{5.112}$$

Using the fact that $\lim_{m\to+\infty}(1-y/m)^m = e^{-y}$ by the definition of the exponential function, we take the limit as $m \to +\infty$ and obtain

$$B_\infty(y) \coloneqq \lim_{m\to+\infty} B_m(y) = \frac{1-e^{-y}(1+y)}{y}e^{-y}. \tag{5.113}$$

Setting the derivative of $B_\infty(y)$ with respect to y to zero, we get

$$2y^2 + 3y + 1 = e^y. \tag{5.114}$$

Solving this numerically yields the critical point $y \approx 1.7933$, which we plug back into $B_\infty(y)$ to find

$$B_\infty(1.7933) = \frac{1-e^{-1.7933}(1+1.7933)}{1.7933}e^{-1.7933} \approx 0.112. \tag{5.115}$$

□

By lemma 4, we expect the error at the C-points after each MGRIT iteration to be reduced by a worst-case factor somewhere in between 11% and 18% for sufficiently large N, which suggests MGRIT will converge in a small number of iterations. While taking the largest possible m might seem like the best choice by lemma 4, note that the larger m is, the more the problem is solved sequentially via relaxations rather than the coarse-grid correction. Since each MGRIT iteration does more work than the full sequential solve, this might translate into a increase in wall-clock time, rather than the reduction we are aiming for. In section 5.8, we investigate the optimal choice of m from a parallel speedup perspective.

5.5.2 Choice of Ψ for MP2 using proximal gradient descent

The choice of coarse-grid operator for MP2, (5.23), is analogous to that for MP1 in that we choose Φ to be the PG operator (5.32) and Ψ to be its implicit analogue, the AP operator (5.35), with coarsened step size. Namely, we choose

$$\Phi \coloneqq \mathcal{P}_{sg}(I - sA_d) \quad \text{and} \quad \Psi^\ell = \mathcal{P}_{m^{\ell-1}sg}(I + m^{\ell-1}sA_d)^{-1}, \quad 2 \le \ell \le \widehat{\ell}, \tag{5.116}$$

where $g \coloneqq \lambda\|(-\cdot)_+\|$, as in (5.23), and the proximal operator P_{sg} is given by (5.34). The coarse-grid operators in (5.116) are chosen under the assumption that, on the first coarse grid (i.e., $\ell = 2$),

$$\Psi \coloneqq \mathcal{P}_{msg}(I + msA_d)^{-1} \approx (\mathcal{P}_{sg}(I - sA_d))^m = \Phi^m \tag{5.117}$$

is a sufficiently accurate approximation, which is consistent with the assumption in (5.58). Before we move on to an in-depth analysis of the approximation proposed in (5.117), we comment more generally on our heuristic of replacing gradient steps by proximal steps in order to derive implicit variants of explicit optimization algorithms.

Remark 12 (The gradient-proximal heuristic)**.** We note that the heuristic presented in subsection 5.5.1 and subsection 5.5.2, by which we replace gradient descent terms by proximals, can be applied to any gradient-based method. For nonquadratic objective functions, we may need to invert a nonlinear system to compute these proximals by means of (3.37). Alternatively, we can simply compute the proximals numerically by minimizing the function in the definition (3.34) using some iterative method; this computational approach is specially useful when the proximal for the relevant function does not have a closed form.

We also note that replacing the explicit gradient steps by implicit proximal steps is only required when taking steps larger than the stability limit of the explicit method. In some machine learning contexts, for example, parameter training stages are performed using arbitrarily small steps well within the stability limit, such that (proximal) gradient descent could be used on coarse levels without drawbacks. Finally, we note that our heuristic still applies in principle to more complex methods with varying step sizes and/or additional parameters. However, determining effective Ψ can become a quite complicated task depending on the complexity of the Φ algorithm.

Analysis of the $\Psi \approx \Phi^m$ approximation for the MP2 operators

In subsection 3.3.2, we noted in the general setting of (5.2) that while the PG sequence generated by (5.32) converges to an exact minimizer of $F := f + g$, the sequence generated by AP actually converges to a minimizer of $\mathrm{env}(f) + g$. However, the error with respect this exact minimizer of (5.2) decreases proportionally to the stepsize s.

Proposition 3 (Error estimate of minimizing $F = f + g$ using AP)**.** *Let $F := f + g$ as in (3.79), and let $\widetilde{F} := \mathrm{env}_s\, f + g$. Let $\mathbf{u}^* = \arg\min_{\mathbf{u}} F(\mathbf{u})$ and $\widetilde{\mathbf{u}} = \arg\min_{\mathbf{u}} \widetilde{F}(\mathbf{u})$, assuming for simplicity they are the unique minimizers of their respective composite functions. Then, the difference between $\mathbf{u}^*$ and $\widetilde{\mathbf{u}}$, i.e., the error of the solution obtained by minimizing (3.79) with AP, satisfies*

$$\widetilde{\mathbf{u}} - \mathbf{u}^* \approx s\nabla f(\mathbf{u}^*) \tag{5.118}$$

for a sufficiently small s, such that $\lim_{s\to 0} \widetilde{\mathbf{u}} = \mathbf{u}^*$.

Proof. Since $\mathbf{u}^*$ is the minimizer of F, by Fermat's rule (fact 14) we must have

$$0 \in \nabla f(\mathbf{u}^*) + \partial g(\mathbf{u}^*) \implies -\nabla f(\mathbf{u}^*) \in \partial g(\mathbf{u}^*). \tag{5.119}$$

Similarly, since $\widetilde{\mathbf{u}}$ is the minimizer of $\widetilde{F}$, we have that

$$0 \in \nabla(\mathrm{env}_s\, f(\widetilde{\mathbf{u}})) + \partial g(\widetilde{\mathbf{u}}). \tag{5.120}$$

By fact 29, we know that

$$s\nabla(\mathrm{env}_s\, f(\widetilde{\mathbf{u}})) = \widetilde{\mathbf{u}} - \mathcal{P}_{sf}(\widetilde{\mathbf{u}}). \tag{5.121}$$

Since f is differentiable, by fact 25 we have that

$$s\nabla f(\mathcal{P}_{sf}(\widetilde{\mathbf{u}})) = \widetilde{\mathbf{u}} - \mathcal{P}_{sf}(\widetilde{\mathbf{u}}). \tag{5.122}$$

Therefore, (5.121) and (5.122) together yield

$$\nabla(\mathrm{env}_s\, f(\widetilde{\mathbf{u}})) = \nabla f(\mathcal{P}_{sf}(\widetilde{\mathbf{u}})). \tag{5.123}$$

Substituting this into (5.120), we get

$$0 \in \nabla f(\mathcal{P}_{sf}(\widetilde{\mathbf{u}})) + \partial g(\widetilde{\mathbf{u}}) \implies -\nabla f(\mathcal{P}_{sf}(\widetilde{\mathbf{u}})) \in \partial g(\widetilde{\mathbf{u}}). \tag{5.124}$$

From (5.122), we can write $\widetilde{\mathbf{u}} = \mathcal{P}_{sf}(\widetilde{\mathbf{u}}) + s\nabla f(\mathcal{P}_{sf}(\widetilde{\mathbf{u}}))$. Substituting this relation into the subdifferential yields

$$-\nabla f(\mathcal{P}_{sf}(\widetilde{\mathbf{u}})) \in \partial g\big(\mathcal{P}_{sf}(\widetilde{\mathbf{u}}) + s\nabla f(\mathcal{P}_{sf}(\widetilde{\mathbf{u}}))\big). \tag{5.125}$$

Taking the limit as $s \to 0$ gives

$$-\nabla f\left(\lim_{s\to 0} \mathcal{P}_{sf}(\widetilde{\mathbf{u}})\right) \in \partial g\left(\lim_{s\to 0} \mathcal{P}_{sf}(\widetilde{\mathbf{u}})\right), \tag{5.126}$$

which recovers the optimality condition of a minimizer of $F = f + g$ shown in (5.119). Therefore, we can conclude that

$$\lim_{s\to 0} \mathcal{P}_{sf}(\widetilde{\mathbf{u}}) = \mathbf{u}^*. \tag{5.127}$$

Assuming that s is sufficiently small, we can approximate

$$\mathcal{P}_{sf}(\widetilde{\mathbf{u}}) \approx \mathbf{u}^*. \tag{5.128}$$

Then, by (5.122) it follows that

$$\widetilde{\mathbf{u}} - \mathbf{u}^* \approx s\nabla f(\mathbf{u}^*). \tag{5.129}$$

□

From the proof above we note that the error is zero, i.e., $\widetilde{\mathbf{u}} = \mathbf{u}^*$, if $\nabla f(\mathbf{u}^*) = 0$, i.e., if $\mathbf{u}^* = \arg\min f$. Therefore, the error is zero if $g = 0$, in which case AP, (5.35), reduces to PP, (5.31). Thus, we recover the fact that the PP and GD sequences both converge to a minimizer of a convex and L-smooth function f.

Based on proposition 3, we expect that approximating one PG step with one AP step becomes more accurate as the stepsize s becomes smaller, i.e., closer to zero. However, in the MGRIT setting we need one AP step to approximate m PG steps, as in (5.117). While proposition 3 may still suggest that (5.117) is a good approximation if we assume that $\mathcal{P}_{msg} \circ \mathcal{P}_{msf} \approx \mathcal{P}_{sg} \circ \mathcal{P}_{sf}$ for sufficiently small m and s, it is of our interest to provide a more robust analysis specifically for MP2, (5.23). Thus, we look at the approximation (5.117) for the more easily tractable coarsening factor $m = 2$ in the following, noting that the general insights obtained by this analysis apply to any $m \geq 2$.

Recall that the MGRIT convergence theory in chapter 4 in terms of the eigenvalues of operators is only well-defined for linear MGRIT operators. However, we can investigate the approximation (5.117) in more detail by looking at how the proximals with respect to the nondifferentiable $g \coloneqq \lambda\|(-\cdot)_+\|$ are applied to each component. We recall from (5.34) that the proximal operator acting on a vector $\mathbf{v}$ with threshold parameter α is defined component-wise as

$$(\mathcal{P}_\alpha(\mathbf{v}))_i = \begin{cases} (\mathbf{v})_i + \alpha, & (\mathbf{v})_i < -\alpha \\ 0, & -\alpha \leq (\mathbf{v})_i \leq 0 \\ (\mathbf{v})_i, & (\mathbf{v})_i > 0, \end{cases} \tag{5.130}$$

where $\alpha = s\lambda$ for the fine grid and $\alpha = 2s\lambda$ for the coarse grid.

To evaluate the approximation $(\Phi^2\mathbf{u})_i \approx (\Psi\mathbf{u})_i$, we assume $s = 1/L$ is small enough such that we can approximate the coarse-grid Ψ step by the second-order Taylor expansion $(I + 2sA)^{-1} \approx I - 2sA + 4s^2A^2$. We now evaluate $(\Phi^2\mathbf{u})_i \approx (\Psi\mathbf{u})_i$ for $(\mathbf{u})_i$ in the intervals outlined by the branches of (5.130). In the following, let $A = A_d$ be either the $d = 1$ (2.5) or $d = 2$ (2.9) Laplacian matrix.

1. Suppose that $(\mathbf{u})_i > 0$; we fall into the third case of (5.130), where the solution component is above the obstacle and is left unchanged, with the proximal acting as the identity operator. In this case, applying Φ twice yields

$$(\Phi^2\mathbf{u})_i = ((I - sA)^2\mathbf{u})_i = (\mathbf{u})_i - 2s(A\mathbf{u})_i + s^2(A^2\mathbf{u})_i, \tag{5.131}$$

   while applying Ψ once yields

$$(\Psi\mathbf{u})_i \approx (\mathbf{u})_i - 2s(A\mathbf{u})_i + 4s^2(A^2\mathbf{u})_i. \tag{5.132}$$

Hence, the coarse-grid approximation error is given by

$$(\Phi^2\mathbf{u})_i - (\Psi\mathbf{u})_i \approx -3s^2(A^2\mathbf{u})_i, \tag{5.133}$$

which recovers the linear approximation error of (5.58).

2. Suppose that $(\mathbf{u})_i < -2s\lambda$ and that its immediate neighbors are similarly deeply under the obstacle; we fall into the first case in (5.130) for both the coarse and fine grids. The solution components in this local neighborhood must be pushed upwards by the proximal steps. Assuming the spatial diffusion is sufficiently small such that the intermediate fine-grid step for node i and its neighbors remains in this branch (i.e., $(\mathbf{v}^{(1)})_j < -s\lambda$ for all j in the stencil), the first fine-grid step yields locally

$$(\mathbf{v}^{(1)})_j = ((I - sA)\mathbf{u})_j + s\lambda, \tag{5.134}$$

such that the second fine-grid step evaluated at component i yields

$$(\Phi^2\mathbf{u})_i = ((I - sA)\mathbf{v}^{(1)})_i + s\lambda. \tag{5.135}$$

Substituting the local definition of $\mathbf{v}^{(1)}$ and expanding, we get

$$\begin{aligned}(\Phi^2\mathbf{u})_i &= ((I - sA)^2\mathbf{u})_i + s\lambda - s(A(s\lambda\mathbf{1}))_i + s\lambda \\ &= (\mathbf{u})_i - 2s(A\mathbf{u})_i + s^2(A^2\mathbf{u})_i + 2s\lambda - s^2\lambda\sum_j A_{ij} \\ &= (\mathbf{u})_i - 2s(A\mathbf{u})_i + s^2(A^2\mathbf{u})_i + 2s\lambda,\end{aligned} \tag{5.136}$$

where $\sum_j A_{ij} = 0$ for $A = A_d$ by the definition of the Laplacian matrices, (2.5) for $d = 1$ and (2.9) for $d = 2$. For one coarse-grid step, we get

$$(\Psi\mathbf{u})_i \approx (\mathbf{u})_i - 2s(A\mathbf{u})_i + 4s^2(A^2\mathbf{u})_i + 2s\lambda, \tag{5.137}$$

such that the difference is

$$(\Phi^2\mathbf{u})_i - (\Psi\mathbf{u})_i \approx -3s^2(A^2\mathbf{u})_i, \tag{5.138}$$

which is also the same linear approximation error of (5.58).

3. Suppose that $-2s\lambda \leq (\mathbf{u})_i < -s\lambda$; this is an intermediate case where the component is moderately inside the obstacle. For the fine grid, since $(\mathbf{u})_i < -s\lambda$, the first step is an upwards push, yielding

$$(\mathbf{v}^{(1)})_i = ((I - sA)\mathbf{u})_i + s\lambda. \tag{5.139}$$

Assuming we now have $-s\lambda \leq (\mathbf{v}^{(1)})_i < 0$, the second fine-grid step falls into the middle branch of (5.130) and maps the component to zero, meaning $(\Phi^2\mathbf{u})_i \approx 0$. On the other hand, for the coarse grid, since $(\mathbf{u})_i \geq -2s\lambda$, it immediately falls into the middle branch of the coarse proximal operator (with $\alpha = 2s\lambda$), meaning $(\Psi\mathbf{u})_i \approx 0$. Thus, in both grids map the component is mapped to zero, preserving a small approximation error.

4. Finally, suppose that $-s\lambda \leq (\mathbf{u})_i \leq 0$, meaning the solution is slightly past the obstacle. For the fine-grid step, the proximal operator of the first Φ step maps this i-th component directly to zero:

$$(\mathbf{v}^{(1)})_i = 0. \tag{5.140}$$

The second fine-grid step then starts with the GD step spreading information from neighboring components:

$$(\mathbf{v}^{(2)})_i = ((I - sA)\mathbf{v}^{(1)})_i = 0 - s\sum_{j\neq i} A_{ij}(\mathbf{v}^{(1)})_j, \tag{5.141}$$

such that the second Φ step concludes with the proximal evaluation

$$(\Phi^2\mathbf{u})_i = \mathcal{P}_{s\lambda}\left(-s\sum_{j\neq i} A_{ij}(\mathbf{v}^{(1)})_j\right). \tag{5.142}$$

For one coarse-grid Ψ step, we begin by computing the PP step

$$(\mathbf{z})_i \approx (\mathbf{u})_i - 2s(A\mathbf{u})_i + 4s^2(A^2\mathbf{u})_i. \tag{5.143}$$

Depending on the magnitude of this step, we might fall into a different branch of the proximal operator. For example, if $(\mathbf{z})_i$ overshoots into the obstacle, i.e., $(\mathbf{z})_i < -2s\lambda$, the proximal performs an upward shift, yielding

$$(\Psi\mathbf{u})_i = (\mathbf{z})_i + 2s\lambda, \tag{5.144}$$

such that the Φ^2 and Ψ trajectories diverge.

Therefore, we expect that the approximation error (5.117) will be small and behave exactly as the linear MP1 approximation of m GD steps by one PP, (5.58), when the i-th component of the solution vector $(\mathbf{u})_i$ is either above the obstacle, deeply inside the obstacle, or moderately inside it. However, for $(\mathbf{u})_i$ slightly past the obstacle, i.e., $-s\lambda \leq (\mathbf{u})_i \leq 0$, the approximation may behave erratically, such that relaxation steps are required to correct this coarse-grid approximation error.

Remark 13 (Possible proximal branch-crossing in other cases)**.** In the case-by-case analysis above, we assumed that the spatial diffusion term $s(A\mathbf{u})_i$ was sufficiently small such that intermediate steps remained within their starting proximal branches. In practice, however, the diffusion term $(A\mathbf{u})_i$ can push the fine- and coarse-grid trajectories into different branches across all cases if sufficiently large in magnitude. For instance, in Case 1 where $(\mathbf{u})_i > 0$, a large positive $(A\mathbf{u})_i$) can cause the fine-grid intermediate step $(\mathbf{v}^{(1)})_i = (\mathbf{u})_i - s(A\mathbf{u})_i$ to drop below zero, triggering the active set. Meanwhile, the coarse grid takes a larger spatial step $2s(A\mathbf{u})_i$ and may overshoot deeply into the obstacle, evaluating a different branch of the proximal operator.

5.6 MGRIT convergence in the (generalized) gradient

Given the exact fine-grid trajectory $\{\mathbf{u}_i\}$ and the MGRIT approximation $\{\mathbf{v}_i\}$ of $\{\mathbf{u}_i\}$, MGRIT convergence criteria in our numerical experiments are set in terms of the norm of the MGRIT coarse-grid residual $\mathbf{r}(\mathbf{v}) = \mathbf{w} - \mathcal{A}(\mathbf{v})$ over the entire space-iteration domain, where $\mathbf{v} \in \mathbb{R}^{N \cdot N_t}$ (see (5.50)), while the sequential convergence is given in terms of the gradient of the objective function $F(\mathbf{u}_i)$ at iteration i, where $\mathbf{u}_i \in \mathbb{R}^N$. The following lemma relates these two measures of convergence by establishing a bound for the (generalized) gradient (5.145) of the MGRIT approximation $\mathbf{v}_i$ at iteration i in terms of associated error $\mathbf{e}_i = \mathbf{u}_i - \mathbf{v}_i$ and the (generalized) gradient of the exact $\mathbf{u}_i$.

The setting for the lemma and subsequent remarks is the general problem of miniziming $F = f + g$ as in (3.79), where f is a convex L-smooth function, and g is a continuous, convex and possibly nondifferentiable function. More specifically, we consider the proximal-gradient iteration for minimizing $F = f + g$, with its convergence measured in terms of the generalized gradient of F:

$$G_{sF}(\mathbf{u}_i) := \frac{1}{s}\left[\mathbf{u}_i - \mathcal{P}_{sg}(\mathbf{u}_i - s\nabla f(\mathbf{u}_i))\right] = \frac{1}{s}(I - \mathcal{P}_{sg} \circ \mathcal{G}_{sf}(\mathbf{u}_i)). \tag{5.145}$$

Note that $\mathcal{P}_{sg} \circ \mathcal{G}_{sf}$ is the same as the PG operator (5.32); in particular, $\mathcal{P}_{sg} \circ \mathcal{G}_{sf} = I - sG_{sF}$. Furthermore, when $g = 0$, $F = f$ and $G_{sF} = \nabla f$. Therefore, the problem of minimizing f (3.48) using the GD method (5.30) can be seen as a special case of the problem of minimizing $F = f + g$ with the PG method (5.32). We note that for both MP1 and MP2, we take $s = 1/L$. For MP1 and MP2-1D, we have that $L = \|A_1\|_2$, where A_1 is the 1D Laplacian matrix (2.5), while for MP2-2D, we have that $L = \|A_2\|_2$, where A_2 is the 2D Laplacian matrix (2.9). Moreover, we consider a general parallel-in-iteration method for an all-at-once system as in (4.3), such that the following results are not particular to the MGRIT method.

Lemma 5. *Let $f : \mathbb{R}^N \to \mathbb{R}$ be a convex and L-smooth function, $g : \mathbb{R}^N \to \mathbb{R}$ a continuous convex function. Consider a parallel-in-iteration method described by an all-at-once system of the form (4.3) with $\Phi := I - sG_{sF}$ the PG operator of $f + g$ from (5.32), with step size $s = 1/L$. Let $\mathbf{v}_i \approx \mathbf{u}_i$ be an approximation of the ith component of the solution of (4.3), with associated error $\mathbf{e}_i = \mathbf{u}_i - \mathbf{v}_i$. Then, the generalized gradient (5.145) of the approximate solution satisfies*

$$\|G_{sF}(\mathbf{v}_i)\| \leq L\sqrt{2}\,\|\mathbf{e}_i\| + \|G_{sF}(\mathbf{u}_i)\|. \tag{5.146}$$

Proof. Since $\mathbf{e}_i = \mathbf{u}_i - \mathbf{v}_i$, we can write $\mathbf{v}_i = \mathbf{u}_i - \mathbf{e}_i$. By fact 22, $\Phi := I - sG_{sF} = \mathcal{P}_{sg} \circ \mathcal{P}_{sf}$ is 2/3-averaged. Therefore, by plugging $\alpha = 2/3$ and $T = I - s\mathcal{G}_{sf}$ into (3.31), we get

$$\begin{aligned}\frac{1}{2}\|sG_{sF}(\mathbf{v}_i) - sG_{sF}(\mathbf{u}_i)\|^2 &\leq \|\mathbf{v}_i - \mathbf{u}_i\|^2 - \|(I - sG_{sF})(\mathbf{v}_i) - (I - sG_{sF})(\mathbf{u}_i)\|^2 \\ &\leq \|\mathbf{v}_i - \mathbf{u}_i\|^2 = \|\mathbf{e}_i\|^2.\end{aligned} \tag{5.147}$$

Taking the square root on both sides, we get

$$\|sG_{sF}(\mathbf{v}_i) - sG_{sF}(\mathbf{u}_i)\| \leq \sqrt{2}\,\|\mathbf{e}_i\|. \tag{5.148}$$

Using the triangle inequality, we get

$$\begin{aligned}\|sG_{sF}(\mathbf{v}_i)\| &= \|sG_{sF}(\mathbf{v}_i) - sG_{sF}(\mathbf{u}_i) + sG_{sF}(\mathbf{u}_i)\| \\ &\leq \|sG_{sF}(\mathbf{v}_i) - sG_{sF}(\mathbf{u}_i)\| + \|sG_{sF}(\mathbf{u}_i)\| \\ &\leq \sqrt{2}\,\|\mathbf{e}_i\| + s\,\|G_{sF}(\mathbf{u}_i)\|.\end{aligned} \tag{5.149}$$

Therefore,

$$\begin{aligned}\|G_{sF}(\mathbf{v}_i)\| &\leq \frac{\sqrt{2}}{s}\,\|\mathbf{e}_i\| + \|G_{sF}(\mathbf{u}_i)\| \\ &= L\sqrt{2}\,\|\mathbf{e}_i\| + \|G_{sF}(\mathbf{u}_i)\|.\end{aligned} \tag{5.150}$$

□

Bound (5.146) in lemma 5 shows that, when $\mathbf{u}_i$ approximates a minimizer of $F(\mathbf{u})$ with accuracy $\|G_{sF}(\mathbf{u}_i)\|$, then our parallel-in-iteration method will approach the same accuracy $\|G_{sF}(\mathbf{v}_i)\|$ for $\mathbf{v}_i$ if we run sufficient iterations of the parallel-in-iteration to make $\|\mathbf{e}_i\|$ small relative to $\|G_{sF}(\mathbf{u_i})\|/L\sqrt{2}$. While lemma 5 holds generally for both MP1 and MP2 settings, we are able to derive a slightly tighter bound for the MP1 setting with $\Phi = I - s\nabla f$ the GD operator (5.30).

Lemma 6. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex and L-smooth function. Consider using MGRIT to minimize f with the GD fine-grid operator $\Phi := I - s\nabla f$, where $s = 1/L$ is the time-step length. Let $k = 1, \dots, K$ denote the MGRIT iteration and $i = 1, \dots, N_t$ be a point in the fine grid. Let $\mathbf{u}_i$ denote the exact solution, $\mathbf{u}_i^{(k)}$ the MGRIT approximate solution and $\mathbf{e}_i^{(k)} = \mathbf{u}_i - \mathbf{u}_i^{(k)}$ the error of the approximate MGRIT solution, at time i and iteration k. Then,*

$$\left\|\nabla f\left(\mathbf{u}_i^{(k)}\right)\right\| \leq L\left\|\mathbf{e}_i^{(k)}\right\| + \left\|\nabla f\left(\mathbf{u}_i\right)\right\|. \tag{5.151}$$

Proof. Since f is L-smooth, ∇f is a Lipschitz function with Lipschitz constant L, such that by (3.22) we have

$$\left\|\nabla f\left(\mathbf{u}_i - \mathbf{e}_i^{(k)}\right) - \nabla f\left(\mathbf{u}_i\right)\right\| \leq L\left\|\left(\mathbf{u}_i - \mathbf{e}_i^{(k)}\right) - \left(\mathbf{u}_i\right)\right\| = L\left\|\mathbf{e}_i^{(k)}\right\|. \tag{5.152}$$

Since $\mathbf{e}_i^{(k)} = \mathbf{u}_i - \mathbf{u}_i^{(k)}$, we can write $\mathbf{u}_i^{(k)} = \mathbf{u}_i - \mathbf{e}_i^{(k)}$. Therefore,

$$\begin{aligned}
\left\|\nabla f\left(\mathbf{u}_i^{(k)}\right)\right\| &= \left\|\nabla f\left(\mathbf{u}_i - \mathbf{e}_i^{(k)}\right)\right\| \\
&\leq \left\|\nabla f\left(\mathbf{u}_i - \mathbf{e}_i^{(k)}\right) - \nabla f\left(\mathbf{u}_i\right)\right\| + \left\|\nabla f\left(\mathbf{u}_i\right)\right\| \\
&\leq L\left\|\left(\mathbf{u}_i - \mathbf{e}_i^{(k)}\right) - \mathbf{u}_i\right\| + \left\|\nabla f\left(\mathbf{u}_i\right)\right\| \\
&= L\left\|\mathbf{e}_i^{(k)}\right\| + \left\|\nabla f\left(\mathbf{u}_i\right)\right\|.
\end{aligned} \tag{5.153}$$

□

Note that lemmas 5 and 6 can also be restated in terms of residuals rather than errors; we present the result for the linear case in the following remark.

Remark 14. Recall the all-at-once system (4.3) and the related residual equation (4.4). If Φ is a linear operator such that lemma 5 holds, then $\mathcal{A}\mathbf{e} = \mathbf{r}$, such that $\mathbf{e} = \mathcal{A}^{-1}\mathbf{r}$. Therefore, $\|\mathbf{e}_i\| \leq \|\mathbf{e}\| = \|\mathcal{A}^{-1}\mathbf{r}\| \leq \|\mathcal{A}^{-1}\|\|\mathbf{r}\|$, which together with (5.146) implies

$$\left\|G_{sF}\left(\mathbf{v}_i\right)\right\| \leq L\sqrt{2}\|\mathcal{A}^{-1}\|\|\mathbf{r}\| + \left\|G_{sF}\left(\mathbf{u}_i\right)\right\|. \tag{5.154}$$

Similarly, if lemma 6 holds, by (5.151) we get

$$\left\|\nabla f\left(\mathbf{v}_i\right)\right\| \leq L\|\mathcal{A}^{-1}\|\|\mathbf{r}\| + \left\|\nabla f\left(\mathbf{u}_i\right)\right\|. \tag{5.155}$$

As such, considering large $i \approx N_t$, we find that a sufficiently small space-time residual implies accurate approximation of the minimizer.

In particular, let $\mathbf{v}_i^{(k)}$ be the approximate solution at time i and iteration k, and let $\mathbf{e}_i^{(k)} = \mathbf{u}_i - \mathbf{v}_i^{(k)}$ be the error of the approximation. Similarly, let $\mathbf{v}^{(k)}$ be the approximate solution vector at iteration k. Next, we consider the assumption of linear convergence of the all-at-once system. For MGRIT, linear convergence has been proven for certain linear problems as in [20, 89, 24], and widely observed numerically even for nonlinear problems [19, 28], including in the results we present in subsection 5.7.2.

Remark 15. Suppose the parallel-in-iteration method described in lemma 5 converges linearly in error for the problem of minimizing $F = f + g$ with convergence factor $\rho_{\mathbf{e}}$, that is,

$$\left\|\mathbf{e}^{(k)}\right\| \leq \rho_{\mathbf{e}} \left\|\mathbf{e}^{(k-1)}\right\| \leq (\rho_{\mathbf{e}})^{(k)} \left\|\mathbf{e}^{(0)}\right\|. \tag{5.156}$$

for $\rho_{\mathbf{e}} \in (0,1)$ some constant independent of k. Then, from (5.146) we get linear convergence of the gradient at any point i in time:

$$\left\|G_{sF}\left(\mathbf{v}_i^{(k)}\right)\right\| \leq \left(L\sqrt{2}\left\|\mathbf{e}^{(0)}\right\|\right)(\rho_{\mathbf{e}})^{(k)} + \left\|G_{sF}\left(\mathbf{u}_i\right)\right\|. \tag{5.157}$$

In particular, suppose that after $k = k_*$ iterations $\left(L\sqrt{2}\left\|\mathbf{e}^{0}\right\|\right)(\rho_{\mathbf{e}})^{k_*} := \epsilon_*$ is some sufficiently small number. Then,

$$\left\|G_{sF}\left(\mathbf{v}_i^{(k)}\right)\right\| \leq \epsilon_* + \left\|G_{sF}\left(\mathbf{u}_i\right)\right\|. \tag{5.158}$$

If we have linear convergence in the residual, that is,

$$\left\|\mathbf{r}^{(k)}\right\| \leq \rho_{\mathbf{r}} \left\|\mathbf{r}^{(k-1)}\right\| \leq (\rho_{\mathbf{r}})^{(k)} \left\|\mathbf{r}^{(0)}\right\|, \tag{5.159}$$

for $\rho_{\mathbf{r}} \in (0,1)$ some constant independent of k, then from (5.154) we obtain an analogous result for convergence of the gradient in terms of the residual for a linear Φ:

$$\left\|G_{sF}\left(\mathbf{v}_i^{(k)}\right)\right\| \leq \left(L\sqrt{2}\left\|\mathcal{A}^{-1}\right\|\left\|\mathbf{r}^{(0)}\right\|\right)(\rho_{\mathbf{r}})^{(k)} + \left\|G_{sF}\left(\mathbf{u}_i\right)\right\|. \tag{5.160}$$

Tighter bounds hold under the assumptions of lemma 6, replacing $L\sqrt{2}$ by L and the generalized gradient of $F = f + g$, G_{sF} (5.145), by the gradient of f, ∇f.

In particular, we conclude that when using MGRIT with the GD fine-grid operator (5.30) for MP1 (5.15), and the PG fine-grid operator (5.32) for MP2 (5.23), minimizing the MGRIT residual implies minimizing the gradient of $\mathbf{v}_i^{(k)}$ to a similar accuracy as that of $\mathbf{u}_i$, with (5.160) providing a bound that can inform the number of iterations required to make $\|G_{sF}(\mathbf{v}_i^{(k)})\|$ close to $\|G_{sF}(\mathbf{u}_i)\|$. For example, assuming a desired accuracy of the minimizer of $\|G_{sF}(\mathbf{u}_{N_t})\| = \tau$, one could guarantee accuracy 2τ of the MGRIT approximation $\mathbf{v}_{N_t}^{(k_*)}$ with at most k_* iterations, where k_* is determined according to $\tau = L\sqrt{2}\|\mathcal{A}^{-1}\|\|\mathbf{r}^{(0)}\|(\rho_r)^{k_*}$.

5.7 Numerical results

Here, we present results of our parallel-in-iteration algorithm applied to the optimization problems MP1 (5.15) and MP2 (5.23) discretized with the optimization methods discussed in subsection 5.3.1 and subsection 5.3.2, respectively. It is important to note that we are not currently implementing MGRIT in parallel, which is left for future work; the goal of this thesis is to present the method for optimization problems and investigate its convergence speed (the number of MGRIT iterations required to obtain an accurate solution) numerically, which is done using a serial implementation. In a later section, section 5.8, we also provide estimates for speedup in a parallel setting. We note that while MGRIT by construction approximates the entire trajectory $\{\mathbf{u}_i\}$, $0 \leq i \leq N_t$, of the sequential solve, we are for optimization problems really interested in calculating the final approximation $\mathbf{u}_{N_t}$, which is the MGRIT approximate solution at the final iteration point N_t, such that we give special importance to $\mathbf{u}_{N_t}$.

First, we solve the problem sequentially using gradient descent or proximal gradient descent. Next, we solve the same problem with MGRIT, i.e., we use MGRIT aiming to accelerate the computation of the fine-grid trajectory $\{\mathbf{u}_i\}$ in parallel. On coarse levels in the MGRIT hierarchy we use an implicit variant of the optimization method used on the finest level. We illustrate convergence results for a particular set of choices for the parameters N (dimension of the Laplacian matrix), m (coarsening factor) and $\widehat{\ell}$ (number of levels). Although we only present results for the fine-level step-size choice $s = 1/L$, further tests (not shown here for brevity) indicate that MGRIT performs similarly well for larger step sizes up to the stability limit of $2/L$.

For our initial examples, we chose $\widehat{\ell} = 2$ and $m = 4$. Later, we present results for different values of m, $\widehat{\ell}$ and N to analyze how MGRIT's performance is affected by the choices of these parameters. We choose 10^{-8} as the tolerance criteria for all iterative methods. For sequential solves, this applies to the reduction in the 2-norm of the gradient relative to its original value. For MGRIT solves, the tolerance criteria refers to the reduction on the 2-norm of the MGRIT residual, (4.4). For MP1, (5.15), we measure the gradient of the quadratic function f, while for MP2, (5.23), we measure the generalized gradient, (5.145), of the EOP function of the form $F := f + g$. The initial MGRIT guess $\mathbf{u}^0 \in \mathbb{R}^N$ is chosen to be random with components in $[0, 1]$, after which the condition $\mathbf{u}_0 = \mathbf{w}_0$ is enforced. In our numerical tests, we apply the standard linear MGRIT scheme, algorithm 6, for MP1 (5.15) and the FAS-MGRIT scheme, algorithm 7, for MP2 (5.23). For both MP1 and MP2, MGRIT is iterated until the 2-norm of the residual (4.4) is reduced by 8 orders of magnitude from its initial value.

Remark 16 (On determining the number of fine-grid time points)**.** We note that the sequential solve provides us with N_t, exactly the number of iterations required for convergence within the given tolerance threshold, which we then use to specify the MGRIT fine-grid time domain to be $\left[0, \frac{N_t}{L}\right]$, with $N_t + 1$ fine-grid time points. However, solving the problem sequentially before solving it with MGRIT is of course not how we use the method in practice. Instead, the final time should be chosen adaptively, for example, using the following strategy.

1. Given some tolerance threshold to which we want to reduce the gradient to, we take an educated guess at the final time required to achieve this reduction. This guess can possibly be informed by theoretical or practically observed convergence rates of the underlying optimization method. Setting the MGRIT time window with this guess for the final time, we run MGRIT until either the gradient at the final point reaches the prescribed tolerance, or the gradient at the final point stalls.

2. In the case where the gradient reaches the prescribed tolerance, we either have accurately guessed the correct final time, such that the number of time points is close to what we would get by solving the problem sequentially first with the prescribed accuracy, or we have overestimated the final time such that we have too many time points. Otherwise, if the gradient at the final time point stalls, then we have underestimated the final time and do not have enough time points to achieve the desired gradient reduction. Iterating MGRIT past this point is redundant, since MGRIT will only reduce the residual of the underlying algebraic system (4.4), but without reducing the gradient.

3. In the latter case, we must start a new MGRIT solve across a new time interval, starting from the final time point of the halted solve and setting the solution at the final time point of the halted solve as the initial guess for the new solve's all-at-once system (4.3); essentially, this is the same as extending the original trajectory. Then, we take a new educated guess for the final time, which could be estimated based on the approximate convergence rate of the optimization method observed from the halted solve. Having set the new time horizon, we run MGRIT again and repeat this process until the desired gradient reduction is obtained.

Following this strategy, it may be that underestimating or overestimating the final time does not lead to performing any more work than when perfectly guessing the final time. However, we might lose some parallel speed-up efficiency, as in both cases we do more sequential work than when we work with the ideal number of time points, in which case

the parallelization is perfectly optimized. For our discussion on parallel speed-up efficiency, see section 5.8.

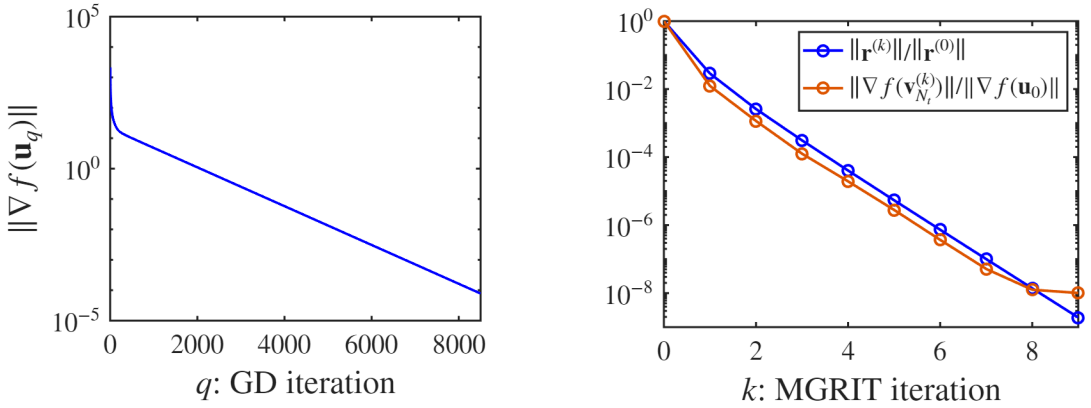
Remark 17 (Final time point N_t can be an F-point)**.** In our numerical experiments, we relax the assumption that N_t must be a C-point, i.e., $N_t = N_T m$. When $N_t \neq N_T m$, given N_t and the chosen coarsening factor m, we allow for the final coarse-grid interval to have fewer than m points, so that we can have a fine grid of $N_t + 1$ points without satisfying $N_t = N_T m$. We note that whether N_t is an F- or C-point does not affect the number of MGRIT iterations required for convergence.

5.7.1 MGRIT for gradient descent applied to MP1

We solve (5.15) using GD, (5.30). The 1D Laplacian matrix $A_1 \in \mathbb{R}^{N\times N}$ in (2.5) with 2-norm L is taken with $N = 40$, the coarsening factor is chosen to be $m = 4$ and the number of levels to be $\widehat{\ell} = 2$. We also choose $\mathbf{b} \in \mathbb{R}^N$ to be random with components in $[0, 1]$. For MGRIT, we choose the proximal-point operator (5.31) as the coarse-grid operator. First, we solve the problem sequentially, with $N_t = 8,503$ iterations required for the convergence of the GD algorithm (5.30). We then generate the MGRIT fine grid $t_i = i\Delta t$, $i = 0, \ldots, N_t$ with $\Delta t = 1/L$. Next, we generate the coarse-grid $T_j = j\Delta t$, $j = 0, 1, \ldots, N_T$, with $\Delta t = m\Delta t = 4/L$, such that the coarse grid has $N_T = 2,126$ points. MGRIT then solves (4.3). Convergence for the sequential and MGRIT solves is illustrated in figure 5.2.

In figure 5.2, we see the MGRIT convergence both in terms of the MGRIT residual and the gradient at the final time point N_t. We recall that the MGRIT convergence criteria is given in terms of space-time residual (4.4), but using the gradient of solution at the final time point $\mathbf{v}_{N_t}^{(k)}$ of each MGRIT iteration as reference, we see that the gradient of the function is also reduced by a similar factor. Recalling remark 15, we can interpret (5.160) as follows: if we look at a semi-log plot of $\left\|\nabla f\left(\mathbf{v}_i^{(k)}\right)\right\|$ as a function of k, as in figure 5.2(b), then it will be upper bounded by a straight line with slope equal to $\rho_{\mathbf{r}} \in (0, 1)$, which is the slope of the residual curve, up until the point where $\|\nabla f(\mathbf{u}_i)\| \approx (L\,\|\mathcal{A}^{-1}\|\,\|\mathbf{r}^0\|)\,(\rho_{\mathbf{r}})^k$, at which point the curve flattens out to a constant value of $\|\nabla f(\mathbf{u}_i)\|$. Therefore, the numerical results presented here are consistent with our theoretical predictions in section 5.6.

In figure 5.3, we further explore the behavior of the gradient and the residual (4.4) as MGRIT converges. As expected, we see that as k increases, the gradient curve for the kth iteration in figure 5.3(a) approximates the curve in figure 5.2(a) increasingly closely. We note that in both figure 5.3(a) and (b), the gradient and MGRIT residual are reduced by



(a) Sequential convergence

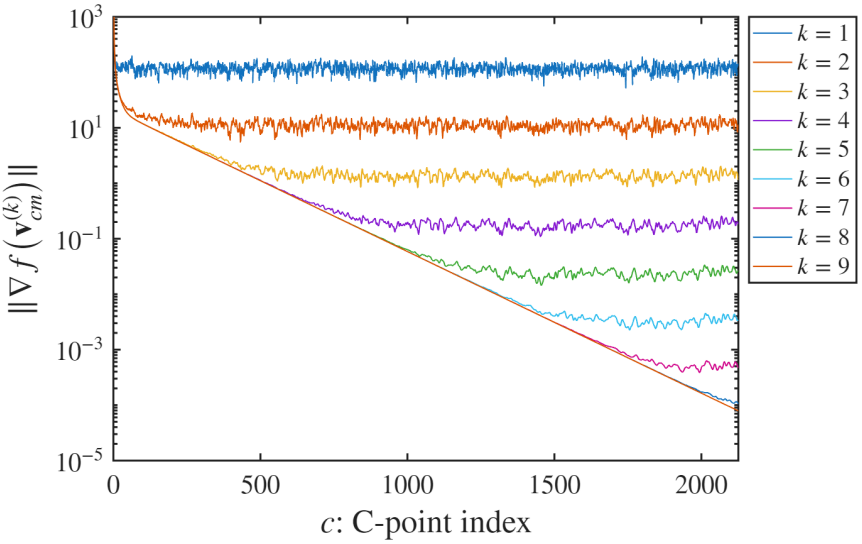
(b) MGRIT convergence

Figure 5.2: **(a)**: Convergence of MP1, (5.15), with a standard sequential solve using GD method (5.30); $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. **(b)**: Convergence of MP1, (5.15), with MGRIT using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\mathbf{r}^{(k)}$ denotes residual (4.4) at iteration k, $\nabla f(\mathbf{v}_{N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point N_t and MGRIT iteration k, $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual.

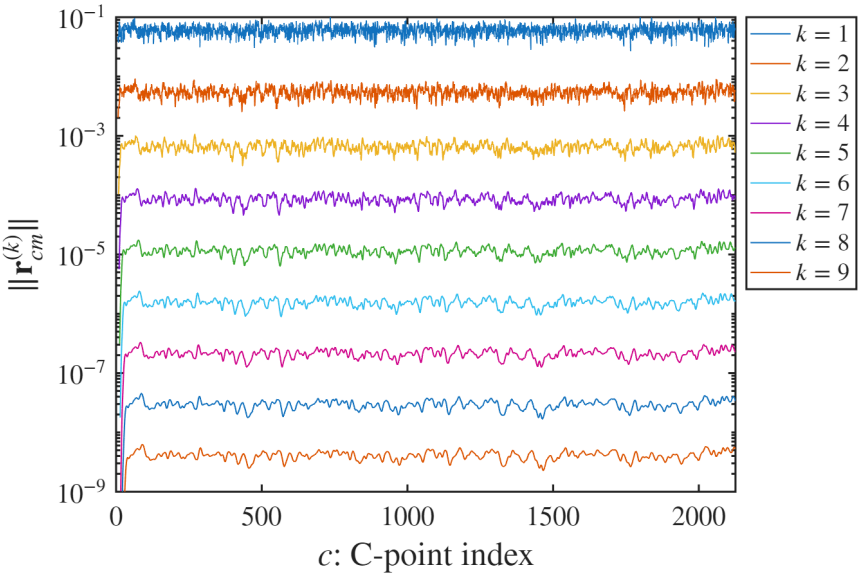
a consistent factor for each iteration. We also note that the behavior on the initial time points of each iteration in figure 5.3(b) is consistent with the exactness property of MGRIT with FCF-relaxations; for $m = 4$, at the kth MGRIT iteration, the first $2mk = 8k$ fine points are solved exactly. figure 5.3(b) shows that MGRIT converges in a small number of iterations, which is crucial for obtaining parallel speedup; see section 5.8.

In table 5.1, we present MGRIT convergence results for MP1 (5.15) with different coarsening factors m, number of MGRIT levels $\widehat{\ell}$ and different sizes for the matrix A_1 in (2.5). We note that the larger the coarsening factor, the smaller the maximum number of allowed levels.

From subsection 5.4.2 we recall MP1's (5.15) connection to the 1D heat equation, for which MGRIT convergence behaviour is well understood [27, 24]. For parabolic problems, MGRIT has been observed to have a desirable property of multigrid convergence known as h-independent convergence [94], which means that the convergence speed does not depend on the size of the fine grid.



(a) Gradient convergence by iteration



(b) Residual convergence by iteration

Figure 5.3: **(a)**: 2-norm of $\nabla f\left(\mathbf{v}_{cm}^{(k)}\right)$, where ∇f is the gradient of f, and $\mathbf{v}_{cm}^{(k)}$ is the approximate solution at coarse time point cm, $c = 0, \ldots, N_T$, and iteration k. **(b)**: 2-norm of $\mathbf{r}_{cm}^{(k)}$, where $\mathbf{r}_{cm}^{k}$ is the residual (4.4) at coarse time point jm, $c = 0, \ldots, N_T$, and iteration k.

Table 5.1: MGRIT convergence results for different values of N, m and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid.

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$
$m=4$	9	9	9	9	9	9
$m=16$	8	8	8			
$m=64$	8	8				
$m=256$	7					

(a) $N=40$, $N_t=8,503$.

	$\ell=2$	$\ell=3$	$\ell=4$	$\ell=5$	$\ell=6$	$\ell=7$	$\ell=8$
$m=4$	9	9	9	9	9	9	9
$m=16$	8	8	8				
$m=64$	8	8					
$m=256$	7						

(b) $N=60$, $N_t=18,192$.

	$\ell=2$	$\ell=3$	$\ell=4$	$\ell=5$	$\ell=6$	$\ell=7$	$\ell=8$
$m=4$	9	9	9	9	9	9	9
$m=16$	8	8	8				
$m=64$	8	8					
$m=256$	8						
$m=1024$	7						

(c) $N=80$, $N_t=30,299$.

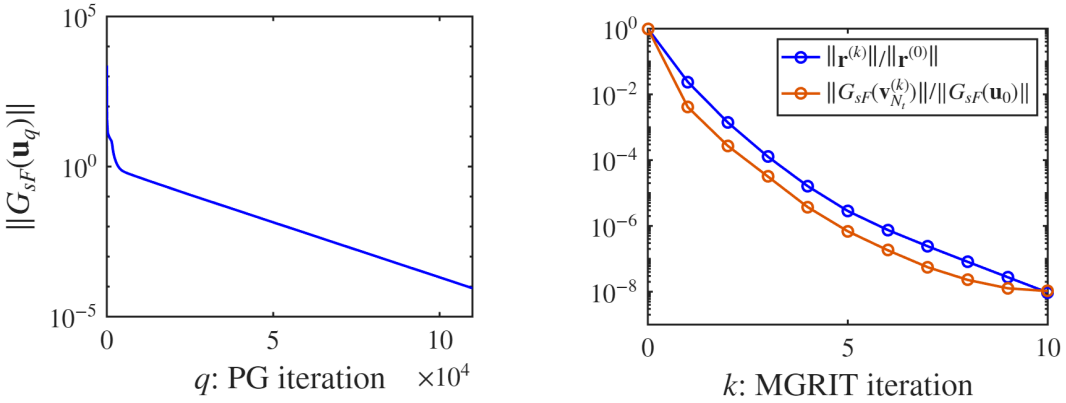
5.7.2 MGRIT for proximal gradient applied to MP2

We now consider solving the EOP problem, MP2 (5.23), using PG method (5.32). For MGRIT, we choose the AP operator (5.35) to be the coarse-grid operator. We noted in subsection 5.3.2 how the sequence generated by AP does not necessarily converge to the same set of minimizers as that of PG. However, the similarity between these operators justifies the use of the AP method as the coarse-grid method (recall discussions in subsection 5.4.2 and section 5.5 on choosing coarse-grid operators).

Numerical results for MP2-1D

For the MP2-1D example, we consider MP2 in (5.23) with $d=1$ and $N=256$, with MGRIT parameters $m=4$ and $\widehat{\ell}=2$. We again start by presenting results for the sequential and

MGRIT convergence in figure 5.4. In this example, the PG method requires $N_t = 109,888$ iterations to reach the prescribed halting tolerance, i.e., that the generalized gradient of F, G_{sF} in (5.145), be reduced by 8 orders of magnitude. MGRIT reaches the prescribed residual tolerance in 10 iterations.

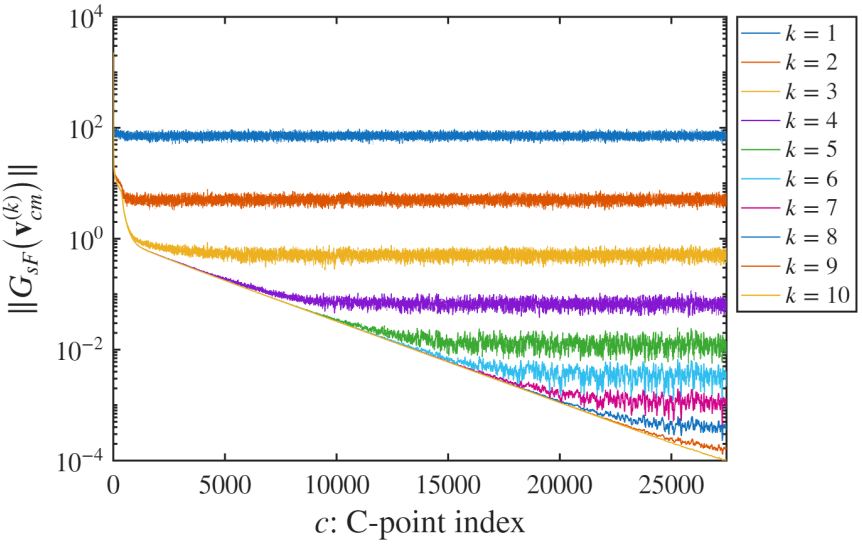


(a) Sequential convergence **(b)** MGRIT convergence

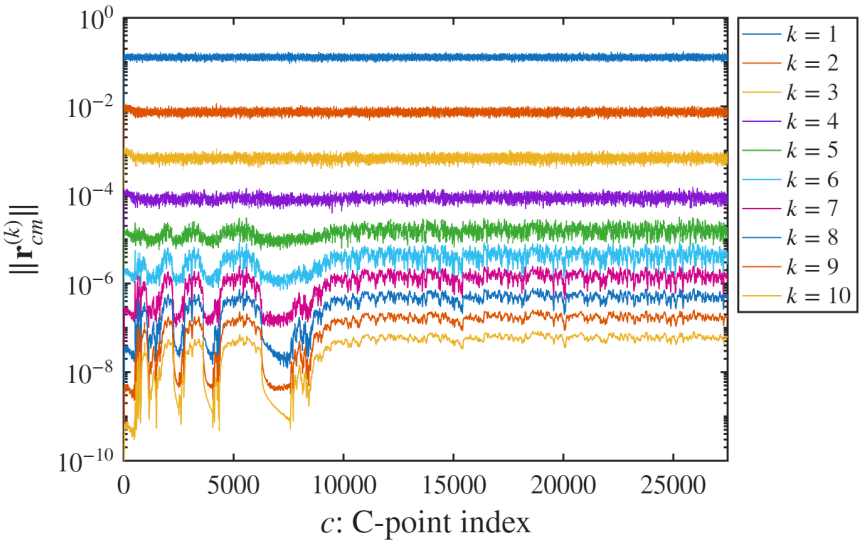
Figure 5.4: **(a)**: Convergence of MP2-1D, (5.23), with a standard sequential solve using PG method (5.32); $G_{sf}(\mathbf{u}_q)$ denotes the generalized gradient (5.145) of the qth sequential iterate. **(b)**: Convergence of MP2-1D, (5.23), with MGRIT using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $\mathbf{r}^{(k)}$ denotes residual (4.4) at iteration k, $G_{sF}(\mathbf{v}_{N_t}^{(k)})$ denotes generalized gradient (5.145) of the approximate solution at the final time point N_t and MGRIT iteration k, $G_{sF}(\mathbf{u}_0)$ denotes the generalized gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual.

As in subsection 5.7.1, it is interesting to investigate the relation between the MGRIT residual (4.4) and the generalized gradient (5.145); see figure 5.5. Much like in figure 5.3(a), the gradient in figure 5.5(a) is consistently reduced across iterations. The space-time residual in figure 5.5(b), however, has an unusual behavior over the first coarse time points as the number of iterations increases. To further explore this, in figure 5.6 we look at the spatial components of MGRIT residual vector $\mathbf{r}^{(k)}$ (4.4) at iteration k in order to relate the residual to the exact solution $\widehat{u}$ and obstacle ϕ of the 1D EOP in (5.17); see figure 5.1.

With the solution $\widehat{u}$ and obstacle ϕ in figure 5.1 overlaid on the residual (4.4) in figure 5.6, we see that the residual decreases more slowly and less smoothly around the spatial coordinates where the membrane attaches to the obstacle and the problem is nonsmooth. This nonsmoothness causes the slope of the convergence curves in figure 5.4(b) to taper off, compared to figure 5.2(b). Looking back at figure 5.5(b), we observe that for large k, when MGRIT approaches the exact trajectory $\{\mathbf{u}_i\}$, the MGRIT residual is initially



(a) Gradient convergence by iteration



(b) Residual convergence by iteration

Figure 5.5: (a): 2-norm of $G_{sF}\left(\mathbf{v}_{cm}^{(k)}\right)$, where G_{sF} is the generalized gradient of F (5.145), and $\mathbf{v}_{cm}^{(k)}$ is the approximate solution at coarse time point cm, $c = 0, 1, \ldots, N_T$ and iteration k. **(b)** 2-norm of $\mathbf{r}_{cm}^{(k)}$, where $\mathbf{r}_{cm}^{(k)}$ is the residual (4.4) at coarse time point cm, $c = 0, 1, \ldots, N_T$, and iteration k.

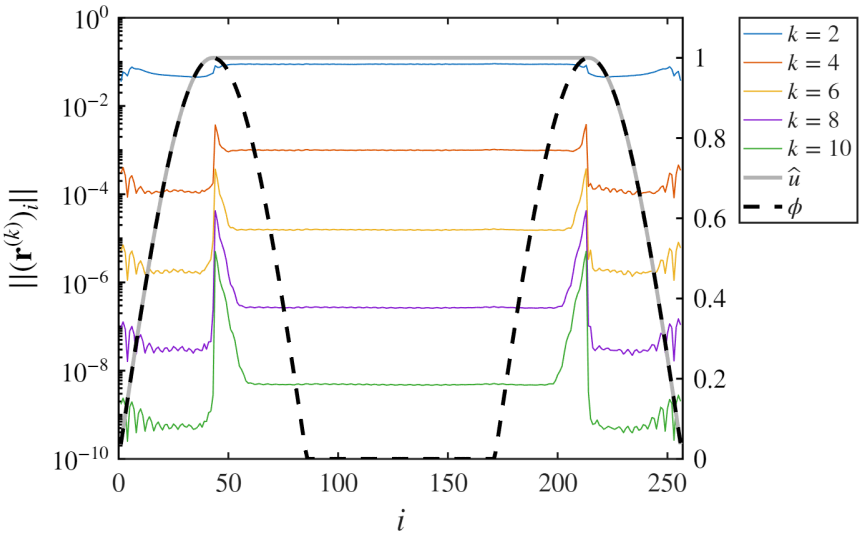


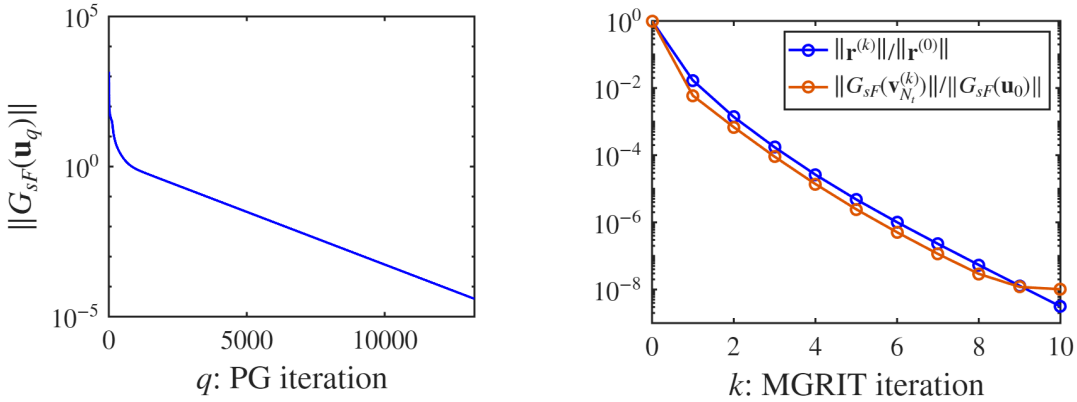
Figure 5.6: Left axis: 2-norm of $(\mathbf{r}^{(k)})_i$, where $(\mathbf{r}^{(k)})_i$ is the ith spatial component of residual (4.4) at iteration k. **Right axis:** $\widehat{u}$ is the exact solution and ϕ is the obstacle of the 1D EOP, as in figure 5.1. The i points form a 256-point discretization of the interior of the domain $\Omega = [0, 3\pi]$ of both $\widehat{u}$ and ϕ.

nonsmooth as a function of time point. This reflects that, at early time points, the set of spatial points where the nonsmooth constraint is active is not fully determined yet. Once the active set is fixed, the residual behaves similarly to how it does for MP1 (5.15), as seen in figure 5.3(b); we relate this to the discussion in subsection 5.5.2.

Numerical results for MP2-2D

For the MP2-2D example, we consider MP2 in (5.23) with $d = 2$ and $N^2 = 64^2$, with MGRIT parameters $m = 4$ and $\widehat{\ell} = 2$. As in subsection 5.7.1, we start by presenting the sequential and MGRIT convergence results in figure 5.7, noting that the sequential solver required $N_t = 13,256$ iterations to converge. We then compare the generalized gradient (5.145) and the MGRIT residual (4.4) at the coarse time points in figure 5.8. As with figure 5.6 for the MP2-1D, we also look at the 2-norm of the spatial components of the residual (4.4) in figure 5.9. This time, in order to present a 3D plot, we only present the residual at the final iteration $k = 10$.

Comparing figure 5.9 to figure 5.1, we see a similar relationship between the residual, the solution and the obstacle as we did with figure 5.6 and figure 5.1 for MP2-1D. To



(a) Sequential convergence

(b) MGRIT convergence

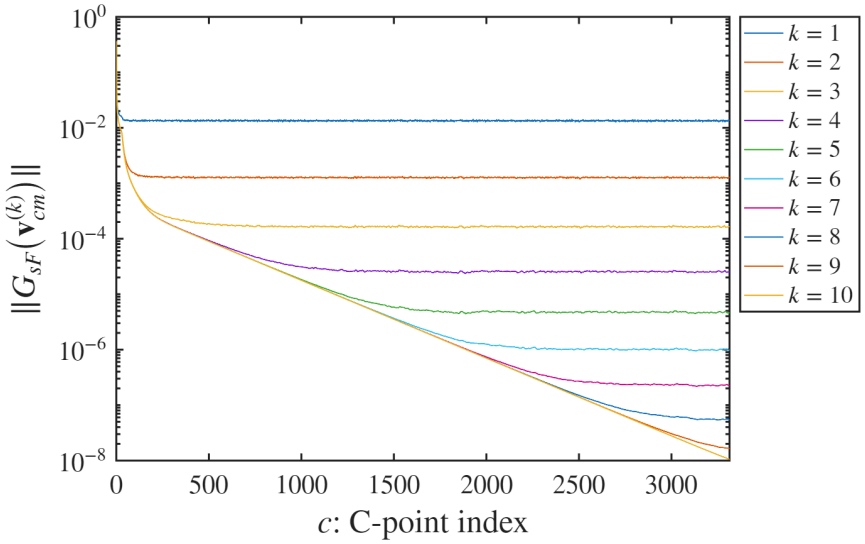
Figure 5.7: **(a)**: Convergence of MP2-2D, (5.23), with a standard sequential solve using PG method (5.32); $G_{sf}(\mathbf{u}_q)$ denotes generalized gradient (5.145) of the qth sequential iterate. **(b)**: Convergence of MP2-2D, (5.23), with MGRIT using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $\mathbf{r}^{(k)}$ denotes the residual (4.4) at iteration k, $G_{sF}(\mathbf{u}_{N_t}^k)$ denotes the generalized gradient (5.145) of the approximate solution at the final time point N_t and MGRIT iteration k, $G_{sF}(\mathbf{u}_0)$ denotes the generalized gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual.

better illustrate this relationship, we present countours plots in figure 5.10, from which we observe that the residual of the final MGRIT iteration is larger in the regions where the membrane attaches to the obstacle, as with the 1D case. Overall, we observe that MGRIT for MP2-2D behaves similar to that described for MP2-1D in subsection 5.7.1, and MGRIT convergence for both MP2-1D and MP2-2D behave according to the theory in section 5.6.

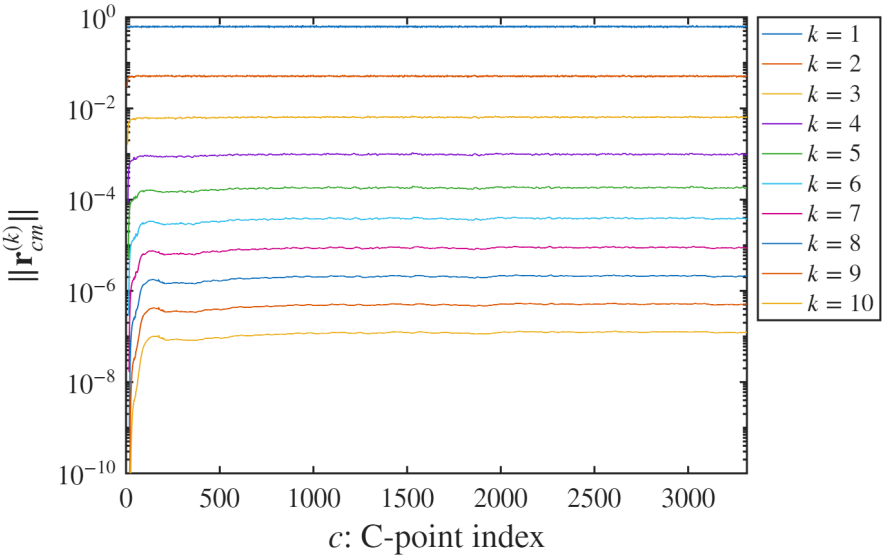
As with MP1 in table 5.1, we investigate the convergence results for MP2, (5.23), with different coarsening factors m, number of MGRIT levels $\widehat{\ell}$ and different sizes for the matrices A_1 in (2.5) for MP2-1D and A_2 in (2.9) for MP2-2D. Overall, MGRIT consistently converges in a small number of iterations, as for MP1 in table 5.1, and we obtain iteration numbers that are almost independent of problem size.

5.8 MGRIT parallel speedup estimates

In our numerical results, we verified that MGRIT was successful in solving both MP1, (5.15), and MP2, (5.23), in a small number of iterations. However, each MGRIT iteration does considerably more work than a full sequential solve. Since we are interested in the



(a) Gradient convergence by iteration



(b) Residual convergence by iteration

Figure 5.8: (a): 2-norm of $G_{sF}\left(\mathbf{v}_{cm}^{(k)}\right)$, where G_{sF} is the generalized gradient of F, (5.145), and $\mathbf{v}_{cm}^{(k)}$ is the approximate solution vector at coarse time point cm, $c = 0, 1, \ldots, N_T$, and iteration k. **(b)**: 2-norm of $\mathbf{r}_{cm}^{(k)}$, where $\mathbf{r}_{cm}^{(k)}$ is residual (4.4) at coarse time point cm, $c = 0, 1, \ldots, N_T$, and iteration k.

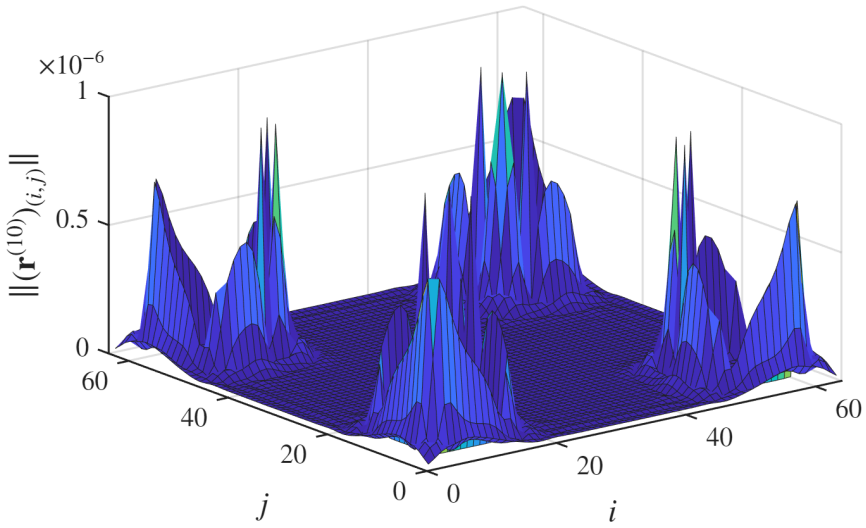


Figure 5.9: 2-norm of $(\mathbf{r}^{(10)})_{(i,j)}$, where $(\mathbf{r}^{(10)})_{(i,j)}$ is the (i,j)th spatial component of residual (4.4) for the 10th and final MGRIT iteration.

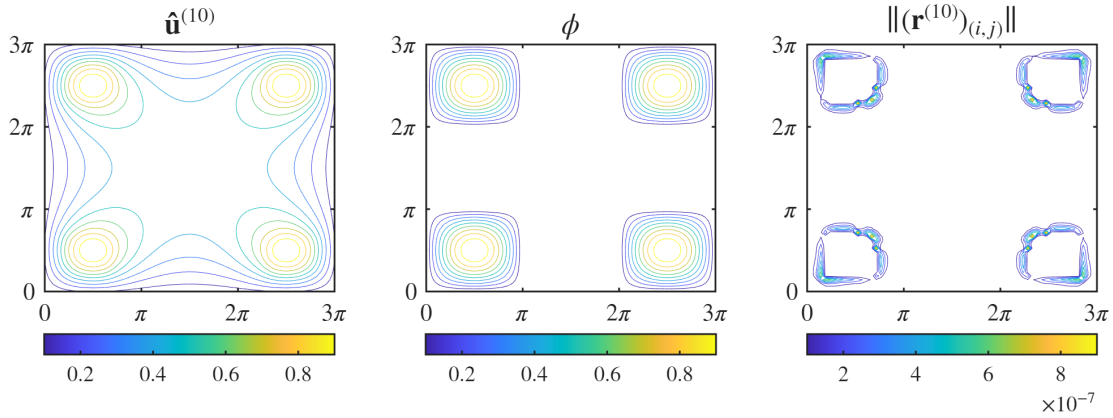


Figure 5.10: Contour plots of the approximate solution $\widehat{\mathbf{u}}$ and obstacle ϕ for the 2D EOP, as in figure 5.1, and the residual of the 10th and final MGRIT iteration in figure 5.9.

wall-clock time reduction in solving in parallel, we need to compute the relative cost of the MGRIT solve compared to the sequential solve, and estimate how much time would be saved by solving the problem in parallel. For that, we must first derive a formula for estimating the wall-clock time speedup as a function of the relevant parameters. Based on [83] and [69], which provide speedup estimates for the parareal method, we derive a formula for the MGRIT speedup estimate. In the following estimates, we assume that

Table 5.2: MP2-1D convergence results for different values of n, m and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid.

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$	$\widehat{\ell}=8$
$m=4$	10	10	11	12	12	12	13
$m=16$	8	9	10	10			
$m=64$	10	10	10				
$m=256$	10	10					
$m=1024$	11						

(a) $N=256$, $N_t=109,888$.

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$	$\widehat{\ell}=8$	$\widehat{\ell}=9$	$\widehat{\ell}=10$
$m=4$	9	12	11	11	12	12	12	12	12
$m=16$	10	10	11	11					
$m=64$	8	10	10						
$m=256$	9	9							
$m=1024$	10								

(b) $N=512$, $N_t=369,612$.

Table 5.3: MP2-2D convergence results for different values of n, m and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid.

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$
$m=4$	10	13	13	13	13	13
$m=16$	11	12	12			
$m=64$	10	10				
$m=256$	9					

(a) $N^2=64^2$, $N_t=13,256$.

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$	$\widehat{\ell}=8$
$m=4$	10	10	10	10	11	11	11
$m=16$	9	9	9				
$m=64$	9	9					
$m=256$	9						

(b) $N^2=128^2$, $N_t=46,431$.

parallel communication costs are negligible, as assumed for the parareal speedup estimates in [69, 83].

5.8.1 Estimating 2-level MGRIT wall-clock time speedup

First, we consider a 2-level MGRIT scheme as in section 4.1. Let N_f denote the number of time intervals on the fine-grid level $\ell = 1$, and let t_f denote the wall-clock time required to apply the fine-grid time-stepping operator Φ once. Similarly, let N_c denote the total number of time intervals on the coarse-grid level $\ell = 2$, and let t_c denote the wall-clock time required to apply the coarse-grid time-stepping operator Ψ once. The coarsening factor, m, is given by $m = \frac{N_f}{N_c}$. Let N_p denote the number of processors; full parallelization is obtained when we take $N_p = N_c = \frac{N_f}{m}$, such that we allot each coarse-grid step to its own processor, along with m fine-grid steps. Finally, let N_{it} be the number of MGRIT iterations required for convergence to a set tolerance.

The speedup, given by the ratio of the computation time for the sequential solve and the MGRIT solve, is given by

$$S_2\left(N_p\right) = \frac{N_f t_f}{N_{it}\left(N_c t_c + \frac{2N_f t_f}{N_p}\right)}. \tag{5.161}$$

The numerator $N_f t_f$ represents the total cost of the sequential solve. In the denominator, we have the cost of the MGRIT solve: for each of the N_{it} MGRIT iterations, the $\frac{2N_f t_f}{N_p}$ term accounts for the fine-grid cost of the parallel FCF-relaxation and correcting the coarse-grid solution, which in total amount to the cost of two full sequential solves that are solved in parallel by N_p processors, while the $N_c t_c$ term accounts for the cost of computing the C-point residual. Recalling that $\frac{N_f}{N_c} = m$ and assuming full parallelization (i.e., $N_p = \frac{N_f}{m}$), we rearrange (5.161) to find

$$S_2\left(m\right) = \frac{N_f t_f}{N_{it}\left(N_c t_c + \frac{2N_f t_f}{N_p}\right)} = \frac{1}{N_{it}\left(\frac{1}{m}\frac{t_c}{t_f} + \frac{2}{N_p}\right)} = \frac{1}{N_{it}\left(\frac{1}{m}\frac{t_c}{t_f} + \frac{2m}{N_f}\right)}. \tag{5.162}$$

In section 5.7, we observed that N_{it} does not vary significantly with m, so in what follows we assume it is a constant independent of m. We further assume t_c is independent of m, such that setting the derivative of S_2 with respect to m equal to zero then yields

$$S_2'\left(m\right) = 0 \iff -\frac{1}{m^2}\frac{t_c}{t_f} + \frac{2}{N_f} = 0 \iff m = \sqrt{\frac{N_f}{2}\frac{t_c}{t_f}}, \tag{5.163}$$

where for simplicity we assume the minimizer $\sqrt{\frac{N_f}{2}\frac{t_c}{t_f}}$ is an integer, such that we can choose $m = m^* := \sqrt{\frac{N_f}{2}\frac{t_c}{t_f}}$ as the optimal coarsening factor; in our results in the tables below, we round $\sqrt{\frac{N_f}{2}\frac{t_c}{t_f}}$ to the nearest integer. Denote $\alpha := \frac{t_c}{t_f}$, such that $m^* = \sqrt{\frac{\alpha N_f}{2}}$. Taking $m = m^*$ in (5.162), and simplifying gives

$$S_2\left(m^*\right) = \frac{1}{N_{it}\left(\frac{1}{\sqrt{\frac{\alpha N_f}{2}}}\alpha + 2\frac{\sqrt{\frac{\alpha N_f}{2}}}{N_f}\right)} = \frac{1}{N_{it}\left(\sqrt{\frac{2\alpha}{N_f}} + \sqrt{\frac{2\alpha}{N_f}}\right)} = \frac{\sqrt{N_f}}{N_{it}2\sqrt{2\alpha}}. \tag{5.164}$$

We observe that the optimal speedup is a function of N_f, N_{it} and $\alpha := \frac{t_c}{t_f}$. Since the optimal speedup is proportional to $\sqrt{N_f}$, it is clear that more speedup can be obtained and MGRIT will be more effective when the fine-level optimization method converges slowly and requires a large N_f. Similarly, to keep N_{it} small it is important to choose a coarse-grid operator that closely matches the fine operator, see [24, 21]. Finally, α can be brought close to 1 by making t_c as close as possible to t_f, for example, by optimizing the implicit computations done with the coarse-grid operator or by substituting it by a different one with smaller t_c, without increasing N_t.

5.8.2 Estimating 3-level MGRIT wall-clock time speedup

Next, we introduce a third level, $\ell = 3$, which is obtained from coarsening the $\ell = 2$ coarse grid level by the same coarsening factor m. Associated with $\ell = 3$ is the time-stepping operator Ψ^3, defined according to (5.59) (i.e., an operator similar to Ψ with an m times larger step size), which takes $m^2\Delta t$-sized steps. Let N_{cc} be the number of time intervals on the $\ell = 3$ level, and let t_{cc} denote the wall-clock time required to compute one Ψ^3 step. We assume that $t_{cc} = t_c$, since we use the same implicit scheme on both coarse levels $\ell = 2$ and $\ell = 3$, and $N_{cc} = \frac{N_c}{m} = \frac{N_f}{m^2}$, assuming N_{cc} is an integer.

For $\widehat{\ell} = 3$, we get a speedup estimate formula of

$$S_3\left(N_p\right) = \frac{N_f t_f}{N_{it}\left(N_{cc}t_{cc} + \frac{3N_c t_c}{\min\{N_p, N_c/m\}} + \frac{2N_f t_f}{N_p}\right)}. \tag{5.165}$$

The reasoning is similar to that for $\widehat{\ell} = 2$, with the difference that here we replace the original coarse-grid work in (5.161), with cost $N_c t_c$, with a new 2-grid MGRIT solve. Note

that on the $\ell = 2$ level, a post F-relaxation step must be done to distribute the correction from the coarse-grid solve in $\ell = 3$, such that the total relaxation cost on the $\ell = 2$ grid is roughly that of 3 sequential coarse solves. Since there are only $\frac{N_c}{m}$ intervals on the first coarse level, we can use at most $\frac{N_c}{m}$ processors on that level. Assuming full parallelization on the fine level $N_p = N_c$, we have that $\min\{N_p, \frac{N_c}{m}\} = \min\{N_c, \frac{N_c}{m}\} = \frac{N_c}{m}$. In that case,

$$S_3\left(N_p\right) = \frac{N_f t_f}{N_{it}\left(N_{cc}t_{cc} + \frac{3N_c t_c}{N_c/m} + \frac{2N_f t_f}{N_c}\right)} = \frac{1}{N_{it}\left(\frac{1}{m^2}\frac{t_c}{t_f} + \frac{2m}{N_f}\left(1 + \frac{3}{2}\frac{t_c}{t_f}\right)\right)}. \tag{5.166}$$

Following a similar reasoning to that for finding the optimal 2-level coarsening factor (5.163), we set the derivative of S_3 with respect to m equal to zero and get

$$S_3'\left(N_p\right) = 0 \iff -\frac{2}{m^3}\frac{t_c}{t_f} + \frac{2}{N_f} + \frac{2}{N_f}\frac{3}{2}\frac{t_c}{t_f} = 0 \iff m = \sqrt[3]{\frac{N_f}{t_f/t_c + 3/2}}. \tag{5.167}$$

Assuming $\sqrt[3]{\frac{N_f}{t_f/t_c+3/2}}$ is a integer, we take $m = m^* := \sqrt[3]{\frac{N_f}{t_f/t_c+3/2}}$ to be the optimal coarsening factor; we take the nearest integer, otherwise. Evaluating the 3-level speedup (5.166) with $m = m^*$ and substituting $\frac{2}{(m^*)^2}\frac{t_c}{t_f} = \frac{2m^*}{N_f}\left(1 + \frac{3}{2}\frac{t_c}{t_f}\right)$ by (5.167) yields

$$S_3(m^*) = \frac{1}{N_{it}\left(\frac{1}{(m^*)^2}\frac{t_c}{t_f} + \frac{2m^*}{N_f}\left(1 + \frac{3}{2}\frac{t_c}{t_f}\right)\right)} = \frac{1}{N_{it}\left(\frac{1}{(m^*)^2}\frac{t_c}{t_f} + \frac{2}{(m^*)^2}\frac{t_c}{t_f}\right)}. \tag{5.168}$$

Rearranging the fraction, substituting $m^* = \sqrt[3]{\frac{N_f}{t_f/t_c+3/2}}$ and $\alpha = t_c/t_f$ then yields

$$S_3(m^*) = \frac{(m^*)^2}{N_{it}3\alpha\gamma^{1/3}} = \frac{(m^*)^2}{N_{it}3\alpha\gamma} = \frac{N_f^{2/3}}{3\sqrt{3}N_{it}\alpha(1/\alpha + 3/2)^{2/3}}. \tag{5.169}$$

Much like (5.164), the optimal 3-level speedup is a function only of N_f, N_{it} and $\alpha = t_c/t_f$, such the considerations made for 2-level case also apply here. We note that this analysis can be further extended for more levels, but here we restrict ourselves to the 2- and 3-level cases. For a large number of levels, however, we leave a heuristic for choosing the coarsening factor, as described in the following remark.

Remark 18 (Coarsening factor heuristic for a large number of levels)**.** Assuming that $N_{\widehat{\ell}c} = N_f/m^{\widehat{\ell}}$ denotes the number of points on the coarsest level $\widehat{\ell}$, since we must have $N_{\widehat{\ell}c} \geq 1$, we find that

$$m^{\widehat{\ell}} \leq N_f, \tag{5.170}$$

which implies that m is bounded by

$$m \leq N_f^{1/\widehat{\ell}}. \tag{5.171}$$

By taking the base-m logarithm of (5.170), we find

$$\widehat{\ell} \leq \log_m (N_f). \tag{5.172}$$

We note that as $\widehat{\ell} \to \log_m N_f$, the range of permissible values of m shrinks to $[2, 2+c]$ for some small integer c. Thus, we intuit that for a large number of levels (relative to the number of fine-grid points), we may skip a rigorous estimation of the optimal coarsening factor and simply take the largest possible coarsening factor as a rough estimate for the optimal coarsening factor.

5.8.3 MGRIT parallel speedup estimates for MP1 and MP2

Using the formulas derived for 2 and 3 levels, we now present the computed MGRIT parallel speedup estimates for MP1 in (5.15) and MP2 in (5.23) in the following tables. First, we compute m^* for $\ell = 2$ and $\ell = 3$ using (5.163) and (5.167), respectively; if the resulting value is not an integer, we round it to the nearest integer and call that m^* instead. We choose $m = m^*$ as the coarsening factor for the MGRIT solve, which gives us N_{it}^* iterations in our numerical experiments; note that this is not the optimal number of MGRIT iterations, but simply the value of N_{it} obtained experimentally when using $m = m^*$. We then use $m = m^*$ and $N_{it} = N_{it}^*$ in the speedup formulas (5.162) and (5.166), where we also use an empirically estimated $\alpha = \frac{t_c}{t_f}$.

Next, we consider a hypothetical *ideal* case in which $\frac{t_c}{t_f} = 1 = \widehat{\alpha}$. Even if $\frac{t_c}{t_f} = 1$ is unlikely, this provides a lower bound for the value of the optimal coarsening factor m^*, since m^* in (5.163) and (5.167) is increasing with respect to $\frac{t_c}{t_f}$. Following the previously outlined process, we compute the optimal $\widehat{m}^*$, and then obtain $\widehat{N_{it}}^*$ numerically, which are used alongside $\widehat{\alpha} = 1$ for the 2- and 3-level speedup formulas to provide an estimated upper bound for speedup. We note that since the fine grid, $t_i = i\Delta t$, $i = 0, 1, \ldots, N_t$, has $N_t + 1$ points, there are $N_f = N_t$ fine-grid intervals. Similarly, $N_c = N_T$.

First, we note that 2-level MGRIT is predicted to provide speedup over sequential solves in all of our estimates. For most cases, the estimated speedups for $\ell = 3$ were considerable improvements over the $\ell = 2$ speedup. For all cases, increasing N led to improved speedup. As the sizes N of the Laplacian matrices (2.5) and (2.9) increase, more sequential iterations

Table 5.4: Speedup estimates for MP1, (5.15), different values of n and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 2.9$ for $n = 40$, $\alpha = 3.1$ for $n = 60$, and $\alpha = 3.5$ for $n = 80$. N_p denotes the number of processors required to achieve optimal speedup.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	112	7	2.73	76
3	16	8	3.67	532

(a) $N = 40$, $N_t = 8,503$.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	167	8	3.37	109
3	21	8	5.92	860

(b) $N = 60$, $N_t = 18,050$.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	229	8	4.09	132
3	26	8	8.04	1155

(c) $N = 80$, $N_t = 30,018$.

Table 5.5: Speedup estimates for MP1, (5.15), for different values of n and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on a hypothetical lower bound for $\frac{t_c}{t_f}$ given by $\widehat{\alpha} = 1$. N_p denotes the number of processors required to achieve optimal speedup.

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	65	8	4.07	131
3	15	8	9.37	567

(a) $N = 40$, $N_t = 8,503$.

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	95	8	5.93	190
3	19	8	15.04	950

(b) $N = 60$, $N_t = 18,050$.

ℓ	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	123	8	7.65	245
3	23	8	22.04	1306

(c) $N = 80$, $N_t = 30,018$.

Table 5.6: Speedup estimates MP2-1D, (5.23), for different values of n and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 2.4$ for $N = 256$, and $\alpha = 3.9$ for $N = 512$. N_p denotes the number of processors required to achieve optimal speedup.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	363	10	7.56	303
3	39	10	21.12	2818

(a) $N = 256$, $N_t = 109,888$.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	849	10	10.88	436
3	59	9	33.05	6265

(b) $N = 512$, $N_t = 369,612$.

are needed until convergence is achieved, which in turn increases the size of the MGRIT time grids. Optimal speedup formula (5.164) shows that larger grids often benefit from more parallelization. On the other hand, increasing N is also correlated with an increased $\frac{t_c}{t_f}$ ratio, due to the cost of the implicit coarse-grid time-stepping operations increasing relatively more than that of the explicit fine-grid ones. However, in our experiments the larger $\frac{t_c}{t_f}$ ratios were offset by the increase in N_t, such that speedup was improved with larger N.

Table 5.7: Speedup estimates for MP2-1D, (5.23), for different values of n and $\widehat{\ell}$, using the PG operator (5.32) on the fine grid and the AP operator (5.35) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on an hypothetical lower bound for $\frac{t_c}{t_f}$ given by $\widehat{\alpha} = 1$. N_p denotes the number of processors required to achieve optimal speedup.

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	234	10	11.72	470
3	35	11	37.12	3140

(a) $N = 256$, $N_t = 109,888$.

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	430	10	21.49	860
3	53	9	104.03	6974

(b) $N = 512$, $N_t = 369,612$.

Table 5.8: Speedup estimates for MP2-2D, (5.23), for different values of n and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 13$ for $n^2 = 64^2$, and $\alpha = 7$ for $n^2 = 128^2$. N_p denotes the number of processors required to achieve optimal speedup.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	294	9	1.25	46
3	20	10	1.02	663

(a) $N^2 = 64^2$, $N_t = 13,256$.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	403	9	3.19	116
3	30	9	4.76	1548

(b) $N^2 = 128^2$, $N_t = 46,431$.

Table 5.9: Speedup estimates for MP2-2D, (5.23), for different values of n and $\widehat{\ell}$, using the PG operator (5.32) on the fine grid and the AP operator (5.35) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on an hypothetical lower bound for $\frac{t_c}{t_f}$ given by $\widehat{\alpha} = 1$. N_p denotes the number of processors required to achieve optimal speedup.

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	81	10	4.07	164
3	17	10	9.63	2732

(a) $N^2 = 64^2$, $N_t = 13,256$.

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	152	10	7.61	306
3	26	9	25.03	1786

(b) $N^2 = 128^2$, $N_t = 46,431$.

The results in table 5.8 are exceptions to the above. For the $N^2 = 64^2$ experiment, the estimated $\frac{t_f}{t_c}$ ratio of $\alpha = 13$ was very large. This is expected, since we are using 2D Laplacians (2.9) instead of 1D ones (2.5), thus increasing the relative cost of implicit steps. In fact, α was so large that the 2-level solve is predicted to perform better than the 3-level solve, likely since using 3 levels requires an additional post-relaxation step on the $\ell = 2$ coarse grid. However, when increasing the problem size to $n^2 = 128^2$, the $\frac{t_c}{t_f}$ ratio was reduced to $\alpha = 7$. To understand why this happens, recall that the fine-grid PG operator

(5.32) is $\Phi = \mathcal{P}_{sg} \circ \mathcal{G}_{sf}$, while the coarse-grid AP operator (5.35) is $\Psi = \mathcal{P}_{sg} \circ \mathcal{P}_{sf}$, with $s = 1/L$ for Φ and $s = m/L$ for Ψ. As N increases, the cost of the $\mathcal{P}_{sg}$ step in both Φ and Ψ accounts for a larger share of both t_c and t_f, such that the $\frac{t_c}{t_f}$ ratio is reduced. While still large, $\frac{t_c}{t_f}$ was reduced enough that the 3-level solve is estimated to provide a slight speedup improvement over the 2-level solve.

In particular, table 5.6 and table 5.8 exemplify the conditions for which our method performs best and worse, respectively; in the former, we have large N_t and small α, while the opposite is true for the latter. The implicit steps in our MGRIT implementation were calculated using LU-factorization with saved L and U factors to avoid having to repeatedly invert large matrices directly; improving this calculation can bring the $\frac{t_c}{t_f}$ ratio to lower, more beneficial levels, hence bringing speedup estimates closer to the hypothetical $\frac{t_c}{t_f} = 1$ results in table 5.5, table 5.7 and table 5.9. While not explored here, we expect speedups to potentially improve with more levels; this is supported by table 5.1, table 5.2 and table 5.3, in which we observe that the N_{it} remains fairly consistent across for all possible values ℓ. Finally, increasing the resolution of MP2-2D to $N^2 = 256^2$ or $N^2 = 512^2$ to match the tests for MP2-1D, would dramatically increase N_t, which would also be expected to substantially increase the speedup.

5.9 Conclusions

We have presented a framework for solving optimization problems all-at-once, in the sense that sequential solves are replaced by a parallel solver in which the sequential iterations are computed simultaneously. Using the MGRIT algorithm to parallelize over iteration of the optimization method instead of over time, we demonstrated a multilevel parallel-in-iteration method that can converge in a small number of iterations, and also provides considerable estimated speedup for both smooth and nonsmooth problems. Further improvements to this framework and the consideration of other model problems can be investigated by the continued exploration of the connection between optimization algorithms and ODEs, especially those related to diffusion-dominated problems for which MGRIT performs well. We note that by using MGRIT as the basis for our framework, we are somewhat limited by the fact that MGRIT attempts to approximate the entire trajectory of the sequential solve with high accuracy. While this is a necessity for its original intent of solving time-dependent ODEs, when the solutions at all time points are relevant, for optimization problems we are only interested in obtaining the final solution as fast as possible. Therefore, we would also like to develop a "looser" parallel-in-time framework in which the

requirement to compute an approximate solution with high accuracy is restricted to the final time point, which could allow us to provide more speedup than in our estimates for the MGRIT solve.

Chapter 6

Parallel-in-iteration optimization for momentum-accelerated algorithms

In this chapter, we continue immediately from the end of chapter 5. So far, we have only worked with standalone versions of gradient-based methods such as Gradient Descent (GD), (3.49), and Proximal Gradient Descent (PG), (3.84). However, in practice they are often replaced by momentum-accelerated variants, namely Nesterov's Accelerated Gradient Descent (NAGD) for GD and Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) for PG, which have improved convergence rates for (5.1) and (5.2), for the former and latter, respectively. Based on the momentum-acceleration mechanism employed by these methods, we are able to derive analogous momentum-accelerated variants for Proximal Point (PP) and Alternating Proximals (AP) as well, which we consider in principle as the implicit methods to be employed on the coarse grid(s). In this chapter, we explore the possibility of employing our parallel-in-iteration (P-in-I) framework from chapter 5 to parallelize over iterations of these momentum-accelerated algorithms for solving the model problems from section 5.2. By combining parallel speedup with momnetum

We note in advance that we are currently unsuccessful in this endeavour, with this chapter presenting exploratory theoretical and numerical analysis presented for the sake of future attempts at achieving the objectives we set for ourselves here.

6.1 Momentum-accelerated algorithms

6.1.1 Nesterov accelerated gradient descent (NAGD)

Gradient descent (see subsection 3.2.1) is often replaced by Nesterov's accelerated gradient descent method, referred to here as NAGD [70, 71]. Starting from an initial guess $\mathbf{u}_0 \in \mathbb{R}^N$ and using a stepsize $s = 1/L$, Nesterov's accelerated sequence $\{\mathbf{u}_k, \mathbf{v}_k\}$ is given by

$$\begin{aligned} \mathbf{u}_k &= \mathbf{v}_{k-1} - s\nabla f(\mathbf{v}_{k-1}), \\ \mathbf{v}_k &= \mathbf{u}_k + \beta_k(\mathbf{u}_k - \mathbf{u}_{k-1}). \end{aligned} \tag{6.1}$$

When the momentum parameter β_k is generated by the sequence $\beta_{k+1} = \frac{t_k - 1}{t_{k+1}}$, where $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$ starting from $t_1 = 1$ (such that $\beta_1 = 0$), the method achieves a sublinear convergence rate of $\mathcal{O}(1/k^2)$ for convex, L-smooth functions [71]. The simpler update rule $\beta_k = \frac{k-1}{k+2}$ is also frequently used; notably, this choice allows the NAGD iteration in (6.1) to be interpreted as the discretization of an Ordinary Differential Equation (ODE) [90] (see (6.8)). Other parameter sequences $\{\beta_k\}_{k\in\mathbb{N}}$ may also be considered, such as those proposed in [74].

Alternatively, we may consider a fixed momentum parameter, setting $\beta_k = \beta$ for all k. Note that if $\beta = 0$, we recover the standard GD update in (3.49). For minimizing a μ-strongly convex and L-smooth function using the optimal fixed stepsize $s = 1/L$, the optimal fixed momentum parameter is given by [71, Theorem 2.2.3]:

$$\beta^* = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}, \tag{6.2}$$

where $\kappa = L/\mu$ is the condition number of f; see definition 36. With this optimal β^*, NAGD converges linearly with a convergence factor of $1 - 1/\sqrt{\kappa}$ [71]. Consequently, $\mathcal{O}\left(\sqrt{\kappa}\log(1/\epsilon)\right)$ iterations are required to achieve an error bound of $f(\mathbf{u}_k) - f(\mathbf{u}^*) < \epsilon$.

Note that as the problem becomes increasingly ill-conditioned (i.e., $\kappa \to \infty$), the optimal momentum parameter approaches one; i.e., $\beta^* \to 1$. In particular, this asymptotic behaviour applies to the discrete Laplacian matrices in (2.5) and (2.9) as the grid size grows; i.e., as $N^d \to \infty$ for $d \in \{1, 2\}$.

Remark 19 (Polyak's heavy ball method)**.** Preceding NAGD is Polyak's Heavy Ball (HB) method [77, 78, 76]. The key difference between the NAGD and HB methods is that in

the HB method, the gradient is evaluated before the momentum update. That is, the HB update for $\{\mathbf{u}_k, \mathbf{v}_k\}$ is given by

$$\begin{aligned} \mathbf{v}_k &= \mathbf{u}_k + \beta_k \left(\mathbf{u}_k - \mathbf{u}_{k-1}\right), \\ \mathbf{u}_{k+1} &= \mathbf{v}_k - s\nabla f\left(\mathbf{u}_k\right). \end{aligned} \tag{6.3}$$

For minimizing a quadratic function of the form (5.24), the HB method has optimal fixed parameters [78, 76] given by

$$s^* = \frac{4}{(\sqrt{L} + \sqrt{\mu})^2}, \quad \beta^* = \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^2. \tag{6.4}$$

While the HB method works well for quadratic functions, unlike NAGD it fails to converge for general L-smooth, μ-strongly convex functions [60].

6.1.2 Nesterov-accelerated proximal point algorithm (NAPP)

As we discussed for GD in subsection 5.5.1, for us to employ NAGD as a candidate for fine-grid method in our P-in-I framework from chapter 5, we need an implicit variant of NAGD that accepts stepsizes larger than the $s > 2/L$ Courant-Friedrichs-Lewy (CFL) limit of GD, which is inherited by NAGD [71]. Then, following the heuristic discussed in remark 12 of replacing gradient steps by proximal steps, we consider what we call the Nesterov-accelerated Proximal Point (NAPP) method, which employs the same acceleration mechanism of NAGD, except for accelerating PP steps rather than GD steps. Starting from $\mathbf{u}_0 \in \mathbb{R}^N$ and letting $s > 0$, the NAPP sequence $\{\mathbf{u}_k, \mathbf{v}_k\}$ update is given by

$$\begin{aligned} \mathbf{u}_k &= P_{sf}(\mathbf{u}_{k-1}), \\ \mathbf{v}_k &= \mathbf{u}_k + \beta_k \left(\mathbf{u}_k - \mathbf{u}_{k-1}\right), \end{aligned} \tag{6.5}$$

where P_{sf} denotes the PP operator from (3.75). We note that a similar momentum-accelerated PP variant inspired by NAGD was introduced in [50], but it differs slightly from NAPP.

6.1.3 FISTA

To accelerate the PG algorithm (3.84), also known as Iterative Shrinkage-Thresholding Algorithm (ISTA), the FISTA [10] was developed as a momentum-accelerated variant that

features the same acceleration mechanism as NAGD (6.1). Starting from $\mathbf{u}_0 \in \mathbb{R}^N$ and letting $s = 1/L$, the FISTA sequence $\{\mathbf{u}_k, \mathbf{v}_k\}$ update is given by

$$\begin{aligned} \mathbf{u}_k &= P_{sg} \circ G_{sf}(\mathbf{u}_{k-1}), \\ \mathbf{v}_k &= \mathbf{u}_k + \beta_k \left(\mathbf{u}_k - \mathbf{u}_{k-1}\right), \end{aligned} \tag{6.6}$$

where $P_{sg} \circ G_{sf}$ is the PG operator from (3.84). In particular, our NAPP method, with update (6.5), is simply the FISTA update (6.6) for $f = 0$, such that the PG operator from (3.84) reduces to the PP operator from (3.75). Much like NAGD improves the convergence rate of GD, FISTA has an improved convergence of rate $\mathcal{O}(1/k^2)$ to a minimizer of (5.2), which is further improved to $\mathcal{O}((1 - 1/\sqrt{\kappa}))^k$ in the strongly convex case [9]. We note that in [9], this method is referred to by Fast Proximal Gradient Method (FPGM).

6.1.4 Nesterov-accelerated alternating proximals (NAAP)

For an implicit variant of FISTA to be employed for coarse-grid steps, we consider applying the NAGD-acceleration mechanism to accelerate AP steps; this yields what we call the Nesterov-accelerated Alternating Proximals (NAAP) method. Starting from $\mathbf{u}_0 \in \mathbb{R}^N$ and letting $s = 1/L$, the NAAP sequence $\{\mathbf{u}_k, \mathbf{v}_k\}$ update is given by

$$\begin{aligned} \mathbf{u}_k &= P_{sg} \circ P_{sf}(\mathbf{u}_{k-1}), \\ \mathbf{v}_k &= \mathbf{u}_k + \beta_k \left(\mathbf{u}_k - \mathbf{u}_{k-1}\right), \end{aligned} \tag{6.7}$$

where $P_{sg} \circ P_{sf}$ is the AP operator from (3.91).

6.2 Connections between momentum-accelerated optimization algorithms and differential equations

The connection between the momentum-accelerated optimization methods discussed here and ODEs still holds, although not as trivially as in section 5.4. For simplicity, in the following we only consider the case where $\beta_k = \beta$ for some fixed choice of $\beta \in (0, 1)$.

6.2.1 Nesterov accelerated gradient descent (NAGD) and the limit of an ODE

In [90], a second-order ODE was derived which is the exact limit of Nesterov's scheme (6.1) as the step size s approaches 0 for the momentum-parameter sequence $\beta_k = \frac{k-1}{k+2}$:

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{u}(t) + \frac{3}{t}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) + \nabla f(\mathbf{u}(t)) = 0, \tag{6.8}$$

for $t > 0$ with initial conditions $\mathbf{u}(0) = \mathbf{u}_0$ and $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(0) = 0$, where $\mathbf{u}_0$ is the initial guess. The time parameter in this ODE is related to the step size s in Nesterov's scheme via $t \approx k\sqrt{s}$.

Now, we seek to derive a similar ODE limit for (6.1) with fixed β following the steps of derivation for (6.8) in [90]. Eliminating $\mathbf{v}_k$ using $\mathbf{v}_k = \mathbf{u}_k + \beta\left(\mathbf{u}_k - \mathbf{u}_{k-1}\right)$, we write (6.1) as a two-step update

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \beta\left(\mathbf{u}_k - \mathbf{u}_{k-1}\right) - s\nabla f\left(\mathbf{u}_k + \beta\left(\mathbf{u}_k - \mathbf{u}_{k-1}\right)\right). \tag{6.9}$$

We set the momentum parameter close to 1 via

$$\beta = 1 - \gamma h, \tag{6.10}$$

where $h = \sqrt{s}$ represents the discrete time-step scaling, and take $s \to 0$, implying $h \to 0$. Letting $t_k := kh$, we approximate the continuous trajectory by $\mathbf{u}_k \approx \mathbf{u}\left(t_k\right)$. Defining the discrete velocity as

$$\mathbf{z}_k := \frac{\mathbf{u}_k - \mathbf{u}_{k-1}}{h} \approx \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t_k), \tag{6.11}$$

from (6.9), we have

$$\mathbf{u}_{k+1} - \mathbf{u}_k = \beta\left(\mathbf{u}_k - \mathbf{u}_{k-1}\right) - s\nabla f\left(\mathbf{v}_k\right), \tag{6.12}$$

which we divide by h to obtain

$$\frac{\mathbf{u}_{k+1} - \mathbf{u}_k}{h} = \beta\frac{\mathbf{u}_k - \mathbf{u}_{k-1}}{h} - \frac{s}{h}\nabla f\left(\mathbf{v}_k\right). \tag{6.13}$$

Noting that $\frac{s}{h} = \frac{h^2}{h} = h$, this is equivalent to

$$\mathbf{z}_{k+1} = \beta\mathbf{z}_k - h\nabla f\left(\mathbf{v}_k\right). \tag{6.14}$$

Rearranging and substituting $\beta = 1 - \gamma h$, we obtain

$$\begin{aligned} \mathbf{z}_{k+1} - \mathbf{z}_k &= (\beta - 1)\mathbf{z}_k - h\nabla f\left(\mathbf{v}_k\right) \\ &= -\gamma h\mathbf{z}_k - h\nabla f\left(\mathbf{v}_k\right); \end{aligned} \tag{6.15}$$

dividing by h then yields

$$\frac{\mathbf{z}_{k+1} - \mathbf{z}_k}{h} = -\gamma \mathbf{z}_k - \nabla f\left(\mathbf{v}_k\right). \tag{6.16}$$

By (6.11), we also have that

$$\begin{aligned} \mathbf{v}_k &= \mathbf{u}_k + \beta\left(\mathbf{u}_k - \mathbf{u}_{k-1}\right) \\ &= \mathbf{u}_k + h\beta \mathbf{z}_k \\ &= \mathbf{u}_k + \mathcal{O}(h), \end{aligned} \tag{6.17}$$

which implies by continuity that

$$\nabla f\left(\mathbf{v}_k\right) = \nabla f\left(\mathbf{u}_k\right) + \mathcal{O}(h). \tag{6.18}$$

Taking the limit as $h \to 0$, and consequently $s \to 0$, we get

$$\frac{\mathbf{z}_{k+1} - \mathbf{z}_k}{h} \to \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{z}(t), \quad \mathbf{z}_k \to \mathbf{z}(t), \quad \mathbf{u}_k \to \mathbf{u}(t), \quad \nabla f\left(\mathbf{v}_k\right) \to \nabla f(\mathbf{u}(t)). \tag{6.19}$$

Putting it all together, we obtain the first-order system

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = \mathbf{z}(t), \quad \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{z}(t) = -\gamma \mathbf{z}(t) - \nabla f(\mathbf{u}(t)), \tag{6.20}$$

which we combine into the second-order ODE limit for (6.1) with fixed β:

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{u}(t) + \gamma \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) + \nabla f(\mathbf{u}(t)) = 0. \tag{6.21}$$

Remark 20 (ODE limit of the HB method)**.** We note that the continuous-time limit ODE (6.21) is identical to the ODE limit of Polyak's HB method (see remark 19). By eliminating $\mathbf{v}_k$ and writing (6.3) as a two-step update, we get

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \beta\left(\mathbf{u}_k - \mathbf{u}_{k-1}\right) - s\nabla f\left(\mathbf{u}_k\right). \tag{6.22}$$

Thus, the only structural difference between this and the eliminated Nesterov scheme in (6.9) is the point at which the gradient is evaluated; while HB evaluates $\nabla f(\mathbf{u}_k)$, NAGD in (6.9) evaluates $\nabla f(\mathbf{v}_k) \coloneqq \nabla f(\mathbf{u}_k + \beta(\mathbf{u}_k - \mathbf{u}_{k-1}))$. By the derivation above, $\mathbf{v}_k = \mathbf{u}_k + \mathcal{O}(h)$, meaning the Nesterov gradient evaluation satisfies $\nabla f\left(\mathbf{v}_k\right) = \nabla f\left(\mathbf{u}_k\right) + \mathcal{O}(\sqrt{s})$. As $s \to 0$, we have that $\nabla f\left(\mathbf{v}_k\right) \to \nabla f\left(\mathbf{u}_k\right)$, such that the distinction between the HB and NAGD algorithms for a sufficiently small step size, in the limit of continuous time. Hence, both converge to (6.21) for the same fixed β.

6.2.2 Nesterov-accelerated proximal point (NAPP) and the limit of an ODE

Recall from section 5.4 that the GD and PP methods can both be interpreted as forward- and backward-Euler discretizations, respectively, of the same ODE, (5.36). Similarly, we would like to establish a connection between the implicit NAPP method with fixed β and the ODE limit (6.21) of NAGD, (6.1), with fixed β. Hence, we consider the fixed-β NAPP iteration described by (6.5) with $\beta_k = \beta$ for all k. Recall (3.34) and note that since

$$\mathbf{u}_{k+1} = P_{sf}\left(\mathbf{v}_k\right) := \arg\min_{\mathbf{u}} \left\{ f(\mathbf{u}) + \frac{1}{2s}\left\|\mathbf{u} - \mathbf{v}_k\right\|^2 \right\}, \tag{6.23}$$

the optimality condition of the minimization problem contained in (6.23) is given by

$$\begin{aligned} 0 = \nabla f\left(\mathbf{u}_{k+1}\right) + \frac{1}{s}\left(\mathbf{u}_{k+1} - \mathbf{v}_k\right) &\iff \mathbf{u}_{k+1} = \mathbf{v}_k - s\nabla f\left(\mathbf{u}_{k+1}\right) \\ &\iff \frac{\mathbf{v}_k - \mathbf{u}_{k+1}}{s} = \nabla f(\mathbf{u}_{k+1}). \end{aligned} \tag{6.24}$$

Using the discrete time-step scaling $h = \sqrt{s}$ and the discrete velocity $\mathbf{z}_k := \frac{\mathbf{u}_k - \mathbf{u}_{k-1}}{h}$ from (6.11), and noting that

$$\begin{aligned} \mathbf{v}_k - \mathbf{u}_{k+1} &= \left(\mathbf{u}_k - \mathbf{u}_{k+1}\right) + \beta\left(\mathbf{u}_k - \mathbf{u}_{k-1}\right) \\ &= -h\mathbf{z}_{k+1} + \beta h \mathbf{z}_k, \end{aligned} \tag{6.25}$$

we have that

$$\frac{\mathbf{v}_k - \mathbf{u}_{k+1}}{s} = \frac{h(\beta\mathbf{z}_k - \mathbf{z}_{k+1})}{h^2} = \frac{\beta\mathbf{z}_k - \mathbf{z}_{k+1}}{h}. \tag{6.26}$$

Therefore, (6.24) is equivalent to

$$\frac{\mathbf{z}_{k+1} - \beta\mathbf{z}_k}{h} = -\nabla f\left(\mathbf{u}_{k+1}\right). \tag{6.27}$$

Substituting $\beta = 1 - \gamma h$, we find that

$$\frac{\mathbf{z}_{k+1} - \beta\mathbf{z}_k}{h} = \frac{\mathbf{z}_{k+1} - \mathbf{z}_k}{h} + \gamma\mathbf{z}_k. \tag{6.28}$$

Then, taking the limit as $h \to 0$, implying $s \to 0$, yields

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{z}(t) + \gamma\mathbf{z}(t) = -\nabla f(\mathbf{u}(t)), \quad \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = \mathbf{z}(t), \tag{6.29}$$

which once again yields the ODE limit in (6.21):

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{u}(t) + \gamma\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) + \nabla f(\mathbf{u}(t)) = 0. \tag{6.30}$$

Hence, both the fixed-β NAPP and NAGD methods converge to the same ODE.

6.2.3 FISTA and NAAP as ODEs

In [4], the ODE formulation of a general family of momentum-accelerated methods related to NAGD was derived as an extension of the NAGD ODE limit from [90]. In particular, the FISTA method (6.6) with $\beta_k = \frac{k-1}{k+2}$ was shown to be a forward-backward Euler discretization of the dynamical system

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{u}(t) + \frac{3}{t}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) + \partial g(x(t)) + \nabla f(\mathbf{u}(t)) \ni 0, \tag{6.31}$$

where the convex L-smooth f and the nondifferentiable convex g are defined as in (5.2). Similarly, in this context we can interpret NAAP as a backward-backward Euler discretization of (6.31). While here we do not derive a similar limit for FISTA and NAAP with fixed β parameters, due to the similarities between (6.31) and (6.8) and given our finding of the (6.21) limit for fixed-β NAGD and NAPP, we expect to find a dynamical system limit similar to (6.31), albeit with a constant in place of the $3/t$ term multiplying the first time derivative.

6.3 Momentum-accelerated parallel-in-iteration optimization using MGRIT

As a natural extension of our P-in-I framework from chapter 5, we consider solving Model Problem 1 (MP1), (5.15) with NAGD and NAPP replacing GD and PP, respectively. We consider all momentum-accelerated algorithms with fixed β parameters, which simplifies the work of finding a suitable implicit coarse-grid approximation for the explicit fine-grid method. Similarly, in principle we also consider FISTA and NAAP for Model Problem 2 (MP2), although we do not explore this approximation in detail here.

6.3.1 Choice of Ψ for MP1 using NAGD

Recall MP1, (5.15). For simplicity, we assume that $\mathbf{b} = \mathbf{0}$; this does not change the choice of operators, but allows us to omit the $\mathbf{b}$ terms in the gradient of the quadratic function. Note that the NAGD iteration (6.1) with fixed β for MP1 is given by

$$\begin{aligned} \mathbf{u}_{k+1} &= \mathbf{v}_k - sA_1\mathbf{v}_k, \\ \mathbf{v}_{k+1} &= \mathbf{u}_{k+1} + \beta\left(\mathbf{u}_{k+1} - \mathbf{u}_k\right). \end{aligned} \tag{6.32}$$

The above system can be written as the following matrix-vector product:

$$\begin{bmatrix} \mathbf{u}_{k+1} \\ \mathbf{v}_{k+1} \end{bmatrix} = \begin{bmatrix} 0 & I - sA_1 \\ -\beta I & (1+\beta)(I - sA_1) \end{bmatrix} \begin{bmatrix} \mathbf{u}_k \\ \mathbf{v}_k \end{bmatrix}. \tag{6.33}$$

Hence, we let

$$\Phi = \begin{bmatrix} 0 & I - sA_1 \\ -\beta I & (1+\beta)(I - sA_1) \end{bmatrix} \tag{6.34}$$

be our fine-grid operator.

Similarly, the NAPP iteration (6.5) with fixed β for MP1 is given by

$$\begin{aligned} \mathbf{u}_{k+1} &= \mathbf{v}_k - sA_1\mathbf{u}_{k+1}, \\ \mathbf{v}_{k+1} &= \mathbf{u}_{k+1} + \beta\left(\mathbf{u}_{k+1} - \mathbf{u}_k\right), \end{aligned} \tag{6.35}$$

or, equivalently,

$$\begin{aligned} (I + sA_1)\mathbf{u}_{k+1} &= \mathbf{v}_k, \\ \mathbf{v}_{k+1} &= \mathbf{u}_{k+1} + \beta\left(\mathbf{u}_{k+1} - \mathbf{u}_k\right). \end{aligned} \tag{6.36}$$

Hence, the NAPP is given in matrix-vector product form as

$$\begin{bmatrix} \mathbf{u}_{k+1} \\ \mathbf{v}_{k+1} \end{bmatrix} = \begin{bmatrix} 0 & (I + sA_1)^{-1} \\ -\beta I & (1+\beta)(I + sA_1)^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{u}_k \\ \mathbf{v}_k \end{bmatrix}. \tag{6.37}$$

Recall that our intention is to approximate m NAGD steps with one NAPP step. In order to maintain consistency with the approximation of m GD steps by one PP step, as discussed in subsection 5.5.1, we assume that a coarse-grid operator Ψ based on NAPP will have $(I + msA_1)^{-1}$ terms in place of the $I - sA_1$ terms in (6.34), such that when $\beta = 0$ we recover the approximation from (5.58). However, which fixed value β to use on the coarse grid is not immediately clear. In the following, we derive one candidate for Ψ based on NAPP. Although we only investigate this candidate for coarse-grid operator, other implicit methods related to NAGD can be considered. For example, with remark 20 in mind, we may consider deriving implicit variants of the HB method for obtaining candidates for the role coarse-grid time-stepping operator.

Deriving the $\Psi \approx \Phi^m$ approximation for NAGD and NAPP

For simplicity, suppose $A_1 = \lambda \in \mathbb{R}$; in particular, we may assume λ to be an eigenvalue of A_1, as given by (2.6). Letting $\boldsymbol{z}_k = \begin{bmatrix} \mathbf{u}_k & \mathbf{v}_k \end{bmatrix}^T$ and denoting

$$T = \begin{bmatrix} 0 & 1 - s\lambda \\ -\beta & (1+\beta)(1 - s\lambda) \end{bmatrix}, \tag{6.38}$$

applying m NAGD steps is described by

$$\boldsymbol{z}_{k+m} = T^m \boldsymbol{z}_k \tag{6.39}$$

for some $\mathbf{z}_k$, $k = 0, 1, \ldots$. Let

$$P = \begin{bmatrix} 0 & (1 + s_c\lambda)^{-1} \\ -\beta_c & (1 + \beta_c)(1 + s_c\lambda)^{-1} \end{bmatrix} \tag{6.40}$$

be the NAPP matrix, where β_c and s_c are yet to be determined coarse-grid momentum-parameter and step-size. We want to find s_c and β_c such that

$$P \approx T^m; \tag{6.41}$$

note that this is an analogue of the desired $\Psi \approx \Phi^m$ approximation. Our approach to optimizing the s_c and β_c parameters for the $P \approx T^m$ approximation is to approximate the determinant of these matrices; that is, we want to find P such that

$$\det(P) \approx \det(T^m) = \det(T)^m. \tag{6.42}$$

Computing the determinant of T yields

$$\det(T) = \beta(1 - s\lambda), \tag{6.43}$$

such that

$$\det(T^m) = \det(T)^m = \beta^m(1 - s\lambda)^m. \tag{6.44}$$

For the determinant of P, we find

$$\det(P) = \beta_c(1 + s_c\lambda)^{-1}. \tag{6.45}$$

Therefore, we want s_c and β_c to satisfy the approximation

$$\beta_c(1 + s_c\lambda)^{-1} \approx \beta^m(1 - s\lambda)^m. \tag{6.46}$$

For a sufficiently small s, we have that

$$(1 - s\lambda)^m \approx (1 + ms\lambda)^{-1}, \tag{6.47}$$

which is the approximation discussed in (5.58). Hence, we take $s_c = ms$, from which it follows that $\beta_c = \beta^m$. Extrapolating this analysis for a scalar λ to a matrix A_1, we then take

$$\Psi = \begin{bmatrix} 0 & (I + msA_1)^{-1} \\ -\beta^m I & (1 + \beta^m)(I + msA_1)^{-1} \end{bmatrix} \tag{6.48}$$

to be our coarse-grid operator for approximating m steps of (6.34).

Remark 21 (On the diagonalizability of Φ and Ψ)**.** Recall that two matrices are simultaneously diagonalizable if and only if they commute. Then we compute $\Phi\Psi$ and $\Psi\Phi$ products to verify the commutatitivity of Φ and Ψ. To simplify the notation, let $P = (I + msA_1)^{-1}$ and $G = (I - sA_1)$; we proved that P and G commute in corollary 2. Then, we have that

$$\Psi\Phi = \begin{bmatrix} -\beta P & (1+\beta)PG \\ -\beta(1+\beta^m)P & -\beta^m G + (1+\beta^m)(1+\beta)PG \end{bmatrix}, \tag{6.49}$$

$$\Phi\Psi = \begin{bmatrix} -\beta^m G & (1+\beta^m)GP \\ -\beta(1+\beta)G & -\beta P + (1+\beta)(1+\beta^m)GP \end{bmatrix}. \tag{6.50}$$

For Φ and Ψ to commute, we require $\Phi\Psi = \Psi\Phi$, which is not true unless $\beta = 0$, in which case we recover the GD and PP operators. Hence, we unfortunately cannot analyze this approximation using fact 43, for example.

However, Φ and Ψ are simultaneously block-diagonalizable. Recall from proposition 1 that A_1, (2.5), is diagonalizable by an orthogonal transformation Q, i.e., $A_1 = Q\Lambda Q^{-1}$, where Λ is the diagonal matrix of eigenvalues of A_1 and the columns of Q form an orthonormal basis of eigenvectors of A_1, which can be obtained from (2.7) using the Gram-Schmidt orthonormalization procedure. Hence, we can define the block-diagonal matrix

$$\mathcal{Q} = \begin{bmatrix} Q & \mathbf{0} \\ \mathbf{0} & Q \end{bmatrix} = I \otimes Q, \tag{6.51}$$

where $I \in \mathbb{R}^{2\times 2}$ is the identity matrix. Applying $\mathcal{Q}$ block-diagonalizes Φ and Ψ into a set of 2×2 matrices for each eigenvalue λ_i of A_1:

$$\mathcal{Q}^{-1}\Phi\mathcal{Q} \Longrightarrow \begin{bmatrix} 0 & (1 - s\lambda_i) \\ -\beta & (1+\beta)\,(1 - s\lambda_i) \end{bmatrix}, \tag{6.52}$$

$$\mathcal{Q}^{-1}\Psi\mathcal{Q} \Longrightarrow \begin{bmatrix} 0 & (1 + ms\lambda_i)^{-1} \\ -\beta^m & (1+\beta^m)\,(1 + ms\lambda_i)^{-1} \end{bmatrix}, \tag{6.53}$$

which recover the T (6.38) and P (6.40) matrices, respectively, for $\lambda = \lambda_i$. Hence, our scalar analysis of $P \approx T^m$ for finding $\Psi \approx \Phi^m$ is justified even in the absence of simultaneous diagonalizability.

6.4 Issues with P-in-I solves of momentum-accelerated algorithms

Initially, we expected that Multigrid Reduction-in-Time (MGRIT) in standard form with implicit discretization on the coarse grid, as explained above, would work well for MP1,

(5.15), with NAGD operator (6.8) as the fine-grid operator and NAPP operator (3.75) as the coarse-grid operator, given the positive results obtained with GD, (5.30). However, in our preliminary work, MGRIT has failed to satisfactorily converge except for when using very small values of $\beta \approx 0$, in which case NAGD behaves almost exactly like GD. Due to this currently unsolved issue, we do not provide numerical results for FISTA applied to solving MP2, (5.23), as originally intended.

6.4.1 Numerical MGRIT results for NAGD applied to MP1

For the following numerical experiments, we consider the setting of subsection 5.7.1, with $N = n = 40$, $m = 4$ and $\ell = 2$, except now we consider the fixed-β NAGD method (6.1) for solving MP1, (5.15). We note that the optimal fixed-momentum parameter from (6.2) is given by $\beta^* = 0.9262$, and that NAGD required only $N_t = 392$ iterations to reduce the gradient by a factor of $\varepsilon = 10^{-8}$; recall from subsection 5.7.1 that the GD method (5.30) (i.e., NAGD with $\beta = 0$) required $8,503$ iterations to solve the same problem with the same accuracy threshold. In figure 6.1, we see that MGRIT, with fine-grid operator Φ (6.34) and coarse-grid operator Ψ (6.48), required 42 iterations to converge. However, while the residual (4.4) was reduced by a factor of $\varepsilon = 10^{-8}$, the gradient at the final time point is only reduced by a factor of roughly 10^{-3}.

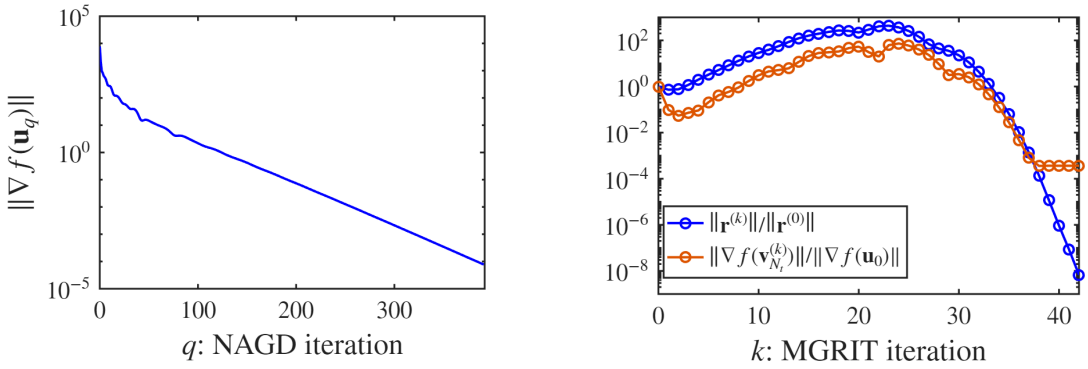
Alternatively, in figure 6.2 we consider taking a smaller fixed $\beta = 0.1$, for which NAGD requires $N_t = 7,652$ iterations to converge. While MGRIT now requires only 10 iterations to reduce the initial residual by a factor of $\epsilon = 10^{-8}$, the gradient at the final time point is only reduced by a factor close to 10^{-4}. Therefore, even for small β, although MGRIT converges in the residual, the gradient reduction is inferior to that of GD in subsection 5.7.1.

Unlike GD, which we have observed to behave similarly to a parabolic Partial Differential Equation (PDE) in subsection 5.4.2, the same does not apply for NAGD as explained in the following remark.

Remark 22 (Nesterov's accelerated gradient ODE and hyperbolic PDE connection)**.** Consider Nesterov's accelerated gradient ODE (6.8) applied to MP1, (5.15), such that $\nabla f(\mathbf{u}(t)) = A_1\mathbf{u}$, where A_1 is the 1D Laplacian matrix in (2.5) with $(A_1\mathbf{u})_i \approx -\frac{\partial^2}{\partial x^2}u(x_i, t)$. Then, we have

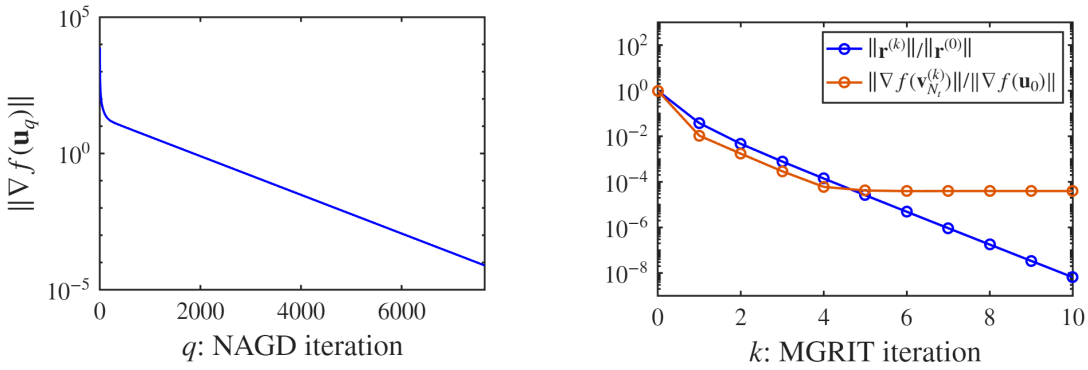
$$\frac{\partial^2}{\partial t^2}u(x,t) + \frac{3}{t}\frac{\partial}{\partial t}u(x,t) - \frac{\partial^2}{\partial x^2}u(x,t) = 0. \tag{6.54}$$

which is a hyperbolic PDE that approaches the wave equation (see (2.142)) as $t \to +\infty$.



(a) Sequential convergence **(b)** MGRIT convergence

Figure 6.1: **(a)**: Convergence of MP1, (5.15), with a standard sequential solve using NAGD method (6.1) with fixed $\beta = 0.9262$; $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. **(b)**: Convergence of MP1, (5.15), with MGRIT using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\mathbf{r}^{(k)}$ denotes residual (4.4) at iteration k, $\nabla f(\mathbf{v}_{N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point N_t and MGRIT iteration k, $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual.



(a) Sequential convergence **(b)** MGRIT convergence

Figure 6.2: **(a)**: Convergence of MP1, (5.15), with a standard sequential solve using NAGD method (6.1) with fixed $\beta = 0.1$; $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. **(b)**: Convergence of MP1, (5.15), with MGRIT using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\mathbf{r}^{(k)}$ denotes residual (4.4) at iteration k, $\nabla f(\mathbf{v}_{N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point N_t and MGRIT iteration k, $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^{(0)}$ denotes the initial MGRIT residual.

This behaviour similar to a hyperbolic PDE, which MGRIT in standard form is known

to struggle with, might be a reason for why Nesterov's accelerated gradient did not work with MGRIT using the standard algorithmic approach operators described in section 5.5. On the other hand, it has recently been shown that MGRIT can be made to work for hyperbolic PDEs by carefully considering non-standard specialized coarse operators: research on on modified semi-Lagrangian coarse-grid operators for advection-dominated and hyperbolic PDEs [23, 21, 19, 22, 58] has shown progress in adapting MGRIT to solve hyperbolic PDEs, which indicates a promising direction for further research on applying MGRIT to momentum-accelerated optimization algorithms.

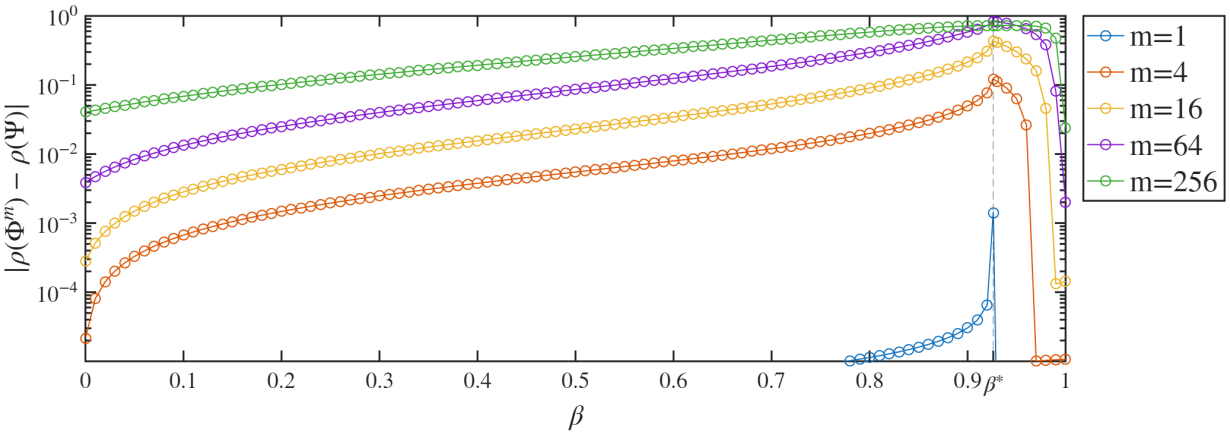
Remark 23 (Other parallel-in-time algorithms for hyperbolic PDEs)**.** As previously mentioned, MGRIT tends to work better for parabolic PDE problems than it does for hyperbolic problems. However, many other parallel-in-time (P-in-T) methods exist, some which are more suitable for hyperbolic problems. For example, [42] names methods such as the Schwarz Waveform Relaxation (SWR) family of methods [33, 36, 37], the parallel Integral Deferred Correction (IDC) family of methods [17, 49], and the ParaExp method [35] as methods which have had success in certain hyperbolic settings. We leave applying our P-in-I framework with different P-in-T methods, such as the ones mentioned in this remark, in consideration for future work.

6.4.2 Failure of the momentum-accelerated MGRIT operators

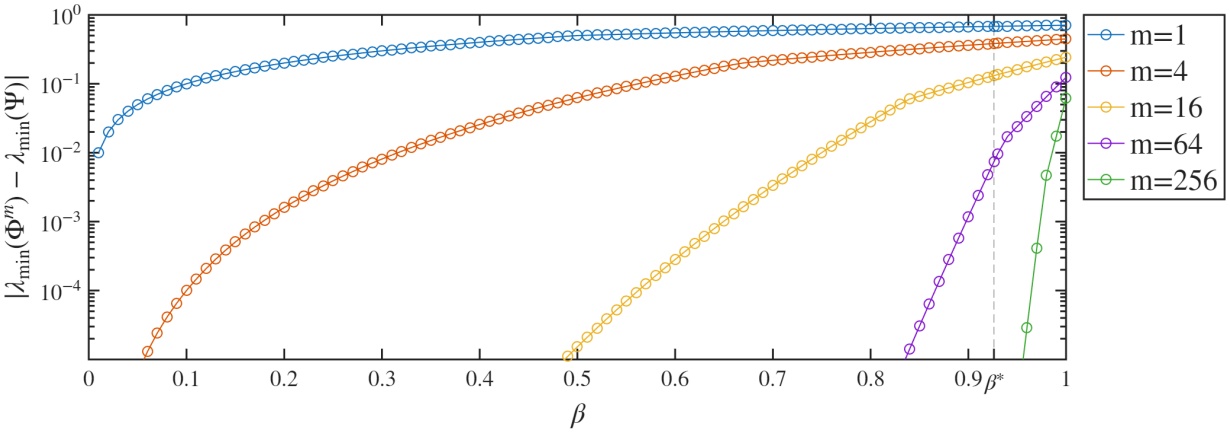
Now, we discuss the failure of the NAPP coarse-grid operator Ψ (6.48) as an approximation of m steps of the NAGD fine-grid operator Φ (6.34). To illustrate how the error of the $\Psi \approx \Phi^m$ approximation varies as a function of the momentum-parameter β and the coarsening factor m, in figure 6.3 we measure the absolute difference between the smallest and largest eigenvalues of Ψ and Φ^m. From figure 6.3, we see that the error of the approximation for both smallest and largest eigenvalues is relatively small for $\beta = 0$ and grows as $\beta \to 1$. In particular, the absolute spectral radius difference is maximized precisely at the optimal β^*.

Furthemore, following the analysis from [74, Section 4] for NAGD applied to minimizing a quadratic function, we can show that NAGD also fails as a relaxation scheme, as discussed in remark 24.

Remark 24 (NAGD fails as a relaxation scheme)**.** We recall from algorithm 6 that the FCF-relaxation (see subsection 4.1.1) is performed using Φ, the fine-grid operator. Then, consider using the fine-grid NAGD operator Φ, (6.34), for relaxation. After a block-diagonalization of Φ, as in (6.52), the error smoothing of one relaxation step associated



(a) Absolute difference between the spectral radii (largest eigenvalues) of Φ^m and Ψ (6.48)



(b) Absolute difference between the smallest eigenvalues of Φ^m and Ψ

Figure 6.3: Error of the $\Psi \approx \Phi^m$ approximation for the fixed-β NAGD (6.1) fine-grid operator Φ (6.34), and the fixed-β NAPP (6.5) coarse-grid operator Ψ (6.48), for varying levels of β, as measured in terms of the absolute difference of their smallest (a) and largest (b) eigenvalues. Here, $\beta^* = 0.9262$ is the optimal fixed β (6.2).

with λ_n is given by

$$\mathbf{e}_n^{(i+1)} = \begin{bmatrix} e_{u,n}^{(i+1)} \\ e_{v,n}^{(i+1)} \end{bmatrix} = \begin{bmatrix} 0 & 1 - z_n \\ -\beta & (1+\beta)(1-z_n) \end{bmatrix} \begin{bmatrix} e_{u,n}^{(i)} \\ e_{v,n}^{(i)} \end{bmatrix} = \Phi_n \mathbf{e}_n^{(i)}, \tag{6.55}$$

where $e_{u,n}^{(i)}$ and $e_{v,n}^{(i)}$ denote the error corresponding to the nth mode of $\mathbf{u}$ and $\mathbf{v}$, respectively,

at time point i. The matrix-vector product above yields the system

$$e_{u,n}^{(i+1)} = (1 - z_n)e_{v,n}^{(i)}, \tag{6.56}$$

$$e_{v,n}^{(i+1)} = -\beta e_{u,n}^{(i)} + (1 + \beta)(1 - z_n)e_{v,n}^{(i)}. \tag{6.57}$$

By isolating $e_{v,n}^{(i)}$ in (6.56) and substituting it into a shifted version of (6.57), we obtain a linear recurrence relation for the scalar error $e_{u,n}$ given by

$$e_{u,n}^{(i+2)} - (1 + \beta)(1 - z_n)e_{u,n}^{(i+1)} + \beta(1 - z_n)e_{u,n}^{(i)} = 0, \tag{6.58}$$

whose coefficients match the characteristic equation of Φ_n, given by

$$r^2 - (1 + \beta)(1 - z_n)r + \beta(1 - z_n) = 0 \tag{6.59}$$

for an eigenvalue r of Φ_n. By [74], this characteristic equation has repeated roots only if $\beta = \beta_n^*$, where

$$\beta_n^* = \frac{1 - \sqrt{\lambda_n/L}}{1 + \sqrt{\lambda_n/L}}, \quad n = 1, \ldots, N, \tag{6.60}$$

and has complex roots when $\beta > \beta_n^*$, in which case the nth mode NAGD trajectory is periodic. Then, recall that $\mu \coloneqq \min_n \lambda_n$ and note by (6.2) that

$$\beta_n^* = \frac{1 - \sqrt{\lambda_n/L}}{1 + \sqrt{\lambda_n/L}} \leq \frac{1 - \sqrt{\mu/L}}{1 + \sqrt{\mu/L}} \coloneqq \beta^* \tag{6.61}$$

holds for all n, with equality only for $\lambda_n = \mu$; i.e., $z_n = 1/\kappa \approx 0$. Hence, for the error to be smoothed without any oscillations, we must have

$$\beta \leq \min_n \beta_n^* = 0, \tag{6.62}$$

where the minimum is attained by $\lambda_n = L$; i.e., $z_n = 1$. Hence the NAGD fine-grid operator Φ, defined in (6.34), only smooths the error without introducing oscillations to any mode for $\beta = 0$, in which case we recover the GD (5.30) operator.

To avoid the non-monotonic, periodic ripples in the NAGD function-value trajectories that occur when taking $\beta > \beta^*$, restarting schemes were developed in [74] for handling this problem in a sequential setting; we briefly comment on them in remark 25. However, by remark 24, for any $\beta > 0$, NAGD as a relaxation method introduces oscillations to the error that are captured by the residual and aliased on the coarse-grid, leading MGRIT to perform inaccurate coarse-grid corrections. Together with the observation that the coarse-grid approximation worsens as β increases, as shown in figure 6.3, we infer that it might be necessary to work only with small $\beta < \beta^*$ in order to ensure the stability of both the relaxation and coarse-grid correction processes.

Remark 25 (Restarting NAGD schemes)**.** For momentum-accelerated optimization algorithms [74, 57, 5, 45, 61] and parameter-dependent algorithms in general [79, 88, 86], several restarting strategies have been recently developed. In particular, for NAGD, the typical sequences for generating the β_k parameters in (6.1) start from $\beta_1 = 0$ and, as $k \to +\infty$, $\beta_k \to 1$. However, as the momentum-parameter grows, after some threshold related to the (presumably unknown) condition number κ of the objective function is passed, the NAGD trajectory is no longer monotone in the function value and starts featuring periodic ripples. Then, in [74] two strategies were presented for restarting the β_k sequence (i.e., returning to $\beta = 0$ for the next iteration), the first when the function value increases,

$$f(\mathbf{u}_k) > f(\mathbf{u}_{k-1}), \tag{6.63}$$

and the second when the gradient satisfies

$$\langle \mathbf{u}_k - \mathbf{u}_{k-1}, \nabla f(\mathbf{u}_{k-1}) \rangle. \tag{6.64}$$

Using these restarting schemes, the β_k parameters are effectively bounded from above. Unfortunately, adaptive strategies such as the restarting NAGD schemes above do not lend themselves naturally to an MGRIT implementation; we discuss this more generally in chapter 8.

6.4.3 Parallel-in-iteration GD vs. sequential NAGD

We recall from fact 41 that GD converges to an ϵ-optimal solution (i.e., $f(\mathbf{u}_k) - f^* \leq \epsilon$, where $f^* := \min f$) in $\mathcal{O}(L/\epsilon)$ and $\mathcal{O}(\kappa \log(1/\epsilon))$ iteration for L-smooth convex and μ-strongly convex problems, respectively. In subsection 6.1.1, we state that NAGD improves these convex and strongly convex complexity rates to $\mathcal{O}(\sqrt{L/\epsilon})$ and $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$. Therefore, for (strongly) convex smooth problems, we expect NAGD to converge in substantially fewer iterations than GD. In particular, even a P-in-I GD solve is likely to require a larger wall-clock time than a sequential NAGD solve. Therefore, until the issues previously discussed in this section are addressed, we do not expect our P-in-I framework, in its current stage of development, to be practically useful for problems such as MP1, (5.15). However, for nonconvex optimization problems, for which the advantage of NAGD over GD is possibly much smaller, this might not necessarily hold true, as per the following remark.

Remark 26 (GD vs. NAGD for nonconvex optimization)**.** For nonconvex smooth optimization problems, GD approaches an ϵ-approximate first-order stationary point (i.e.,

$\|\nabla f(\mathbf{u}_k)\| \leq \epsilon$) in $\mathcal{O}(\epsilon^{-2})$ iterations [71], which is also the same worst-case bound for NAGD. Recent developments on nonconvex restarted NAGD schemes have improved this rate to $\mathcal{O}(\epsilon^{-7/4}\log(1/\epsilon))$ [14] and $\mathcal{O}(\epsilon^{-7/4})$ [61], which still imply a very large number of iterations may be required until convergence. In particular, a P-in-I GD solve of $\mathcal{O}(\epsilon^{-2})$ iterations may require less wall-clock time than a sequential $\mathcal{O}(\epsilon^{-7/4})$ solve. Hence, applying our P-in-I framework for nonconvex smooth optimization problems may be fruitful for speeding up problems typically solved with NAGD, even if GD is used as the fine-grid method.

6.5 Conclusions

In an attempt to parallelize over iterations of momentum-accelerated optimization methods, namely NAGD, we find in our numerical experiments that our parallel-in-iteration framework from [3] and chapter 5 fails to improve on the sequential application of such methods, with unsatisfactory performance in both the error smoothing and coarse-grid correction elements of the MGRIT method. We established that in order to extend our parallel-in-iteration framework to momentum-accelerated algorithms such as NAGD and FISTA, we will likely need to address MGRIT's difficulties with handling hyperbolic PDEs. Adapting MGRIT with this intent is an intricate problem-specific endeavour, which we leave as a suggestion for future work.

Chapter 7

Multi-trajectory parallel-in-iteration optimization

In this chapter, we continue discussing the parallel-in-iteration (P-in-I) framework introduced in section 5.5, considering the same two model problems from section 5.2 and optimization methods described in section 5.3. We recall from chapter 4 that Multigrid Reduction-in-Time (MGRIT) was originally intended for solving differential equation problems in which we want to approximate the entire sequential trajectory $\{\mathbf{u}_i\}_{i=0,1,\dots,N_t}$, which corresponds to the solutions of the system of equations (4.1) obtained by discretizing the Initial Boundary Value Problem (IBVP) of our interest. However, for optimization problems we are only interested in approximating the final point of the sequence, $\mathbf{u}_{N_t}$, which is by assumption an accurate approximation of the exact minimizer $\mathbf{u}^*$ (5.53) of the underlying optimization problem, e.g., the minimizer of a convex smooth function f (5.1) or of the sum $F \coloneqq f + g$ of convex but possibly nondifferentiable functions (5.2).

7.1 Limitations of MGRIT as a parallel-in-iteration solver

In the following discussion, we consider the more general setting of (5.2), for which the convergence of the optimization method (5.52) is measured in terms of the generalized gradient G_{sF}; recall that letting $g = 0$ recovers the setting of (5.1) with $G_{sF} = \nabla f$. In particular, in the next two remarks we highlight two structural features of the MGRIT

algorithm that suggest it is suboptimal in its default state if our goal is to accurately approximate only the tail of the sequential trajectory.

Remark 27 (MGRIT updates prioritize early time points)**.** Recalling algorithms 6 and 7, MGRIT computes the initial points exactly via FCF-relaxation; namely, after k iterations, the first $2mk$ points are computed exactly. At the same time, it propagates these exact early-time solutions to the later time points via FCF-relaxation combined with the coarse-grid correction. In this process, the solutions at the final time points are the last ones to be accurately approximated, as we observe in the numerical experiments from section 5.7. Thus, in practice, for $\mathbf{v}_{N_t}$, the MGRIT approximate solution at the final time point, to approximate the final point of the exact trajectory, $\mathbf{u}_{N_t}$, MGRIT needs the entire approximate trajectory $\{\mathbf{v}_i\}_{i=0,\dots,N_t}$ to approximate the exact trajectory $\{\mathbf{u}_i\}_{i=0,\dots,N_t}$ to the desired accuracy threshold.

Remark 28 (MGRIT's gradient reduction stalls in the final iterations)**.** As seen in figure 5.2, for example, gradient reduction obtained by MGRIT drastically slows down in later MGRIT iterations, even when attaining a linear convergence rate in the intermediary iterations. Recall from algorithms 6 and 7 that the MGRIT halting criteria is satisfied when the full MGRIT residual (4.4) is reduced to a user-defined small fraction ε of the initial residual. While the norm of the (generalized) gradient at the final time point may be very close to the desired accuracy after a small number of iterations, one or more additional iterations may be needed for MGRIT to converge in the residual. In these additional iterations, the gradient at the final time point is barely reduced at a relatively large wall-clock time cost; see section 5.8.

Remarks 27 and 28 suggest that there is room for improving our P-in-I framework beyond a standard MGRIT solve. Another issue discussed in remark 16 is that for MGRIT to provide a satisfactory final-point gradient reduction, we need to accurately estimate the exact number N_t of iterations of the underlying optimization method required to reduce the gradient by the desired amount, requiring N_t+1 fine-grid time points. Our recommendation from remark 16 for handling the case where we underestimate N_t leads to the first idea of approximating multiple MGRIT trajectories rather than just one, which we now discuss in more detail.

7.1.1 The multiple trajectories idea for MGRIT with an insufficient number of time points

Suppose we use MGRIT for solving an all-at-once system (4.3) with $\widetilde{N}_t+1$ fine-grid points, given by

$$\mathcal{A}(\mathbf{u}) = \begin{bmatrix} I & & & \\ -\Phi(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{\widetilde{N}_t} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{\widetilde{N}_t} \end{bmatrix} = \mathbf{w}, \tag{7.1}$$

with initial condition $\mathbf{u}_0 = \mathbf{w}_0$, which is equivalent to approximately computing $\widetilde{N}_t$ iterations of the underlying iterative optimization method (5.52), and that N_t iterations are required for sufficiently reducing the (generalized) gradient by a factor ε of the initial gradient using the fine-level optimization method. Suppose we let the MGRIT approximate trajectory converge to the exact solution of this initial all-at-once system (7.1). If the number of fine-grid points $\widetilde{N}_t$ is smaller than N_t, then the MGRIT approximate solution at the final time point $\mathbf{v}_{\widetilde{N}_t}$ will have a gradient

$$\|G_{sF}(\mathbf{v}_{\widetilde{N}_t})\| = \|G_{sF}(\mathbf{u}_{\widetilde{N}_t})\| > \|G_{sF}(\mathbf{u}_{N_t})\| = \varepsilon\|G_{sF}(\mathbf{u}_0)\|. \tag{7.2}$$

Then, exactly solving a new all-at-once system of $\widetilde{N}_t + 1$ time points described by

$$\mathcal{A}(\widetilde{\mathbf{u}}) = \begin{bmatrix} I & & & \\ -\Phi(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{u}}_0 \\ \widetilde{\mathbf{u}}_1 \\ \vdots \\ \widetilde{\mathbf{u}}_{\widetilde{N}_t} \end{bmatrix} = \begin{bmatrix} \widetilde{\mathbf{w}}_0 \\ \widetilde{\mathbf{w}}_1 \\ \vdots \\ \widetilde{\mathbf{w}}_{\widetilde{N}_t} \end{bmatrix} = \widetilde{\mathbf{w}}, \tag{7.3}$$

with initial condition $\widetilde{\mathbf{w}}_0 = \widetilde{\mathbf{u}}_0 \coloneqq \mathbf{v}_{\widetilde{N}_t} = \mathbf{u}_{\widetilde{N}_t}$, is equivalent to exactly solving the system

$$\widehat{\mathcal{A}}(\widehat{\mathbf{u}}) = \begin{bmatrix} I & & & & & & & \\ -\Phi(\cdot) & I & & & & & & \\ & \ddots & \ddots & & & & & \\ & & -\Phi(\cdot) & I & & & & \\ & & & -\Phi(\cdot) & I & & & \\ & & & & \ddots & \ddots & & \\ & & & & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{\widetilde{N}_t} \\ \mathbf{u}_{\widetilde{N}_t+1} \\ \vdots \\ \mathbf{u}_{2\widetilde{N}_t-1} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{\widetilde{N}_t} \\ \mathbf{w}_{\widetilde{N}_t+1} \\ \vdots \\ \mathbf{w}_{2\widetilde{N}_t-1} \end{bmatrix} = \widehat{\mathbf{w}}, \tag{7.4}$$

with the same initial condition $\mathbf{u}_0 = \mathbf{w}_0$ of (7.1). In particular, if $N_t \leq 2\widetilde{N}_t - 1$, by solving (7.4) exactly we have successfully reduced the gradient to the level of

$$\|G_{sF}(\mathbf{u}_{2\widetilde{N}_t-1})\| \leq \|G_{sF}(\mathbf{u}_{N_t})\| = \varepsilon \|G_{sF}(\mathbf{u}_0)\|. \tag{7.5}$$

However, we note that we do not have to go as far as approximating the second trajectory $\{\widetilde{\mathbf{u}}_i\}_{i=0,\ldots,\widetilde{N}_t}$ exactly; by approximately solving (7.3) with a number of MGRIT iterations, we eventually find some $\widetilde{\mathbf{v}}_i$, where $\widetilde{\mathbf{v}}_i \approx \widetilde{\mathbf{u}}_i$ for some $i = 0, 1, \ldots, \widetilde{N}_t$, such that $\|G_{sF}(\widetilde{\mathbf{v}}_i)\| \leq \|G_{sF}(\mathbf{u}_{N_t})\|$.

We illustrate this idea with a numerical example as follows. Recall the setting of subsection 5.7.1, in which we solve Model Problem 1 (MP1), (5.15), with the Gradient Descent (GD) method (5.30). Let the size of 1D Laplacian matrix, A_1 from (2.5), be $N = 80$. Recall from table 5.1(c) that for $N = 80$, GD requires $N_t = 30,299$ iterations to sufficiently reduce the initial gradient by a factor of $\varepsilon = 10^{-8}$. In figure 7.1, we see that computing $\widetilde{N}_t = 20,000$ GD iterations leads to a gradient reduction of approximately $10^{-6} > \varepsilon$; after another $\widetilde{N}_t$ iterations, the gradient is reduced to a factor smaller than $10^{-9} < \varepsilon$.

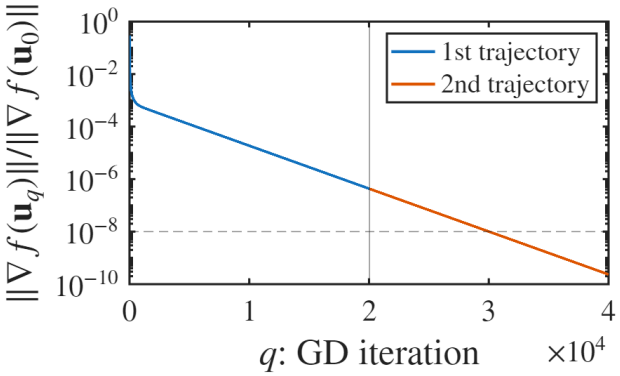


Figure 7.1: Convergence of MP1, (5.15), with a standard sequential solve using GD method (5.30); $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. Each trajectory, the first associated with (7.1) and the second with (7.3), corresponds to $\widetilde{N}_t = 20,000$ GD iterations, with the 10^{-8} gradient reduction threshold being crossed exactly at $q = 30,299$.

In figure 7.2, we consider solving the systems (7.1) and (7.3) with $\widetilde{N}_t = 20,000$ using MGRIT, with coarsening factor $m = 64$ and the maximum number of levels $\widehat{\ell} = 2$. We confirm that computing the first trajectory with high accuracy, i.e., letting MGRIT converge for solving (7.1), reduces the gradient by the expected factor of approximately 10^{-6} in 8 iterations. Then, after 6 MGRIT iterations on the (7.3) system, starting from $\widetilde{\mathbf{u}}_0 = \mathbf{v}_{N_t}^{(8)}$,

the gradient at the final time point is reduced to the 10^{-8} threshold, such that the second trajectory does not need to be computed with full accuracy. Note that combining the first and second high-accuracy trajectories, corresponding to $k = 8$ in figure 7.2 recovers the GD curve from figure 7.1, as expected.

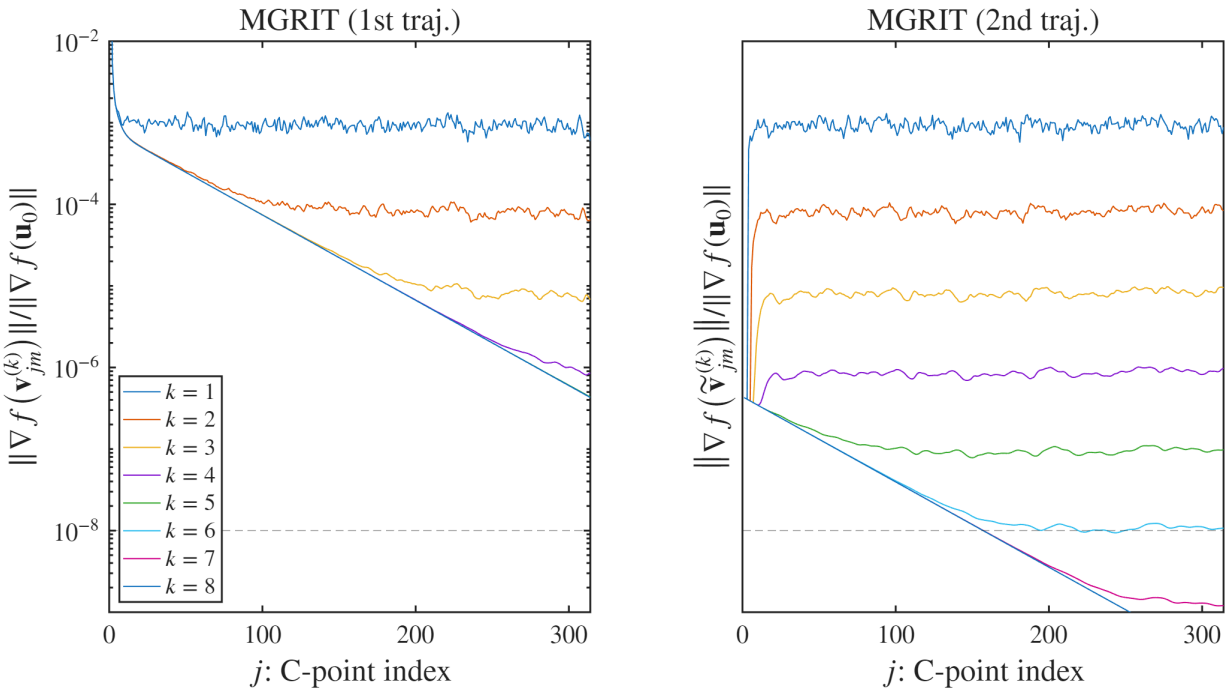


Figure 7.2: Left: 2-norm of $\nabla f\left(\mathbf{v}_{jm}^{(k)}\right)$, where $\mathbf{v}_{jm}^{(k)}$ is the approximate solution at coarse time point jm, $j = 0, \ldots, N_T$, and k is the MGRIT iteration index for the (7.1) system. **Right:** 2-norm of $\nabla f\left(\widetilde{\mathbf{v}}_{jm}^{(k)}\right)$, $\mathbf{v}_{jm}^{(k)}$ is the approximate solution at coarse time point jm, $j = 0, \ldots, N_T$, and k is the MGRIT iteration index for the (7.3) system. Here, $k = 6$ MGRIT iterations is sufficient to reach the desired 10^{-8} accuracy.

We extend this idea further by considering solving (7.1) approximately, then generating a sequence of new systems of the form (7.3) to also be solved inexactly. Our motivation is to investigate whether doing a small number (e.g., 1) of MGRIT iterations per trajectory may lead to a more efficient P-in-I method, since we do not waste time on computing entire trajectories accurately, but rather focus on computing the final point of the sequential trajectory, $\mathbf{u}_{N_t}$, accurately. We formally define our Multi-trajectory (MT) P-in-I strategy in section 7.2.

7.2 MGRIT for multi-trajectory parallel-in-iteration optimization

7.2.1 Defining the multiple trajectories

Consider applying MGRIT to an all-at-once system $\mathcal{A}(\mathbf{u}) = \mathbf{w}$ of the form (4.3) of $N_t + 1$ time points, where N_t is arbitrary, which, after a number of MGRIT iterations, yields approximate solutions

$$\{\mathbf{v}_i\}_{0 \leq i \leq N_t}, \quad \mathbf{v}_i \approx \mathbf{u}_i, \ 0 \leq i \leq N_t. \tag{7.6}$$

In particular, suppose that no approximate solution is a sufficiently accurate approximation of the exact minimizer $\mathbf{u}^*$ (5.53) of the underlying optimization problem, as measured by the generalized gradient of the $\mathbf{v}_i$ in (7.6). Then, we consider solving a new all-at-once system with initial guess $\widetilde{\mathbf{u}}_0 = \mathbf{v}_{N_t}$ (i.e., the MGRIT approximate solution at the final time point of the initial system) for an updated all-at-once system

$$\mathcal{A}(\widetilde{\mathbf{u}}) = \begin{bmatrix} I & & & \\ -\Phi(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{u}}_0 \\ \widetilde{\mathbf{u}}_1 \\ \vdots \\ \widetilde{\mathbf{u}}_{N_t} \end{bmatrix} = \begin{bmatrix} \widetilde{\mathbf{w}}_0 \\ \widetilde{\mathbf{w}}_1 \\ \vdots \\ \widetilde{\mathbf{w}}_{N_t} \end{bmatrix} = \widetilde{\mathbf{w}}, \tag{7.7}$$

where $\widetilde{\mathbf{w}}_0 = \widetilde{\mathbf{u}}_0$ is the new initial condition. After applying MGRIT to obtain an inexact solve of $\mathcal{A}(\widetilde{\mathbf{u}}) = \widetilde{\mathbf{w}}$ with a given number of MGRIT iterations, we pick $\widetilde{\mathbf{v}}_{N_t}$, the MGRIT approximate solution at the final time point of the updated system, to be the new initial guess, repeating this process until we obtain an approximate solution with a sufficiently small gradient. We note that, with each new all-at-once system of the form (7.7), we define the exact trajectory $\widetilde{\mathbf{u}}$ as the solution of the all-at-once system $\mathcal{A}(\widetilde{\mathbf{u}}) = \widetilde{\mathbf{w}}$, i.e., the trajectory generated by N_t steps of the same underlying optimization method (5.52) (e.g., GD (5.30) for MP1, (5.15)) starting from the updated initial guess $\widetilde{\mathbf{u}}_0$ on the fine grid. Since we are now discussing multiple "exact" trajectories $\widetilde{\mathbf{u}}$, none of which will be accurately approximated, we refer to them as *target trajectories*.

For example, consider an optimization method of the form (5.52), starting from some initial guess $\widetilde{\mathbf{u}}_0$, with $\widetilde{\mathbf{w}}_0 = \widetilde{\mathbf{u}}_0$ and $\widetilde{\mathbf{w}}_i = \mathbf{0}$ for all $1 \leq i \leq N_t$. Then, the target trajectory for an MGRIT all-at-once system solve (7.7) starting from $\widetilde{\mathbf{u}}_0$ is given by

$$\begin{aligned} \{\widetilde{\mathbf{u}}_i\}_{0 \leq i \leq N_t} &:= \{\widetilde{\mathbf{u}}_0, \ \widetilde{\mathbf{u}}_1, \ \widetilde{\mathbf{u}}_2, \ \ldots, \ \widetilde{\mathbf{u}}_{N_t}\} \\ &= \{\widetilde{\mathbf{u}}_0, \Phi(\widetilde{\mathbf{u}})_0, \ \Phi^2(\widetilde{\mathbf{u}}_0), \ldots, \Phi^{N_t}(\widetilde{\mathbf{u}}_0)\}, \end{aligned} \tag{7.8}$$

where $\Phi^k(\cdot)$ denotes k successive applications of the operator Φ, i.e., $\Phi^k(\cdot) = \underbrace{\Phi \circ \Phi \circ \cdots \circ \Phi}_{k \text{ times}}(\cdot)$.

Let $\widetilde{\mathbf{u}}_i^{(k)}$ denote the ith time point of the exact solution of the kth all-at-once system (7.7), and let $\widetilde{\mathbf{v}}_i^{(k)}$ be the MGRIT approximate solution at time point i on the kth all-at-once system after a number of MGRIT iterations; we may omit the superscript for $k = 1$, such that $\mathbf{u}_i = \mathbf{u}_i^{(1)}$ denotes the ith point of the target trajectory of the first all-at-once system, i.e., what we previously referred to as the exact trajectory of the (4.3) system. We then have that

$$\begin{aligned} \{\widetilde{\mathbf{u}}_i^{(k)}\}_{0 \leq i \leq N_t} &= \{\widetilde{\mathbf{u}}_0^{(k)}, \Phi(\widetilde{\mathbf{u}}_0^{(k)}),\ \Phi^2(\widetilde{\mathbf{u}}_0^{(k)}), \ldots, \Phi^{N_t}(\widetilde{\mathbf{u}}_0^{(k)})\} \\ &= \{\widetilde{\mathbf{v}}_{N_t}^{(k-1)}, \Phi(\widetilde{\mathbf{v}}_{N_t}^{(k-1)}),\ \Phi^2(\widetilde{\mathbf{v}}_{N_t}^{(k-1)}), \ldots, \Phi^{N_t}(\widetilde{\mathbf{v}}_{N_t}^{(k-1)})\}, \end{aligned} \tag{7.9}$$

Since $\widetilde{\mathbf{u}}_0^{(k)} \coloneqq \widetilde{\mathbf{v}}_{N_t}^{(k-1)}$ is obtained by performing at least one MGRIT iteration on the $(k-1)$th all-at-once system of the form (7.7), it is assumed to be closer to the true optimum $\mathbf{u}^*$ than the previous initial guess $\widetilde{\mathbf{u}}_0^{(k-1)}$.

Assume that $\mathbf{u}^*$ is the unique minimizer of (5.2), i.e., $\operatorname{Arg\,min} F = \{\mathbf{u}^*\}$. If the optimization method (5.52) from which $\Phi(\cdot)$ is obtained generates Féjer-monotone sequences (3.32) with respect to $\{\mathbf{u}^*\}$, the entire target trajectory (7.9) is also moved closer to $\mathbf{u}^*$ with each new trajectory; in particular, by fact 38 this holds true when the optimization method is GD (5.30) for MP1, (5.15), and Proximal Gradient Descent (PG) for Model Problem 2 (MP2), (5.23). Then, if the MGRIT solves on each all-at-once system of the form (7.7) are sufficiently accurate such that the initial guess updates of the form $\widetilde{\mathbf{u}}_0^{(k)} \coloneqq \widetilde{\mathbf{v}}_{N_t}^{(k-1)}$ preserve the Féjer-monotonicity of the combined sequence of target trajectories (7.9), i.e., if the sequence

$$\{\widetilde{\mathbf{u}}_0^{(1)}, \ldots, \widetilde{\mathbf{u}}_{N_t}^{(1)}, \widetilde{\mathbf{u}}_0^{(2)}, \ldots, \widetilde{\mathbf{u}}_{N_t}^{(2)}, \widetilde{\mathbf{u}}_0^{(3)}, \ldots, \widetilde{\mathbf{u}}_0^{(k)}, \ldots\} \tag{7.10}$$

is itself Féjer-monotone with respect to $\{\mathbf{u}^*\}$, then we have that

$$\lim_{k \to +\infty} \widetilde{\mathbf{u}}_i^{(k)} = \mathbf{u}^* \quad \forall i = 0, 1, \ldots, N_t. \tag{7.11}$$

Therefore, for a sufficiently large number k of all-at-once systems, we have that

$$\{\widetilde{\mathbf{u}}_i^{(k)}\}_{0 \leq i \leq N_t} \approx \{\mathbf{u}^*, \ldots, \mathbf{u}^*\}. \tag{7.12}$$

Thus, by solving the system (7.7) for large k, even with relatively low accuracy, we are possibly able to approximate $\mathbf{u}^*$ with higher accuracy than by approximating $\mathbf{u}_{N_t}$ with high accuracy, as is the goal of the single-trajectory P-in-I solve.

7.2.2 The restarting MGRIT algorithm for optimization problems (R-MGRIT)

We now describe the MT P-in-I algorithm which we consider in this thesis, which we refer to as Restarted MGRIT (R-MGRIT), algorithm 8. We adopt this restarted algorithm terminology to relate our framework to the existing literature on restarting optimization methods [74, 57, 5, 45, 61, 79, 88, 86]. In these classical restarting optimization schemes, each restart reinitializes one or more parameters of the underlying optimization algorithm. Here, we treat the initial guess $\widetilde{\mathbf{u}}_0$ of the all-at-once system (7.7) to be the parameter to be reinitialized, which together with the optimization method of the form (5.52) entirely determines the solution $\widetilde{\mathbf{u}}$ and right-hand side $\widetilde{\mathbf{w}}$ of each all-at-once system $\mathcal{A}(\widetilde{\mathbf{u}}) = \widetilde{\mathbf{w}}$ (7.7). Thus, our MT P-in-I framework may be interpreted as a restarting scheme for solving all-at-once systems of the form (5.55), which we previously solved with a standard, single-trajectory P-in-I MGRIT scheme in chapter 5.

Algorithm 8 R-MGRIT($\Phi, \Psi, \mathbf{u}_0, \mathbf{w}, \varepsilon, N_t$).

1: **repeat**
2: Initialize $\mathbf{u}$ given the initial guess $\mathbf{u}_0 \in \mathbb{R}^N$: $\mathbf{u} = \begin{bmatrix}\mathbf{u}_0 & \cdots & \mathbf{u}_0\end{bmatrix}^T \in \mathbb{R}^{NN_t}$.
3: Run 1 iteration of MGRIT($\Phi, \Psi, \mathbf{u}, \mathbf{w}, \varepsilon$) on the all-at-once system $A(\mathbf{u}) = \mathbf{w}$.
4: Set $\mathbf{u}_0 \longleftarrow \mathbf{u}_{N_t}$.
5: **until** $\|G_{sF}(\mathbf{u}_{N_t})\|/\|G_{sF}(\mathbf{u}_0)\| \leq \varepsilon$.

We note that R-MGRIT is one of many possible multi-trajectory P-in-I methods; the general idea is to generate a sequence of all-at-once systems of the form (7.7) and solve each of them inexactly using some parallel-in-time (P-in-T) method (not necessarily MGRIT), using the available information up to the most recent all-at-once system solve to establish the target trajectory of the next all-at-once system. We now make a series of remarks, in which we discuss each of the steps outlined in algorithm 8.

Remark 29 (Constant initialization strategy for R-MGRIT)**.** Notably, in step 2 of algorithm 8 we specified the initialization strategy for the space-time vector $\widetilde{\mathbf{u}}$ of each $\mathcal{A}(\widetilde{\mathbf{u}}) = \widetilde{\mathbf{w}}$ (7.7) system to be a constant space-time vector $\widetilde{\mathbf{u}}^{(0)}$ that simply replicates the (new) initial guess at all time points:

$$\widetilde{\mathbf{u}}^{(0)} = \begin{bmatrix}\widetilde{\mathbf{u}}_0 & \widetilde{\mathbf{u}}_0 & \cdots & \widetilde{\mathbf{u}}_0\end{bmatrix}^T \in \mathbb{R}^{N(N_t+1)}, \tag{7.13}$$

where the $k = 0$ index denotes that this is the approximate-solution vector $\widetilde{\mathbf{u}}$ of (7.7) before any MGRIT iteration is performed on the corresponding all-at-once system. We do

so because, unlike in a standard MGRIT solve, no progress is carried over between each R-MGRIT iteration except for the updated initial guess. Thus, initializing $\widetilde{\mathbf{u}}$ as in (7.13) serves two purposes: *1.* it ensures that every point is initialized to have a relatively small gradient, assuming the new initial guess has a smaller gradient than the previous ones, thus carrying over as much progress as possible, while ensuring that *2.* should we approach a fixed point of the underlying optimization method, we do not deviate away from it.

Other initialization strategies may be considered, such as the random initialization strategy utilized in the numerical experiments from section 5.7. From figure 7.2, in which the approximate-solution vectors of both all-at-once systems (7.1) and (7.3) are randomly initialized, we see this strategy can be relatively ineffective in a multi-trajectory setting if new initial guess has a much smaller gradient than the randomly initialized C-point vectors. By employing the initialization strategy from (7.13) instead, we would instead expect the MGRIT solve on the second trajectory to behave more similarly to that for the first trajectory, such that the right-hand side panel of figure 7.2 would be replaced by a similar plot to that on the left-hand panel, but shifted down to start from the new initial guess. Then, we would expect to require fewer MGRIT iterations on the second-trajectory solve to obtain an approximate solution with the desired 10^{-8} accuracy. Another initialization strategy which might be considered, for example, is initializing $\widetilde{\mathbf{u}}$ with a coarse solve, as is typically done when using parareal [62].

Recalling the setting of figure 7.2, in figure 7.3 we employ the constant initialization strategy from remark 29 for the same problem. We observe that the constant initialization strategy is much more efficient for this problem than the random initialization strategy from figure 7.2, with only 1 MGRIT iteration on the all-at-once system for the second trajectory, (7.3), required to reduce the gradient by a factor of 10^{-8}.

Remark 30 (On the choice of the MGRIT algorithm)**.** We note that the MGRIT operator in step 3 of algorithm 8 must be the Full Approximation Scheme (FAS)-MGRIT algorithm, algorithm 7, if the MGRIT operators $\Phi(\cdot)$ and $\Psi(\cdot)$ are nonlinear. If solving a linear problem, such that each all-at-once system takes the form of the linear system (4.7), we may use the linear MGRIT algorithm, algorithm 6. Note that multilevel and full multigrid (FMG) MGRIT schemes can be used.

Remark 31 (Omitting post-relaxations for R-MGRIT)**.** We note that the usually omitted post F-relaxation step of algorithms 6 and 7 is in principle no longer redundant, since the FCF pre-relaxation performed on the following iteration is now applied to the updated all-at-once system (7.7), rather than the current system. However, F-relaxations only affect F-points; since $N_T = N_t/m$ by construction, the final time point N_t is always a C-point. Hence, R-MGRIT is unaffected by a hypothetical post F-relaxation if we enforce the

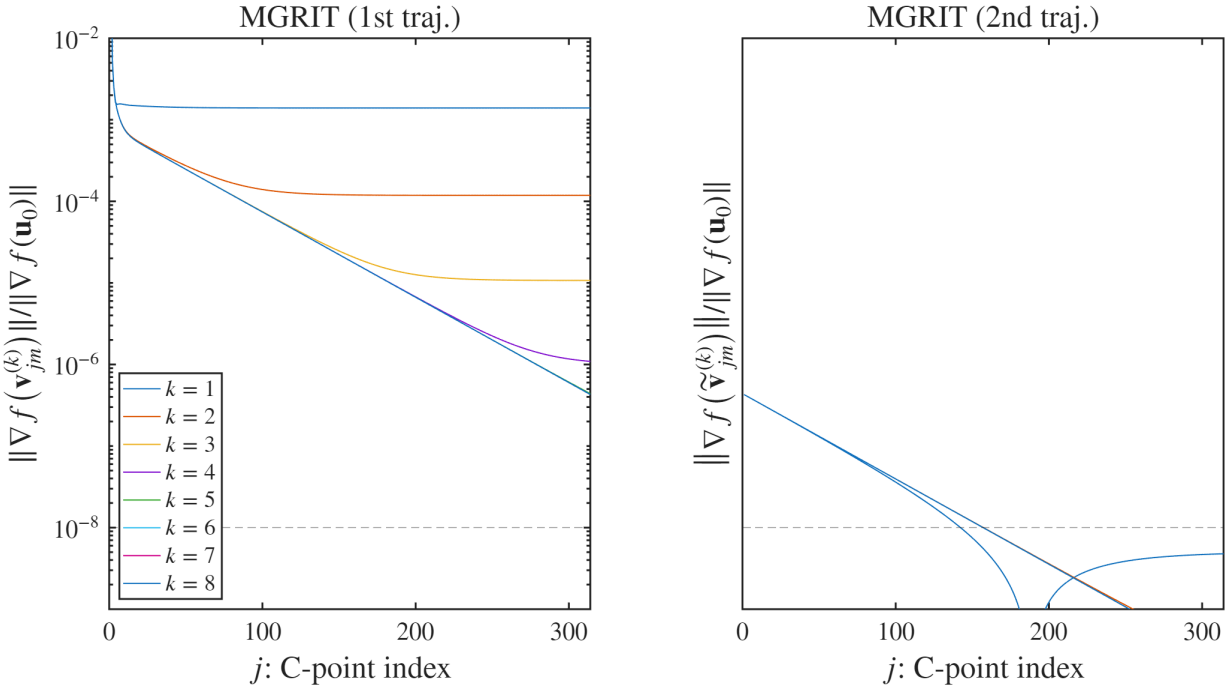


Figure 7.3: Left: 2-norm of $\nabla f\left(\mathbf{v}_{jm}^{(k)}\right)$, where $\mathbf{v}_{jm}^{(k)}$ is the approximate solution at coarse time point jm, $j = 0, \ldots, N_T$, and k is the MGRIT iteration index for the (7.1) system. **Right:** 2-norm of $\nabla f\left(\widetilde{\mathbf{v}}_{jm}^{(k)}\right)$, $\mathbf{v}_{jm}^{(k)}$ is the approximate solution at coarse time point jm, $j = 0, \ldots, N_T$, and k is the MGRIT iteration index for the (7.3) system. Here, $k = 6$ MGRIT iterations is sufficient to reach the desired 10^{-8} accuracy.

condition that the final time point be a C-point. In that case, we omit the post-relaxation stage as in a standard MGRIT solve. Thus, with the computational work measured in terms of relaxation and coarse-grid correction operations, as defined in our speedup formulas in section 5.8, one R-MGRIT iteration requires approximately as much computational work as one standard MGRIT iteration, assuming that the additional cost of initializing each R-MGRIT solve is not significant.

Remark 32 (On the number of MGRIT iterations per restart)**.** In step 3 of algorithm 8, we note that we consider performing only one MGRIT iteration on each all-at-once system of the form (7.7). However, this is simply for ease of presentation; in particular, performing one iteration per restart allows us to use the same index to denote the number of MGRIT iterations and of restarts. In practice, we may perform a different number of MGRIT iterations for each restart as we please, as well as change other parameters such as the

coarsening factor m. In section 7.3, we provide theoretical results, namely theorem 3, based on which we argue for MGRIT that this typically does not make much of a difference; that is, if we perform two MGRIT iterations per restart instead of one, we expect to require half of the number of restarts, although the total number of MGRIT iterations required for convergence will remain the same.

Remark 33 (On the strategy for choosing the new initial guess)**.** In step 4, we establish the strategy of taking the new initial guess $\widetilde{\mathbf{u}}_0^{(k)}$ to be the approximate solution at the final time point of the previous all-at-once system, $\widetilde{\mathbf{v}}_{N_t}^{(k-1)}$, for all $k \geq 2$. This idea comes from the fact that on a highly accurate single-trajectory all-at-once system solve, such as in the numerical experiments from section 5.7, we have that the approximation $\mathbf{v}_{N_t}^{(k)} \approx \mathbf{u}_{N_t}$ becomes highly accurate for a sufficiently large number k of MGRIT iterations, with $\mathbf{u}_{N_t}$ being assumed to minimize the (generalized) gradient over the target trajectory $\{\mathbf{u}\}_{i=0,1,\ldots,N_t}$. However, other strategies may be considered. In particular, in section 7.6 we consider searching for an optimal restart point instead of the naive (but cheap) strategy of taking the approximate solution at the final time point. Note that by changing this initialization strategy, we also need to change the halting criteria in order to evaluate the gradient at whichever point is taken to be the new initial guess.

Remark 34 (On the number N_t of fine-grid time points)**.** We recall from subsection 7.1.1 that the idea for our multi-trajectory strategy arose from considering all-at-once system solves with a number of time points smaller than the number of iterations required for the convergence of the optimization method (5.52). As discussed in subsection 7.2.1, since each restart moves the target trajectory to be approximated by MGRIT closer to $\mathbf{u}^*$ if the underlying optimization method is monotone, we expect algorithm 8 to converge independently of the choice for N_t.

Note that choosing very small N_t may lead the algorithm to converge very slowly, requiring a very large number of restarts (equivalently, MGRIT iterations). However, as shown in sections 7.4 and 7.5, taking a number of time points smaller than N_t often slightly reduces the number of iterations required for convergence while simultaneously improving the parallel speedup. On the other hand, choosing larger values of N_t is expected to not improve convergence and reduce the expected parallel speedup, given the theoretical results derived in section 7.3.

Remark 35 (On the different halting criteria for R-MGRIT)**.** We note that in algorithms 6 and 7, the halting criteria is given by reducing the initial MGRIT residual (4.4) by a factor of ε. On the standard single-trajectory MGRIT P-in-I solves discussed in chapter 5, we recall from section 5.6 that we expect the (generalized) gradient of the approximate solution

at the final time point to be reduced by approximately the same factor ε than what the initial residual (4.4) is reduced by. However, by updating the trajectories (7.9) with each restart, we are able to reduce the (generalized) gradient by a factor equal to or possibly larger than ε, such that we define the halting criteria in terms of the (generalized) gradient rather than the residual, which is less meaningful in this context. Since we now require $\|G_{sF}(\mathbf{v}_{N_t})\| \leq \varepsilon \|G_{sF}(\mathbf{u}_{N_t})\|$ rather than $\|G_{sF}(\mathbf{v}_{N_t})\| \approx \varepsilon \|G_{sF}(\mathbf{u}_{N_t})\|$, we note that this is a slightly stricter criteria than that in algorithms 6 and 7. Furthermore, note that if solving a problem of the form (5.1), the generalized gradients G_{sF} in step 9 are simply the standard ∇f gradients, as we observe by taking $g = 0$ in (5.145).

Finally, in the next remark we note that our idea of chaining approximate solves for multiple target trajectories cannot be applied for solving the typical time integration problems of the form (4.1) MGRIT was originally developed for.

Remark 36 (Multi-trajectory framework does not work for time integration problems)**.** For example, consider the 1D heat equation IBVP from (5.42). Recall from subsection 5.4.2 that the all-at-once system (5.50) for an MGRIT P-in-T solve of the discretized 1D heat equation is exactly the same system as the one obtained for solving MP1, (5.15), P-in-I using GD (3.49). However, the meaning of the initial condition $\mathbf{u}_0 = \mathbf{w}_0$ from (5.50) is completely different for each problem.

For MP1, (5.15), replacing $\mathbf{u}_0$ by $\widetilde{\mathbf{u}}_0$ in the initial condition of (5.50) simply changes the initial guess for the minimization problem without affecting $\mathbf{u}^* = A^{-1}\mathbf{b}$, the exact minimizer of MP1. In particular, in this optimization context it is preferable to start from an initial guess as close as possible to $\mathbf{u}^*$, in order to reduce the number of iterations (3.49) of GD to converge. This is exactly the idea behind our MT P-in-I strategy: to progressively update the initial guess to be closer to the solution, such that less accuracy in approximating the corresponding GD trajectory is required to satisfactorily reduce the gradient.

On the other hand, for the heat equation, $\mathbf{u}_0 = \mathbf{w}_0$ corresponds to a discretization of the initial condition

$$u(x, 0) = u_0(x), \quad x \in \Omega, \tag{7.14}$$

of the continuous heat equation (5.42). Therefore, changing the initial condition of the all-at-once system to $\widetilde{\mathbf{u}}_0 = \mathbf{w}_0$, for example, means that we are changing the physical properties of the IBVP. Thus, by changing the initial condition we are solving a fundamentally different problem; in this example, with different initial temperatures at each point in the domain. Therefore, our proposed MT MGRIT strategy is only applicable in the context of P-in-I MGRIT solves, as discussed in section 5.5.

7.3 Asymptotic final-point analysis of MGRIT and R-MGRIT

In the following, we compare the linear MGRIT algorithm, algorithm 6, and R-MGRIT, algorithm 8, assuming that we only perform one MGRIT iteration on each all-at-once system (7.7). Hence, we may use the same index k to denote the number of MGRIT iterations performed by algorithm 6 on the linear all-at-once system (4.7) and the number of restarts of algorithm 8, i.e., the number of updated all-at-once systems generated by this procedure; we simply refer to k as the iteration number of each algorithm. In our numerical experiments, later presented in section 7.4, we observe that both MGRIT and R-MGRIT appear to have a similar convergence speed. Based on the linear MGRIT error propagation theory developed in [24, 89] and introduced in section 4.4, we prove the following convergence result for 2-level linear MGRIT. Assuming the constant initialization strategy outlined in remark 29, if the number of coarse intervals N_T is sufficiently large then the approximate solutions at the final time point $N_t = N_T m$ of the kth MGRIT and R-MGRIT is approximately the same for all k, with the end points of the C-point approximate solution trajectories of both methods converging to the same asymptotic limit as $N_T \to +\infty$; see theorem 3 for the main result.

For simplicity and concreteness, we consider solving MP1, (5.15), with $\mathbf{b} = \mathbf{0}$, such that $\mathbf{u}^* = \mathbf{0}$ is the unique minimizer. Then, the GD iteration for MP1 (5.30) is simplified to

$$\mathbf{u}_j = \Phi \mathbf{u}_{j-1} = \Phi^j \mathbf{u}_0, \quad j = 1, 2, \ldots, \tag{7.15}$$

where $\Phi \coloneqq I - s\nabla f = I - sA_1$; if Φ is not specified, then our analysis applies for any iterative method described by (7.15) for some linear operator Φ. Although in our analysis we work with MP1, (5.15), under the assumption that $\mathbf{b} = \mathbf{0}$, such that $\mathbf{u}^* = \mathbf{0}$, we note that our analysis applies to the more general case of $\mathbf{b} \neq \mathbf{0}$ and $\mathbf{u}^* = (A_1)^{-1}\mathbf{b}$ up to a change of coordinates, as discussed in remark 37.

Remark 37 (Coordinate shifts for nonzero minimizers)**.** We note that (7.15) describes the general MP1 with nonzero $\mathbf{b}$ up to a coordinate shift of the ith spatial dimension by b_i units, where $i = 1, \ldots, n$ and $\mathbf{b} = \begin{bmatrix} b_1 & \cdots b_n \end{bmatrix}^T$. Since MGRIT is a linear fixed-point iterative method (see section 2.3), it is invariant to affine transformations such as the translation above; that is, a sequence of MGRIT approximate solutions $\{\mathbf{u}_j\}$ converges to the minimizer $\mathbf{u}^* = (A_1)^{-1}\mathbf{b}$ of (5.15) if and only if the shifted sequence converges to $\mathbf{u}^* = \mathbf{0}$.

In the following, we first explicitly determine the MGRIT error propagation matrix for

k MGRIT iterations, generalizing some of the theory first presented in section 4.4 for 1 iteration. Then, we characterize the MGRIT and R-MGRIT approximate solutions at the final time point N_t in terms of the MGRIT parameters, operators and error propagation matrix. Finally, we prove that MGRIT and R-MGRIT end-point approximate solutions converge to the same asymptotic limit for fixed iteration k in the limit of $N_T \to +\infty$ and that their difference for finite N_T is bounded, as well as some related additional results. In this chapter, we let Φ and Ψ denote the fine- and coarse-grid operators, respectively, as we did in subsection 5.5.1.

7.3.1 MGRIT error propagation after k iterations

In what follows, let Φ and Ψ denote the MGRIT fine- and coarse-grid operators, respectively. From (4.89), we recall that the MGRIT C-point error propagation block matrix $\mathcal{E}$ is given block-wise by :

$$(\mathcal{E})_{i,j} = \begin{cases} \Psi^{i-j-2}(\Phi^m - \Psi)\Phi^m & \text{if } i-j \geq 2, \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{7.16}$$

The block matrix $\mathcal{E}$ then propagates the error at C-points after k MGRIT iterations by the relation

$$\mathbf{e}_c^{(k)} = \mathcal{E}^k \mathbf{e}_c^{(0)}, \tag{7.17}$$

where $\mathbf{e}_c^{(0)}$ and $\mathbf{e}_c^{(k)}$ denote the C-point initial error and the C-point error after k iterations, respectively. By taking the kth power of $\mathcal{E}$ (4.89), we explicitly determine the error propagation matrix $\mathcal{E}^k$ for k MGRIT iterations.

Proposition 4 (Block-components of $\mathcal{E}^k$)**.** *For any iteration $k \geq 1$, the block-components of $\mathcal{E}^k$ are given by:*

$$(\mathcal{E}^k)_{i,j} = \begin{cases} \binom{i-j-k-1}{k-1} \Psi^{i-j-2k}((\Phi^m - \Psi)\Phi^m)^k & \textit{if } i-j \geq 2k, \\ \mathbf{0} & \textit{otherwise.} \end{cases} \tag{7.18}$$

Proof. By induction on k. The base case $k = 1$ trivially matches (4.89) since $\binom{i-j-2}{0} = 1$. Now, assume the formula holds for $k-1$. We compute the (i,j)-th block of $\mathcal{E}^k$ via the block matrix product

$$(\mathcal{E}^k)_{i,j} = \sum_{p=0}^{N_T} (\mathcal{E})_{i,p} (\mathcal{E}^{k-1})_{p,j}. \tag{7.19}$$

Note that the terms in the sum are non-zero only if $i-p \geq 2$ and $p-j \geq 2(k-1)$. Thus, if $i-j < 2k$, we have that $(\mathcal{E}^k)_{i,j} = \mathbf{0}$. For $i \geq j+2k$, substituting (7.18) into (7.19) yields

$$(\mathcal{E}^k)_{i,j} = \Psi^{i-j-2k}((\Phi^m - \Psi)\Phi^m)^k \sum_{p=j+2k-2}^{i-2} \binom{p-j-k}{k-2}. \tag{7.20}$$

Recall the hockey-stick (or Fermat's) identify for non-negative integers $n \geq r$,

$$\sum_{i=r}^{n} \binom{i}{r} = \binom{n+1}{r+1}. \tag{7.21}$$

Applying the change of variables $r = p-j-k$, we get that

$$\sum_{p=j+2k-2}^{i-2} \binom{p-j-k}{k-2} = \sum_{r=k-2}^{i-j-k-2} \binom{r}{k-2} = \binom{i-j-k-1}{k-1}, \tag{7.22}$$

where the final equality comes from applying (7.21). □

From proposition 4, we generalize lemma 2 from [24] to determine the C-point errors after k iterations.

Lemma 7 (C-point errors after k iterations)**.** *After k iterations of MGRIT with FCF-relaxation, the approximate solution at the ith C-point satisfies $\mathbf{v}_{im}^{(k)} = \mathbf{u}_{im} - \mathbf{e}_{im}^{(k)}$, where the error at the C-points satisfies*

$$\mathbf{e}_0^{(k)} = \mathbf{e}_m^{(k)} = \mathbf{0}, \tag{7.23}$$

$$\mathbf{e}_{im}^{(k)} = \sum_{j=0}^{i-2k} \binom{i-j-k-1}{k-1} \Psi^{i-j-2k}((\Phi^m - \Psi)\Phi^m)^k \mathbf{e}_{jm}^{(0)}, \quad i = 2k, \ldots, N_T. \tag{7.24}$$

Proof. The error at the i-th C-point, $\mathbf{e}_{im}^{(k)}$, is determined by the product of the i-th block row of $\mathcal{E}^k$ with the initial C-point error vector $\begin{bmatrix}\mathbf{e}_0 & \mathbf{e}_m & \cdots & \mathbf{e}_{N_T m}\end{bmatrix}^T$:

$$\mathbf{e}_{im}^{(k)} = \sum_{j=0}^{N_T} (\mathcal{E}^k)_{i,j} \mathbf{e}_{jm}^{(0)}. \tag{7.25}$$

The statement then follows from substituting (7.18) into (7.25). □

7.3.2 Characterizing (R-)MGRIT final-point approximate solutions

For a constant initial guess $\mathbf{v}^{(0)} = \begin{bmatrix}\mathbf{u}_0 & \cdots & \mathbf{u}_0\end{bmatrix}^T$ (7.13), the initial error at each C-point is $\mathbf{e}_{im}^{(0)} = (\Phi^{im} - I)\mathbf{u}_0$ for $i = 0, \ldots, N_T$. Substituting this into lemma 7, the error after k iterations at the final coarse time point N_T is given by $\mathbf{e}_{N_t}^{(k)} = E_{N_T}^{(k)}\mathbf{u}_0$, where

$$E_{N_T}^{(k)} = \sum_{j=0}^{N_T-2k} \binom{N_T - j - k - 1}{k-1} \Psi^{N_T-j-2k}((\Phi^m - \Psi)\Phi^m)^k(\Phi^{jm} - I). \tag{7.26}$$

By lemma 7, since $N_t = N_T m$ is a C-point by construction we have that $\mathbf{v}_{N_t}^{(k)} = \mathbf{u}_{N_t} - \mathbf{e}_{N_t}^{(k)}$, where $\mathbf{u}_{N_t} = \Phi^{N_t}\mathbf{u}_0$ and $\mathbf{e}_{N_t}^{(k)} = E_{N_T}^{(k)}\mathbf{u}_0$. Then, the MGRIT approximation after k iterations at the final time point N_t, $\mathbf{v}_{N_t}^{(k)}$, can be expressed in terms of Φ and $E_{N_T}^{(k)}$ by

$$\mathbf{v}_{N_t}^{(k)} = V_{N_t}^{(k)}\mathbf{u}_0 \coloneqq (\Phi^{N_t} - E_{N_T}^{(k)})\mathbf{u}_0. \tag{7.27}$$

Let $\widetilde{V}_{N_t}$ denote the operator that yields the MGRIT approximate solution (7.27) after one iteration, i.e., with $k = 1$:

$$\widetilde{V}_{N_t} \coloneqq V_{N_t}^{(1)}. \tag{7.28}$$

For R-MGRIT, after one MGRIT iteration we take the approximate solution at the final time point, $\mathbf{v}_{N_t}^{(1)}$, to serve as the new initial guess for a second, R-MGRIT iteration:

$$\widetilde{\mathbf{u}}_0^{(2)} = \mathbf{v}_{N_t}^{(1)} = \widetilde{V}_{N_t}\mathbf{u}_0. \tag{7.29}$$

Note that this second restarted iteration is identical to the first MGRIT iteration, except for the different initial guess. Hence, to obtain the approximate solution at the final time point for the second iteration we simply apply the same single-iteration operator $\widetilde{V}_{N_t}$ from (7.28) to the new initial guess:

$$\widetilde{\mathbf{v}}_{N_t}^{(2)} = \widetilde{V}_{N_t}\widetilde{\mathbf{u}}_0^{(2)} = \widetilde{V}_{N_t}\big(\widetilde{V}_{N_t}\mathbf{u}_0\big) = \widetilde{V}_{N_t}^2\mathbf{u}_0. \tag{7.30}$$

Let $\widetilde{\mathbf{u}}_0^{(k)}$ be the starting vector for the k-th R-MGRIT iteration. Applying (7.30) recursively, the approximate solution at the final time point is exactly the k-th power of the single-iteration operator $\widetilde{V}_{N_t}$ applied to the original initial guess:

$$\widetilde{\mathbf{v}}_{N_t}^{(k)} = \widetilde{\mathbf{u}}_{N_t}^{(k)} - \widetilde{\mathbf{e}}_{N_t}^{(k)} = \widetilde{V}_{N_t}\widetilde{\mathbf{u}}_0^{(k-1)} = \widetilde{V}_{N_t}^k\mathbf{u}_0. \tag{7.31}$$

Therefore, by (7.28) and (7.27) we have that the R-MGRIT approximate solution after k iterations at the final time N_t, $\widetilde{\mathbf{v}}_{N_t}^{(k)}$, is explicitly described by

$$\widetilde{\mathbf{v}}_{N_t}^{(k)} = \widetilde{V}_{N_t}^{k}\mathbf{u}_0 = (\Phi^{N_t} - E_{N_T}^{(1)})^k \mathbf{u}_0. \tag{7.32}$$

We now provide two alternative characterizations of the $V_{N_t}^{(k)}$ and $\widetilde{V}_{N_t}^{k}$ operators.

Remark 38 (An algebraic characterization of $V_{N_t}^{(k)}$ and $\widetilde{V}_{N_t}^{k}$)**.** In order to more intuitively describe the difference between the MGRIT and R-MGRIT approximate solutions, we consider an alternative description of the $V_{N_t}^{(k)}$ and $\widetilde{V}_{N_t}^{k}$ operators, which we previously described in (7.27) and (7.32), respectively, in terms of the Φ and $E_{N_T}^{(k)}$ (7.26) operators. Let I denote the $N \times N$ spatial identity matrix, and let $\mathcal{I}$ denote the $N(N_t+1) \times N(N_t+1)$ space-time block-identity matrix. We define the following operators to streamline the notation:

- **Broadcasting (B):** Generates a constant space-time trajectory block-vector from a single spatial vector by duplicating it across all time points:

$$B = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \otimes I = \begin{bmatrix} I \\ I \\ \vdots \\ I \end{bmatrix} \in \mathbb{R}^{N(N_T+1)\times N}. \tag{7.33}$$

- **Powers ($\mathcal{P}$):** Generates the exact sequential fine-grid trajectory from (7.15) by applying increasing powers of the spatial operator Φ to the initial guess; when applied to $B\mathbf{u}_0$,

$$\mathcal{P} = \mathrm{diag}(I, \Phi, \ldots, \Phi^{N_t}) = \begin{bmatrix} I & & & \\ & \Phi & & \\ & & \ddots & \\ & & & \Phi^{N_t} \end{bmatrix} \in \mathbb{R}^{N(N_T+1)\times N(N_T+1)}. \tag{7.34}$$

- **Extraction (X):** Extracts the final time block from an $N(N_T+1)$ space-time trajectory block-vector, or the final row from an $N(N_T+1) \times N(N_T+1)$ space-time block-matrix:

$$X = \begin{bmatrix} 0 & 0 & \ldots & 1 \end{bmatrix} \otimes I = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \ldots & I \end{bmatrix} \in \mathbb{R}^{N\times N(N_T+1)}. \tag{7.35}$$

For example, using the operators defined in (7.33) to (7.35), the space-time initial trajectory $\mathbf{v}^{(0)}$ and the target space-time trajectory stemming from $\mathbf{u}_0$, given by $\mathbf{v}^{(0)} = B\mathbf{u}_0$ and $\mathbf{u} = \mathcal{P}B\mathbf{u}_0$, respectively, and the final time point of the target trajectory, $\mathbf{u}_{N_t} = X\mathbf{u}$, take the forms

$$\mathbf{v}^{(0)} = B\mathbf{u}_0 = \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_0 \\ \vdots \\ \mathbf{u}_0 \end{bmatrix}, \quad \mathbf{u} = \mathcal{P}B\mathbf{u}_0 = \begin{bmatrix} \mathbf{u}_0 \\ \Phi\mathbf{u}_0 \\ \vdots \\ \Phi^{N_t}\mathbf{u}_0 \end{bmatrix}, \quad \mathbf{u}_{N_t} = X\mathbf{u} = \Phi^{N_t}\mathbf{u}_0. \tag{7.36}$$

First, consider k iterations of a standard linear MGRIT solve using the algorithm 6. Initializing MGRIT with a constant initial trajectory $\mathbf{v}^{(0)} = B\mathbf{u}_0$ yields an initial space-time error

$$\mathbf{e}^{(0)} = \mathbf{u} - \mathbf{v}^{(0)} = \mathcal{P}B\mathbf{u}_0 - B\mathbf{u}_0 = (\mathcal{P} - \mathcal{I})B\mathbf{u}_0. \tag{7.37}$$

Let $\widehat{\mathcal{E}}$ be the full MGRIT error propagation matrix, which takes the form of (4.75) up to a reordering in terms of F- and C-points; see section 4.3. After k iterations, MGRIT error propagation takes the form

$$\mathbf{e}^{(k)} = \widehat{\mathcal{E}}^k\mathbf{e}^{(0)} = \widehat{\mathcal{E}}^k(\mathcal{P} - \mathcal{I})B\mathbf{u}_0. \tag{7.38}$$

The kth iteration approximate solution space-time vector is given by $\mathbf{v}^{(k)} = \mathbf{u} - \mathbf{e}^{(k)}$, such that $\mathbf{v}^{(k)}_{N_t} = \mathbf{u}_{N_t} - \mathbf{e}^{(k)}_{N_t}$. In terms of the operators from (7.33) to (7.35), we extract the final time block from the target trajectory $\mathbf{u}$,

$$\mathbf{u}_{N_t} = X\mathbf{u} = X\mathcal{P}B\mathbf{u}_0 = \Phi^{N_t}\mathbf{u}_0, \tag{7.39}$$

and the propagated error $\mathbf{e}^{(k)}$,

$$\mathbf{e}^{(k)}_{N_t} = X\mathbf{e}^{(k)} = X\widehat{\mathcal{E}}^k(\mathcal{P} - \mathcal{I})B\mathbf{u}_0 = X\mathcal{E}^k(\mathcal{P} - \mathcal{I})B\mathbf{u}_0, \tag{7.40}$$

where the last equality comes from the fact that since $N_t = N_T m$ (i.e., the last time point is a C-point), the final row of $\widehat{\mathcal{E}}^k$ is identical to the final row of the C-point error propagation matrix $\mathcal{E}^k$ (4.89). Hence, the MGRIT approximate solution at the final time point can be written as a direct mapping from the initial guess $\mathbf{u}_0$:

$$\mathbf{v}^{(k)}_{N_t} = X(\mathbf{u} - \mathbf{e}^{(k)}) = X(\mathcal{P} - \mathcal{E}^k(\mathcal{P} - \mathcal{I}))B\mathbf{u}_0 \coloneqq V^{(k)}_{N_t}\mathbf{u}_0. \tag{7.41}$$

Next, consider R-MGRIT with k trajectories. Let $\widetilde{V}_{N_t}$ denote the operator that yields the MGRIT approximate solution after one iteration, as defined in (7.28) and (7.41):

$$\widetilde{V}_{N_t} \coloneqq V^{(1)}_{N_t} = X(\mathcal{P} - \mathcal{E}(\mathcal{P} - \mathcal{I}))B. \tag{7.42}$$

Then, by (7.32), we may write the R-MGRIT approximate solution at the final time point as

$$\widetilde{\mathbf{v}}_{N_t}^{(k)} = \widetilde{V}_{N_t}^k \mathbf{u}_0 = (X(\mathcal{P} - \mathcal{E}(\mathcal{P} - \mathcal{I}))B)^k \mathbf{u}_0. \tag{7.43}$$

While this algebraic formulation unfortunately does not help our upcoming calculations, it serves to highlight the difference between the MGRIT and R-MGRIT approaches. On one hand, MGRIT generates its approximation solution at the final time point N_t after k iterations via the operator from (7.41),

$$V_{N_t}^{(k)} = X(\mathcal{P} - \mathcal{E}^k(\mathcal{P} - \mathcal{I}))B, \tag{7.44}$$

which broadcasts the initial condition using B, computes the approximate solution using $\mathcal{E}^k$ and then extracts the final time point using X. On the other hand, by (7.43) R-MGRIT instead uses

$$\widetilde{V}_{N_t}^k = (X(\mathcal{P} - \mathcal{E}(\mathcal{P} - \mathcal{I}))B)^k. \tag{7.45}$$

Note then that for every restart, the initial condition is broadcast using B, the approximate solution obtained using $\mathcal{E}$ and the final time point extracted using X. A new trajectory is then generated by broadcasting this approximate solution at the final time point, which is taken to be the new initial guess, such that this procedure of broadcasting, error propagation and extraction repeats k times. Notably, in (7.44) the power of k applies only to the C-point error propagation matrix $\mathcal{E}$, while in (7.45) the power of k applies to the entire $\widetilde{V}_{N_t} = X(\mathcal{P} - \mathcal{E}(\mathcal{P} - \mathcal{I}))B$ operator.

Remark 39 (MGRIT-operator characterization of $\widetilde{V}_{N_t}$)**.** To obtain another characterization of the MGRIT single-iteration operator $\widetilde{V}_{N_t}$ in (7.42), this time in terms of the MGRIT operators Φ and Ψ and the coarsening factor m, we now describe the first MGRIT iteration of an all-at-once system solve of the form (4.7) with initial guess $\mathbf{u}_0$, assuming the constant initialization from (7.13). We recall the steps of the linear MGRIT algorithm, algorithm 6, as described in subsection 4.2.1. Under the constant initialization assumption from remark 29, we are able to simplify the evaluation of these steps and describe the state of the relevant vectors at all time points in terms of the initial guess, as follows.

1. **Pre-relaxation:** Given the constant initialization $\mathbf{v}^{(0)} = \begin{bmatrix} \mathbf{u}_0 & \cdots & \mathbf{u}_0 \end{bmatrix}^T$ (7.13), such that all C-points are initialized to be equal to the initial guess $\mathbf{u}_0$, the first F- (4.5) and C- (4.6) relaxations yield

$$\begin{aligned} \mathbf{v}_0^{(1)} &= \mathbf{u}_0, \\ \mathbf{v}_{jm}^{(1)} &= \Phi^m \mathbf{u}_0, \quad j = 1, \ldots, N_T. \end{aligned} \tag{7.46}$$

Consequently, the F-points immediately preceding the C-points are updated from these new C-points by the second F-relaxation (4.5) as $\mathbf{v}_{jm-1}^{(1)} = \Phi^{m-1}\mathbf{v}_{(j-1)m}^{(1)}$, which then yields

$$\begin{aligned} \mathbf{v}_{m-1}^{(1)} &= \Phi^{m-1}\mathbf{u}_0, \\ \mathbf{v}_{jm-1}^{(1)} &= \Phi^{2m-1}\mathbf{u}_0, \quad j = 2, \ldots, N_T. \end{aligned} \tag{7.47}$$

2. **Coarse-grid residual:** Because the first coarse interval is computed exactly by the FCF-relaxation, $\mathbf{r}_0 = \mathbf{r}_m = \mathbf{0}$; see lemma 2 for an equivalent result in the error. For all subsequent C-points, i.e., $j \geq 2$, by (4.24) the residual $\mathbf{r}^{(1)} = \mathbf{w}^{(1)} - \mathcal{A}\mathbf{v}^{(1)}$ (4.4) at C-points is the same:

$$\begin{aligned} \mathbf{r}_{jm}^{(1)} &= \Phi\mathbf{v}_{jm-1}^{(1)} - \mathbf{v}_{jm}^{(1)} \\ &= \Phi(\Phi^{2m-1}\mathbf{u}_0^{(1)}) - \Phi^m\mathbf{u}_0 \\ &= (\Phi^{2m} - \Phi^m)\mathbf{u}_0, \end{aligned} \tag{7.48}$$

which we factor and denote by

$$\overline{\mathbf{r}}^{(1)} := (\Phi^m - I)\Phi^m\mathbf{u}_0. \tag{7.49}$$

3. **Coarse-grid correction:** The correction vectors (4.26) are obtained by propagating the residual $\overline{\mathbf{r}}^{(1)}$ (7.49) across the coarse grid via the coarse operator Ψ. The corrected C-points (4.27) of index $j \geq 2$ are then given by

$$\begin{aligned} \widetilde{\mathbf{v}}_{jm}^{(1)} &= \Phi^m\mathbf{u}_0 + \sum_{i=0}^{j-2} \Psi^i\overline{\mathbf{r}}^{(1)} \\ &= \left(I + \sum_{i=0}^{j-2} \Psi^i(\Phi^m - I)\right)\Phi^m\mathbf{u}_0, \quad j = 2, \ldots, N_T; \end{aligned} \tag{7.50}$$

recall that there is no correction for $j = 0, 1$ since $\mathbf{r}_0 = \mathbf{r}_m = \mathbf{0}$.

In particular, taking $j = N_T$ in (7.50), together with (7.27) (where $k = 1$) and (7.28) yields

$$\widetilde{V}_{N_t} := V_{N_t}^{(1)} = \left(I + \sum_{i=0}^{N_T-2} \Psi^i(\Phi^m - I)\right)\Phi^m. \tag{7.51}$$

We note that similar characterizations of $V_{N_t}^{(k)}$ (7.27) for each $k \geq 2$ may be obtained by following similar steps.

7.3.3 Error operators under a simultaneous diagonalizability assumption

Now, we assume that the fine- and coarse-grid operators, Φ and Ψ, respectively, can be simultaneously diagonalized by the same unitary transformation Q (4.98), such that (4.99) is satisfied with ϕ_n and ψ_n denoting the nth eigenvalues of Φ and Ψ, respectively. We further assume that Φ and Ψ are stable, such that $\rho(\Psi) < 1$ and $\rho(\Phi) < 1$. In particular, we proved in subsection 5.5.1 that this assumption holds for $\Phi = I - sA_1$ (5.56) and $\Psi = (I + msA_1)^{-1}$ (5.57), the MP1 operators which we consider in our numerical experiments from sections 5.7 and 7.4.

Following similar steps to those in subsection 5.5.1 for simultaneously diagonalizing Φ and Ψ, it was shown in [24] that after an eigenvector expansion of $\mathbf{e}_c^{(1)} = \mathcal{E}\mathbf{e}_c^{(0)}$ on the basis of eigenvectors given by the columns of Q (4.98), we obtain

$$\mathcal{E}_n = \begin{bmatrix} 0 & & & & & \\ 0 & 0 & & & & \\ 1 & 0 & 0 & & & \\ \psi_n & 1 & 0 & 0 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ \psi_n^{N_T-2} & \cdots & \cdots & 1 & 0 & 0 \end{bmatrix} (\phi_n^m - \psi_n)\phi_n^m, \tag{7.52}$$

which corresponds to the action of $\mathcal{E}$ (4.89) on the nth mode $n = 1, 2, \ldots, N$. Then, analogously to (4.106), we have that

$$\|\mathcal{E}_n\|_1 = \|\mathcal{E}_n\|_\infty = |\phi_n^m - \psi_n| \frac{(1 - |\psi_n|^{N_T-1})}{(1 - |\psi_n|)} |\phi_n^m|, \tag{7.53}$$

where we note that the maximum row sum is given by the last row, with coarse-time index N_T. Analogously to (4.107), the norm in (7.53) is strictly upper-bounded by its asymptotic limit,

$$\|\mathcal{E}_n\|_\infty \leq \lim_{N_T \to +\infty} |\phi_n^m - \psi_n| \frac{(1 - |\psi_n|^{N_T-1})}{(1 - |\psi_n|)} |\phi_n^m| = \frac{|(\phi_n^m - \psi_n)||\phi_n^m|}{1 - |\psi_n|}. \tag{7.54}$$

Similarly, it follows from proposition 4 that the action of $\mathcal{E}^k$ on the nth mode is given by $\mathcal{E}_n^k$, where the components of this matrix are given by

$$(\mathcal{E}_n^k)_{i,j} = \begin{cases} \binom{i-j-k-1}{k-1} \psi_n^{i-j-2k} ((\phi_n^m - \psi_n)\phi_n^m)^k & \text{if } i - j \geq 2k, \\ 0 & \text{otherwise.} \end{cases} \tag{7.55}$$

Following similar simultaneous diagonalization steps with respect to Q (4.98), it also follows that the modal equivalent of $E_{N_T}^{(k)}$ from (7.26) for the nth mode is

$$E_{N_T,n}^{(k)} = ((\phi_n^m - \psi_n)\phi_n^m)^k \sum_{j=0}^{N_T-2k} \binom{N_T - j - k - 1}{k-1} \psi_n^{N_T-j-2k}(\phi_n^{jm} - 1), \tag{7.56}$$

such that the modal equivalents of (7.27) and (7.32) are

$$v_{N_t,n}^{(k)} = V_{N_t,n}^{(k)} u_{0,n} = (\phi_n^{N_t} - E_{N_T,n}^{(k)}) u_{0,n}, \tag{7.57}$$

and

$$\widetilde{v}_{N_t,n}^{(k)} = \widetilde{V}_{N_t,n}^{k} u_{0,n} = (\phi_n^{N_t} - E_{N_T,n}^{(1)})^k u_{0,n}, \tag{7.58}$$

respectively. We now establish the asymptotic modal error operator as $N_T \to +\infty$.

Lemma 8 (Asymptotic modal error operator)**.** *As $N_T \to +\infty$, the modal error operator $E_{N_T,n}^{(k)}$ from (7.56) asymptotically converges to a well-defined limit modal operator*

$$E_{\infty,n}^{(k)} := -\left(\frac{(\phi_n^m - \psi_n)\phi_n^m}{1 - \psi_n}\right)^k = \lim_{N_T \to +\infty} E_{N_t,n}^{(k)}. \tag{7.59}$$

Proof. Applying the change of variables $p = N_T - 2k - j$ reverses the summation order, rewriting the modal error operator from (7.56) as:

$$E_{N_T,n}^{(k)} = ((\phi_n^m - \psi_n)\phi_n^m)^k \sum_{p=0}^{N_T-2k} \binom{p + k - 1}{k-1} \psi_n^p(\phi_n^{(N_T-2k-p)m} - 1). \tag{7.60}$$

Let $M = N_T - 2k$. We can split the summation into two distinct parts and evaluate their limits as $M \to \infty$. For the first part involving the ϕ_n term, let

$$S_M = \sum_{p=0}^{M} \binom{p + k - 1}{k-1} \psi_n^p \phi_n^{(M-p)m}. \tag{7.61}$$

To bound this sum, let $\rho = \max(|\psi_n|, |\phi_n^m|)$. By hypothesis, both $|\phi_n| < 1$ and $|\psi_n| < 1$, which implies $\rho < 1$. Taking the absolute value yields:

$$|S_M| \leq \sum_{p=0}^{M} \binom{p + k - 1}{k-1} |\psi_n|^p |\phi_n^m|^{M-p} \leq \sum_{p=0}^{M} \binom{p + k - 1}{k-1} \rho^p \rho^{M-p} = \rho^M \sum_{p=0}^{M} \binom{p + k - 1}{k-1}. \tag{7.62}$$

Using the hockey-stick identity, the sum of the binomial coefficients evaluates to $\binom{M+k}{k}$, giving the bound:

$$|S_M| \leq \binom{M+k}{k} \rho^M. \tag{7.63}$$

The expression $\binom{M+k}{k}$ is a polynomial in M of degree k, while ρ^M is an exponentially decaying term. Since exponential decay always dominates polynomial growth as $M \to \infty$, we have $\lim_{M\to+\infty} \binom{M+k}{k}\rho^M = 0$, meaning $\lim_{M\to+\infty} S_M = 0$.

The second part of the summation in (7.60) is the partial sum of the negative binomial series. As $M \to \infty$, this yields the Maclaurin series expansion for $-(1-\psi_n)^{-k}$:

$$\lim_{M\to+\infty} -\sum_{p=0}^{M} \binom{p+k-1}{k-1} \psi_n^p = -(1-\psi_n)^{-k}. \tag{7.64}$$

Because $|\psi_n| < 1$, this series converges absolutely. Substituting these two limits back into the original expression yields

$$\lim_{N_T\to+\infty} E^{(k)}_{N_T,n} = ((\phi_n^m - \psi_n)\phi_n^m)^k \left(0 - \frac{1}{(1-\psi_n)^k}\right) = -\left(\frac{(\phi_n^m - \psi_n)\phi_n^m}{1-\psi_n}\right)^k, \tag{7.65}$$

which recovers the definition of $E^{(k)}_{\infty,n}$ in (7.59). □

We now prove that the modal error operator $E^{(k)}_{N_T,n}$ (7.56) is bounded by the maximum absolute row sum of the modal C-point error propagation matrix $\mathcal{E}_n^k$ (7.55).

Lemma 9 (Finite-time modal error operator bound)**.** *The absolute value of the finite-time modal error operator $E^{(k)}_{N_T,n}$ is bounded by $|E^{(k)}_{N_T,n}| \leq B^{(k)}_{N_T,n}$, where*

$$B^{(k)}_{N_T,n} := |(\phi_n^m - \psi_n)|^k |\phi_n^m|^k \sum_{j=0}^{N_T-2k} \binom{N_T-j-k-1}{k-1} |\psi_n|^{N_T-j-2k} = \|\mathcal{E}_n^k\|_\infty. \tag{7.66}$$

Proof. To prove $|E^{(k)}_{N_T,n}| \leq B^{(k)}_{N_T,n}$, simply note that $|\phi_n^{jm} - 1| \leq 1$ and apply this inequality to the absolute value of (7.56). We establish that $B^{(k)}_{N_T,n} = \|\mathcal{E}_n^k\|_\infty$ by noticing that $B^{(k)}_{N_T,n}$ is exactly the absolute sum of the final row of index N_T from $\mathcal{E}_n^k$ by (7.55), and that the maximum absolute row sum is given by this row. □

7.3.4 Asymptotic final-point (R-)MGRIT approximate solutions

We now show that for any fixed k, both MGRIT and R-MGRIT converge to the same final-point approximate solution, asymptotically for $N_T \to +\infty$.

Theorem 3 (Common asymptotic limit of MGRIT and R-MGRIT approximate solutions)**.** *Suppose Φ and Ψ satisfy the simulatenous diagonalizability assumptions of subsection 7.3.3. Then, as $N_T \to +\infty$, the approximate modal solutions from (7.57) and (7.58) converge to the same limit, denoted by $v_{\infty,n}^{(k)} := \widetilde{V}_{\infty,n}^k u_{0,n}$, where the underlying scalar asymptotic limit operator is defined as*

$$\widetilde{V}_{\infty,n} := \frac{(\phi_n^m - \psi_n)\phi_n^m}{(1-\psi_n)} = -E_{\infty,n}^{(1)}. \tag{7.67}$$

Similarly, the full approximate solution vectors from (7.27) and (7.32) converge to the same limit as $N_T \to +\infty$, denoted by $\mathbf{v}_\infty^{(k)} := \widetilde{V}_\infty^k \mathbf{u}_0$, where the underlying asymptotic limit operator is defined as

$$\widetilde{V}_\infty := (I-\Psi)^{-1}(\Phi^m - \Psi)\Phi^m. \tag{7.68}$$

Proof. Substituting the asymptotic limits of $E_{N_T}^{(k)}$ derived in lemma 8 into the limits of the MGRIT and R-MGRIT operators, $V_{N_t,n}^{(k)}$ (7.57) and $\widetilde{V}_{N_t,n}^k$ (7.58), respectively:

$$\lim_{N_T\to+\infty} V_{N_t,n}^{(k)} = \lim_{N_T\to+\infty} (\phi_n^{N_t} - E_{N_T,n}^{(k)}) = -\lim_{N_T\to+\infty} E_{N_T,n}^{(k)} = \widetilde{V}_{\infty,n}^k, \tag{7.69}$$

$$\lim_{N_T\to+\infty} \widetilde{V}_{N_t,n}^k = \left(\lim_{N_T\to+\infty} (\phi_n^{N_t} - E_{N_T,n}^{(1)})\right)^k = -\lim_{N_T\to+\infty} (E_{N_T,n}^{(1)})^k = \widetilde{V}_{\infty,n}^k. \tag{7.70}$$

Then, taking the limit of (7.57) and (7.58) as $N_T \to +\infty$ yields the common limit

$$\lim_{N_T\to+\infty} v_{N_t,n}^{(k)} = \widetilde{V}_{\infty,n}^k u_{0,n} = \lim_{N_T\to+\infty} \widetilde{v}_{N_t,n}^{(k)}. \tag{7.71}$$

Expanding the initial guess $\mathbf{u}_0$ in terms of the orthonormal eigenvectors $\{\mathbf{q}_n\}_{n=1}^N$ (the columns of Q (4.98)), we get $\mathbf{u}_0 = \sum_{n=1}^N u_{0,n}\mathbf{q}_n$. Similarly, we expand the MGRIT and R-MGRIT approximate solutions as $\mathbf{v}_{N_t}^{(k)} = \sum_{n=1}^N v_{N_t,n}^{(k)}\mathbf{q}_n$ and $\widetilde{\mathbf{v}}_{N_t}^{(k)} = \sum_{n=1}^N \widetilde{v}_{N_t,n}^{(k)}\mathbf{q}_n$, respectively. Applying the asymptotic modal limits in (7.69) and (7.70) to the approximate solution vectors yields the common limit

$$\lim_{N_T\to+\infty} \mathbf{v}_{N_t}^{(k)} = \lim_{N_T\to+\infty} \widetilde{\mathbf{v}}_{N_t}^{(k)} = \sum_{n=1}^N \widetilde{V}_{\infty,n}^k u_{0,n}\mathbf{q}_n. \tag{7.72}$$

Recognizing that reconstructing the matrix operator from the scalar eigenvalues $\widetilde{V}_{\infty,n}$ and eigenvectors $\mathbf{q}_n$ is equivalent to replacing the eigenvalues ϕ_n and ψ_n with their corresponding commuting matrices Φ and Ψ, we identify the limit operator as

$$\widetilde{V}_\infty = Q\,\mathrm{diag}(\widetilde{V}_{1,\infty},\ldots,\widetilde{V}_{\infty,n})Q^* = (I-\Psi)^{-1}(\Phi^m-\Psi)\Phi^m. \tag{7.73}$$

Applying this limit operator k times to the initial guess yields $\mathbf{v}_\infty^{(k)} = \widetilde{V}_\infty^k \mathbf{u}_0$ as the common asymptotic limit after k iterations. □

While the modal results are relevant, for example, in the context of fact 43, we note that a simpler and more general proof for the vector-wise asymptotic limit can be obtained as an extension of remark 39.

Remark 40 (Simultaneous diagonalizability is not required for the asymptotic limit)**.** By hypothesis, $\rho(\Psi) < 1$. Therefore, the (Neumann) geometric series in (7.50) converges, such that

$$\sum_{i=0}^{j-2} \Psi^i = (I-\Psi^{j-1})(I-\Psi)^{-1}. \tag{7.74}$$

Note that $\lim_{j\to+\infty}\sum_{i=0}^{j-2}\Psi^i = (I-\Psi)^{-1}$. Then, taking the limit of $\mathbf{v}_{N_t}^{(k)} = \mathbf{v}_{N_T m}^{(k)}$ from (7.50) as $N_T \to +\infty$ matches the asymptotic limit from theorem 3:

$$\begin{aligned}\lim_{N_T\to+\infty} \widetilde{\mathbf{v}}_{N_T m}^{(k)} &= \left(I + (I-\Psi)^{-1}(\Phi^m - I)\right)\Phi^m\widetilde{\mathbf{u}}_0 \\ &= (I-\Psi)^{-1}(\Phi^m-\Psi)\Phi^m\widetilde{\mathbf{u}}_0 \coloneqq \widetilde{\mathbf{v}}^\infty.\end{aligned} \tag{7.75}$$

Therefore, the vector-wise asymptotic limit from theorem 3 holds even if Φ and Ψ are not simultaneously diagonalizable.

7.3.5 Finite-time bounds for (R)-MGRIT

By the triangle inequality, we can bound the finite-time difference between the MGRIT and R-MGRIT approximate solutions by their differences with respect to their shared asymptotic limit:

$$\|\mathbf{v}_{N_t}^{(k)} - \widetilde{\mathbf{v}}_{N_t}^{(k)}\| \le \|\mathbf{v}_{N_t}^{(k)} - \mathbf{v}_\infty^{(k)}\| + \|\widetilde{\mathbf{v}}_{N_t}^{(k)} - \mathbf{v}_\infty^{(k)}\|. \tag{7.76}$$

In our numerical experiments, we observe that the final-point approximate solutions of both MGRIT and R-MGRIT are close to their common asymptotic limit from theorem 3 of all $k \ge 1$, such that $\|\widetilde{\mathbf{v}}_{N_t}^{(k)} - \mathbf{v}_\infty^{(k)}\| \approx 0$ and $\|\mathbf{v}_{N_t}^{(k)} - \mathbf{v}_\infty^{(k)}\| \approx 0$. We rigorously bound the finite-time difference below.

Theorem 4 (Finite-time difference between MGRIT and R-MGRIT approximate solutions)**.** *For any given eigenmode n, the absolute difference between the finite-time modal approximate solutions produced by the MGRIT and R-MGRIT schemes at the final time point N_t satisfies*

$$|v_{N_t,n}^{(k)} - \widetilde{v}_{N_t,n}^{(k)}| \leq F_{N_T,n}^{(k)} |u_{0,n}|, \tag{7.77}$$

where the modal difference bound $F_{N_T,n}^{(k)}$ is defined as

$$F_{N_T,n}^{(k)} \coloneqq |\phi_n^{N_t}| + |E_{\infty,n}^{(k)} - E_{N_T,n}^{(k)}| + |\phi_n^{N_t}|^k + |E_{\infty,n}^{(k)} - (-E_{N_T,n}^{(1)})^k| + \sum_{i=1}^{k-1} \binom{k}{i} |\phi_n^{N_t}|^{k-i} |E_{N_T,n}^{(1)}|^i. \tag{7.78}$$

Furthermore, the difference between the full approximate solution vectors at the final time point is bounded by

$$\|\mathbf{v}_{N_t}^{(k)} - \widetilde{\mathbf{v}}_{N_t}^{(k)}\|_2 \leq \max_n \{F_{N_T,n}^{(k)}\} \|\mathbf{u}_0\|_2. \tag{7.79}$$

Proof. By adding and subtracting the common asymptotic modal limit operator $\widetilde{V}_{\infty,n}^k$ and applying the triangle inequality, the finite-time difference between the approximate solutions is bounded by their respective operator distances to this limit:

$$|v_{N_t,n}^{(k)} - \widetilde{v}_{N_t,n}^{(k)}| \leq |V_{N_t,n}^{(k)} - \widetilde{V}_{N_t,n}^k| |u_{0,n}| \leq \left(|V_{N_t,n}^{(k)} - \widetilde{V}_{\infty,n}^k| + |\widetilde{V}_{N_t,n}^k - \widetilde{V}_{\infty,n}^k| \right) |u_{0,n}|. \tag{7.80}$$

For the MGRIT operator, substituting (7.57) into the distance bound yields

$$|V_{N_t,n}^{(k)} - \widetilde{V}_{\infty,n}^k| \leq |\phi_n^{N_t}| + |E_{\infty,n}^{(k)} - E_{N_T,n}^{(k)}|. \tag{7.81}$$

For the R-MGRIT operator, expanding (7.58) using the binomial theorem gives

$$|\widetilde{V}_{N_t,n}^k - \widetilde{V}_{\infty,n}^k| \leq |\phi_n^{N_t}|^k + |E_{\infty,n}^{(k)} - (-E_{N_T,n}^{(1)})^k| + \left| \sum_{i=1}^{k-1} \binom{k}{i} (-1)^i (\phi_n^{N_t})^{k-i} (E_{N_T,n}^{(1)})^i \right|. \tag{7.82}$$

Applying the triangle inequality to the summation in (7.82) and combining this with (7.81) in (7.80) yields the total modal bound $F_{N_T,n}^{(k)} |u_{0,n}|$ as defined in (7.78).

To extend this modal result to the full approximate solution vectors at time N_t, we

sum over all N eigenmodes. Using Parseval's identity, we have

$$\begin{aligned}\|\mathbf{v}_{N_t}^{(k)} - \widetilde{\mathbf{v}}_{N_t}^{(k)}\|_2^2 &= \sum_{n=1}^{N} |v_{N_t,n}^{(k)} - \widetilde{v}_{N_t,n}^{(k)}|^2 \\ &\leq \sum_{n=1}^{N} (F_{N_T,n}^{(k)})^2 |u_{0,n}|^2 \\ &\leq \left(\max_n \{F_{N_T,n}^{(k)}\}\right)^2 \sum_{n=1}^{N} |u_{0,n}|^2 \\ &= \left(\max_n \{F_{N_T,n}^{(k)}\}\right)^2 \|\mathbf{u}_0\|_2^2.\end{aligned} \tag{7.83}$$

Taking the square root of both sides of (7.83) yields (7.79), completing the proof. □

Note that since $\lim_{N_T\to+\infty} |\phi_n^{N_t}| = 0$ and $\lim_{N_T\to+\infty} E_{N_T,n}^{(k)} = \lim_{N_T\to+\infty} (-E_{N_T,n}^{(1)})^k = E_{\infty,n}^{(k)}$ for all n and k, (7.78) implies that $\lim_{N_T\to+\infty} F_{N_T,n}^{(k)} = 0$ for all n and k, which recovers the asymptotic equivalence established in theorem 3. To conclude our finite-time comparison between MGRIT and R-MGRIT, we present an analogue to fact 43 which provides tight error bounds for k iterations of MGRIT and R-MGRIT given the constant initialization strategy discussed in remark 29. In particular, we note that the MGRIT and R-MGRIT C-point errors feature similar finite-time upper bounds which converge to the same asymptotic limit as $N_T \to +\infty$.

Theorem 5 (Error propagation bounds for k (R-)MGRIT iterations)**.** *If the MGRIT algorithm is initialized with a constant space-time initial guess trajectory $\mathbf{v}^{(0)} = B\mathbf{u}_0$, where B is defined in (7.33), then the global space-time error vectors at the C-points after k iterations for the MGRIT ($\mathbf{e}_c^{(k)}$) and R-MGRIT ($\widetilde{\mathbf{e}}_c^{(k)}$) schemes satisfy:*

$$\|\mathbf{e}_c^{(k)}\|_2 \leq \max_n \left\{ B_{N_T,n}^{(k)} C_{N_T,n} \right\} \|\mathbf{u}_0\|_2, \tag{7.84}$$

$$\|\widetilde{\mathbf{e}}_c^{(k)}\|_2 \leq \max_n \left\{ \widetilde{B}_{N_T,n}^{(k)} C_{N_T,n} \right\} \|\mathbf{u}_0\|_2, \tag{7.85}$$

where $C_{N_T,n} = \sum_{j=0}^{N_T} |\phi_n^{jm} - 1|^2$ encapsulates the initial C-point error, $B_{N_T,n}^{(k)}$ is defined in (7.66), and $\widetilde{B}_{N_T,n}^{(k)} := \|\mathcal{E}_n\|_\infty |\widetilde{V}_{N_T,n}|^{k-1}$, where $\|\mathcal{E}_n\|_\infty$ is given in (7.53). Furthermore, as $N_T \to +\infty$, both operator bounds converge to the same asymptotic limit, the kth power of (7.54):

$$\lim_{N_T\to+\infty} B_{N_T,n}^{(k)} = \lim_{N_T\to+\infty} \widetilde{B}_{N_T,n}^{(k)} = \left(\frac{|(\phi_n^m - \psi_n)||\phi_n^m|}{1 - |\psi_n|} \right)^k = \left(\lim_{N_T\to+\infty} \|\mathcal{E}_n\|_\infty \right)^k \tag{7.86}$$

Proof. The initial space-time error at the j-th C-point for the n-th mode is $e^0_{jm,n} = (\phi_n^{jm} - 1)u_{0,n}$. Taking the squared Euclidean norm of space-time initial C-point error vector yields (7.87):

$$\|\mathbf{e}^{(0)}_{c,n}\|_2^2 = \sum_{j=0}^{N_T} |\phi_n^{jm} - 1|^2 |u_{0,n}|^2 = C^2_{N_T,n}|u_{0,n}|^2. \tag{7.87}$$

After k MGRIT iterations, the C-point error is $\mathbf{e}_c^{(k)} = \mathcal{E}^k \mathbf{e}_c^{(0)}$. Expanding into orthogonal components, we bound the C-point error by

$$\|\mathbf{e}_c^{(k)}\|_2^2 = \sum_{n=1}^{N} \|\mathcal{E}^k \mathbf{e}^{(0)}_{c,n}\|_2^2 \leq \max_n \left(\|\mathcal{E}_n^k\|_2^2 C^2_{N_T,n}\right) \sum_{n=1}^{N} |u_{0,n}|^2. \tag{7.88}$$

Substituting $\|\mathcal{E}_n^k\|_2 \leq \|\mathcal{E}_n^k\|_\infty = B^{(k)}_{N_T,n}$ and $\|\mathbf{u}_0\|_2^2 = \sum_{n=1}^N |u_{0,n}|^2$ yields (7.84).

For the k-th R-MGRIT iteration, the algorithm computes one MGRIT iteration initialized by $\widetilde{\mathbf{u}}_0^{(k)} = \sum_{n=1}^N \widetilde{V}^{k-1}_{N_T,n} u_{0,n}\mathbf{q}_n$. Note that the expression for $|\widetilde{\mathbf{e}}^{(k)}_{c,n}|$ simply replaces $|u_{0,n}|^2$ in (7.87) with $|\widetilde{V}^{k-1}_{N_T,n}u_{0,n}|^2$. Since only one MGRIT iteration is performed for each restart, the error is bounded by $\|\mathcal{E}_n\|_\infty = B^{(1)}_{N_T,n}$:

$$\|\widetilde{\mathbf{e}}_c^{(k)}\|_2^2 \leq \max_n \left(\left(B^{(1)}_{N_T,n}\right)^2 |\widetilde{V}_{N_T,n}|^{2(k-1)} C^2_{N_T,n} \right) \sum_{n=1}^{N} |u_{0,n}|^2. \tag{7.89}$$

Substituting (7.53) into this expression yields (7.85).

We now evaluate the limits of both bounds as $N_T \to +\infty$. For the MGRIT bound, recognizing the summation in $B^{(k)}_{N_T,n}$ as the partial negative binomial expansion from the proof of lemma 8, we pass the limit to infinity to obtain

$$\lim_{N_T\to+\infty} B^{(k)}_{N_T,n} = \frac{(|(\phi_n^m - \psi_n)||\phi_n^m|)^k}{(1-|\psi_n|)^k}. \tag{7.90}$$

For the R-MGRIT bound, the modal approximate solution operator is $\widetilde{V}_{N_T,n} = \phi_n^{N_t} - E^{(1)}_{N_T,n}$. Applying the triangle inequality and noting that $|E^{(1)}_{N_T,n}| \leq B^{(1)}_{N_T,n}$ by lemma 9 we get

$$|\widetilde{V}_{N_T,n}| \leq |\phi_n|^{N_t} + B^{(1)}_{N_T,n}. \tag{7.91}$$

Since $\lim_{N_T\to+\infty} |\phi_n|^{N_t} = 0$, we have $\lim_{N_T\to+\infty} |\widetilde{V}_{N_T,n}| = \lim_{N_T\to+\infty} B^{(1)}_{N_T,n}$. Consequently, by (7.54) we arrive at

$$\lim_{N_T\to+\infty} \widetilde{B}^{(k)}_{N_T,n} = \lim_{N_T\to+\infty} \left(B^{(1)}_{N_T,n}|\widetilde{V}_{N_T,n}|^{k-1}\right) = \left(\frac{|(\phi_n^m - \psi_n)||\phi_n^m|}{1-|\psi_n|}\right)^k, \tag{7.92}$$

completing the proof. □

In particular, by theorem 5 we expect that if MGRIT converges linearly in the error or in the residual, as assumed in remark 15, R-MGRIT also converges with similar linear convergence rates if N_T is sufficiently large.

7.3.6 On a truncated R-MGRIT strategy

We have established both MGRIT and R-MGRIT converge to the same asymptotic final-point approximate solutions as the number of time points grows infinitely large. We now briefly remark how this supports a time-truncated R-MGRIT strategy, in which we consider taking a fixed fraction γ of the number of time points N_t required for the underlying optimization method to converge to the desired accuracy. We begin by presenting a corollary to lemma 7, which characterizes the R-MGRIT approximate solutions at all C-points under the constant initialization strategy described in (7.13).

Corollary 3 (R-MGRIT approximate solutions at C-points)**.** *Let $\widetilde{\mathbf{u}}^{(k)}$ denote the kth target (exact) trajectory $\widetilde{\mathbf{v}}^{(k)}$ denote the approximate kth trajectory after 1 MGRIT iteration on an all-at-once system of the form (7.7), where $\widetilde{\mathbf{u}}_0^{(k)}$ denotes the initial guess of the kth system. Then, the R-MGRIT approximate solutions generated by algorithm 8 at iteration $k = 1, 2 \ldots$ and C-point index $0 \leq j \leq N_T$ satisfy*

$$\widetilde{\mathbf{v}}_0^{(k)} = \widetilde{\mathbf{u}}_0^{(k)}, \qquad \widetilde{\mathbf{v}}_m^{(k)} = \Phi^m \widetilde{\mathbf{u}}_0^{(k)}, \tag{7.93}$$

$$\widetilde{\mathbf{v}}_{jm}^{(k)} = \left(I + \sum_{i=0}^{j-2} \Psi^i (\Phi^m - I) \right) \Phi^m \widetilde{\mathbf{u}}_0^{(k)}, \quad j = 2, \ldots, N_T. \tag{7.94}$$

Proof. Follows directly from (7.51), recognizing that each R-MGRIT restart behaves exactly like the first iteration of a standard MGRIT solve with updated initial guess $\widetilde{\mathbf{u}}_0$. □

By (7.68) and (7.94) we have that, for any fixed $j \geq 2$,

$$\|\widetilde{\mathbf{v}}_{jm}^{(k)} - \widetilde{\mathbf{v}}_{\infty}^{(k)}\| \leq \|\Psi^{j-1}\| \leq \|\Psi\|^{j-1}. \tag{7.95}$$

Therefore, for all sufficiently large j such that $\|\Psi\|^{j-1} \approx 0$, we have that $\widetilde{\mathbf{v}}_{jm}^{(k)} \approx \widetilde{\mathbf{v}}_{\infty}^{(k)}$. Hence, in practice, we only need to compute $\overline{N_T} < N_T$ coarse-grid correction steps for some sufficiently large $\overline{N_T}$ such that

$$\widetilde{\mathbf{v}}_{\overline{N_T}m}^{(k)} \approx \widetilde{\mathbf{v}}_{N_t}^{(k)} \approx \widetilde{\mathbf{v}}_{\infty}^{(k)}; \tag{7.96}$$

this is illustrated numerically, for example, in the middle panel of figure 7.5.

Thus, we propose an R-MGRIT strategy of using algorithm 8 with a truncated time domain, where we take $\overline{N_t} := \gamma N_t$ for the number of fine-grid time points and $\overline{N_T} := (\gamma N_t)/m = \gamma N_T$ for the number of C-points, where N_t here denotes the number of iterations required for convergence of the underlying optimization method (5.52). For any sufficiently large γ (e.g., $\gamma = 0.2$ in our numerical experiments), the R-MGRIT approximate solution at final time point of the γ-truncated time domain satisfies (7.96), such that we are able to obtain approximately the same gradient reduction with γ-truncated R-MGRIT as with a standard MGRIT solve.

7.4 Numerical results for MGRIT vs. R-MGRIT

For our numerical experiments, we consider two model problems from section 5.2: MP1, (5.15), and MP2, (5.23), which are special cases of (5.1) and (5.2), respectively. First, we solve the problem sequentially using GD, (5.30), or PG, (5.32), the former for MP1 and the latter for MP2. This provides us with N_t, exactly the number of iterations required for convergence within the given tolerance threshold, which we then use to specify the MGRIT fine-grid mesh of $i = 0, \ldots, N_t$ points, as illustrated in figure 4.1, with spacing $\Delta t = 1/L$. In practice, one can obtain reasonable estimates for N_t given theoretical or empirically observed converge rates of the underlying optimization method; see remark 16.

Next, we solve the same problem parallel-in-iteration with MGRIT in its standard (algorithms 6 and 7) and restarting (algorithm 8) variants. For the latter, based on remark 34 and subsection 7.3.6, we consider truncating the fine-grid time domain by a fixed factor $\gamma = 0.2$. In this case, we solve all-at-once systems of the form (7.7) corresponding to a fifth of the number of iterations of the underlying optimization method (either GD for MP1 or PG for MP2) as the all-at-once system of the form (4.3) solved by standard MGRIT, As in section 5.7, we illustrate convergence results for a particular set of choices for the parameters N (dimension of the Laplacian matrix), m (coarsening factor) and $\widehat{\ell}$ (number of levels). Later, we present results for different values of m, $\widehat{\ell}$ and N to analyze how MGRIT's performance is affected by the choices of these parameters.

We choose 10^{-8} as the tolerance criteria for all iterative methods. The initial MGRIT guess $\mathbf{u}^0 \in \mathbb{R}^N$ is chosen to be random with components in $[0, 1]$, after which the initial condition $\mathbf{u}_0 = \mathbf{w}_0$ of the all-at-once system (4.7) is enforced. For both model problems, we fix a step size $s^{(\ell)} = m^{\ell-1}/L$ for the ℓth level operator, $\ell = 1, \ldots, \widehat{\ell}$, where m denotes the coarsening factor and $\widehat{\ell}$ the maximum number of levels; note that we use the same operator

Ψ (with the level-appropriate step size $s^{(\ell)}$) on all levels $\ell \geq 2$. The MGRIT operators are the same as in section 5.7, namely the GD (5.30) and Proximal Point (PP) (5.31) operators for MP1 and the PG (5.32) and Alternating Proximals (AP) (5.35) operators for MP2; see subsections 5.5.1 and 5.5.2 for the relevant discussion.

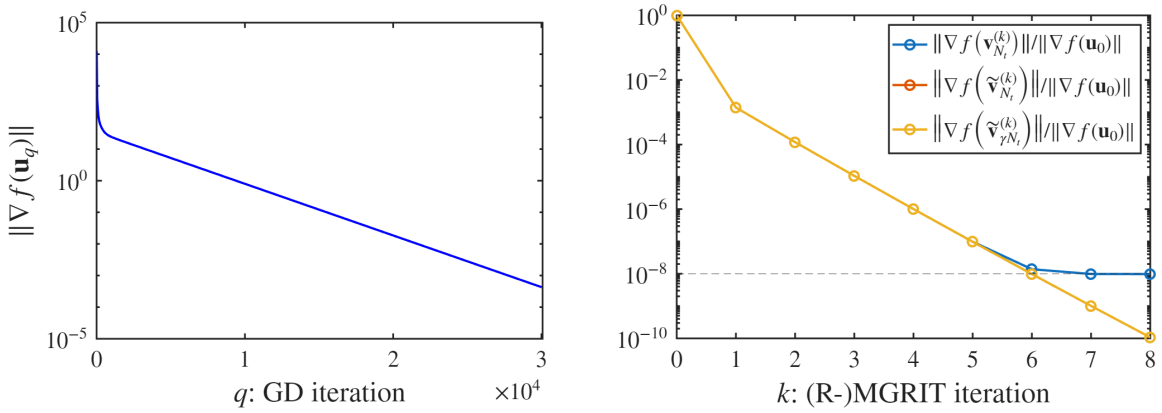
Remark 41 (Numerical results from chapter 5 vs. chapter 7)**.** Note that the numerical experiments in this chapter are setup almost identically to those in section 5.7, except that here we enforce the constant initialization strategy (7.13) and that we enforce the condition that the final time point N_t must be a C-point. Therefore, results for standard MGRIT solves are almost identical to those in section 5.7, with the difference being attributed to the fact that the C-points where initialized randomly in section 5.7, while here they are made to be identical to the initial guess of each all-at-once system solve. Due to the similarities between the experiments from chapters 5 and 7, in subsections 7.4.1 and 7.4.2 we focus on comparing the behaviour of the MGRIT and R-MGRIT algorithms applied to MP1, (5.15), and MP2, (5.23), respectively, and refer to subsections 5.7.1 and 5.7.2 for comments on the problems themselves.

7.4.1 Numerical experiments for gradient descent applied to MP1

We solve MP1, (5.15), using GD, (5.30). For this example, the 1D Laplacian matrix $A_1 \in \mathbb{R}^{N\times N}$ in (2.5) with 2-norm L is taken with $N = n = 80$, the coarsening factor is chosen to be $m = 64$ and the number of levels to be $\widehat{\ell} = 2$. We also choose $\mathbf{b} \in \mathbb{R}^N$ to be random with components in $[0, 1]$. First, we solve the problem sequentially, with $30,016$ iterations required for the convergence of the gradient descent algorithm (3.49). However, for $N_T = N_t/m$ to be an integer, in our MGRIT solves we take $N_t = 30,080$, the next multiple of 64. We then generate the MGRIT fine grid $t_i = i\Delta t$, $i = 0, \ldots, N_t$ with $\Delta t = 1/L$. Next, we generate the coarse-grid $T_j = j\Delta t$, $j = 0, 1, \ldots, N_T$, with $\Delta t = m\Delta t = 64/L$, such that the coarse grid has $N_T + 1 = 471$ points. MGRIT then solves (4.7) using algorithm 6. We then consider truncating the total number of fine-grid intervals from N_t to $\overline{N_t} = \gamma N_t$. For a truncation factor $\gamma = 0.2$, we work with $\overline{N_t} = 6,080$ (extending from $\gamma N_t = 6,016$ to the next multiple of $m = 64$) fine-grid intervals and $\overline{N_T} = 95$ coarse-grid intervals.

Convergence for the sequential and MGRIT solves is illustrated in figure 7.4, in which we observe that MGRIT requires 8 iterations for convergence, while R-MGRIT only requires 6 iterations. We present the gradient at the C-point approximate solutions of each iteration of MGRIT and R-MGRIT in figure 7.5, in which we observe that the truncated R-MGRIT possesses a similar convergence behaviour as the full-time domain R-MGRIT

solve. Therefore, by truncating N_t to a sufficiently large (yet relatively small) $\overline{N_t} = N_t/\gamma$, we obtain convergence in at most as many iterations as MGRIT.



(a) Sequential convergence **(b)** MGRIT convergence

Figure 7.4: **(a)**: Convergence of MP1, (5.15), with a standard sequential solve using GD method (3.49); $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. **(b)**: Convergence of MP1, (5.15), with MGRIT using GD (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\nabla f(\mathbf{v}_{N_t}^{(k)})$ and $\nabla f(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denote the gradient of the approximate solution at the final time point N_t and iteration k of MGRIT and R-MGRIT, respectively, $\nabla f(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point γN_t and iteration k of γ-truncated R-MGRIT for $\gamma = 0.2$, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

In figure 7.5, we do not observe faster convergence for R-MGRIT over MGRIT, with both schemes require 6 iterations to reduce the initial gradient by a factor of approximately 10^{-8}. In practice, however, we observe that R-MGRIT attains a slight improvement in the gradient reduction after 6 iterations. While the theory does not predict faster gradient reduction for R-MGRIT, we note from theorem 5 that R-MGRIT possesses slightly stricter error reduction bounds for finite N_T if $|\widetilde{V}_{N_t,n}| < 1$, with $\widetilde{V}_{N_t,n}$ defined in (7.58).

Although the numerical results illustrated in figure 7.5 are only presented for the norm of the gradient, the theory from section 7.3 allows us to interpret the results in terms of the approximate solutions. As predicted by theorem 4, the difference between the MGRIT and R-MGRIT final-point approximate solutions for each fixed iteration index $1 \leq k \leq 6$ is very small. Moreover, we observe that (7.96) holds for $\overline{N_T} = \gamma N_T$ with $\gamma = 0.2$; by figure 7.5, with γ-truncated R-MGRIT attaining a similar gradient reduction factor to MGRIT and full-time R-MGRIT.

For iterations $k \geq 7$, we observe that N_T is no longer large enough for the $\mathbf{v}_{N_t}^{(k)} \approx \mathbf{v}_{\infty}^{(k)}$

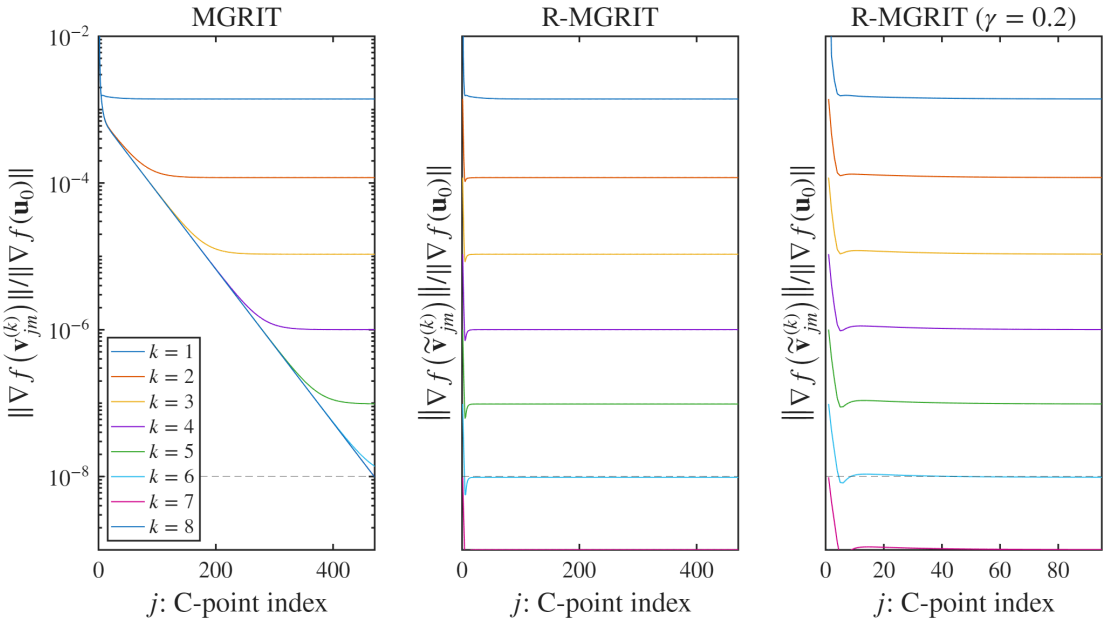


Figure 7.5: 2-norm of the gradient of the MP1, (5.15), approximate solutions at coarse time index j and iteration k of MGRIT, R-MGRIT and γ-truncated R-MGRIT (with $\gamma = 0.2$), from left to right, divided by the 2-norm of the gradient of the initial guess $\mathbf{u}_0$; $\nabla f(\mathbf{v}_{jm}^{(k)})$ and $\nabla f(\widetilde{\mathbf{v}}_{jm}^{(k)})$ denote the gradient of the approximate solution at the C-point jm for $j = 0, 1, \ldots, N_T$ (restricted to $j = 0, 1, \ldots, \overline{N_T}$ for truncated R-MGRIT) and iteration k of MGRIT and R-MGRIT, respectively, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

approximation to hold and the MGRIT final-point gradient stalls, with a final gradient reduction factor of approximately 10^{-8} as predicted by lemma 6. By approximating new target trajectories rather than the initial one, R-MGRIT iterations of index $k \geq 7$ continue to reduce the gradient by approximately the same linear rate, as first illustrated in figure 7.4. Thus, as discussed in remark 34, R-MGRIT does not require an accurate estimation of N_t, the number of iterations required for convergence of GD (5.30) to a minimizer of MP1, (5.15), with the prescribed 10^{-8} accuracy threshold.

In table 7.1, we present MGRIT and R-MGRIT convergence results for MP1 (5.15) with different coarsening factors m, number of MGRIT levels $\widehat{\ell}$ and different sizes for the matrix A_1 in (2.5); we note that all R-MGRIT results hold identically for a truncation factor $\gamma = 0.2$. Having established in table 5.1 that the number of iterations does not change much as $\widehat{\ell}$ increases, here we only present results for the smallest ($\widehat{\ell} = 2$) and largest ($\widehat{\ell} = \ell$) possible number of levels given the number of time points N_t and the

coarsening factor m.

Table 7.1: MP1 (5.15) MGRIT convergence results for with different values of m and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid. In parentheses, results for R-MGRIT. Here, ℓ denotes the maximum number of levels for the given coarsening factor. For both $n = 60$ and $n = 80$, we have $\ell = 8$ for $m = 4$, $\ell = 4$ for $m = 16$, $\ell = 3$ for $m = 64$ and $\ell = 2$ for $m = 256$ and $m = 1024$.

	$\widehat{\ell} = 2$	$\widehat{\ell} = \ell$
$m = 4$	9(9)	9(9)
$m = 16$	8(7)	8(7)
$m = 64$	8(6)	8(6)
$m = 256$	7(5)	

(a) $N = 60$, $N_t = 18,050$

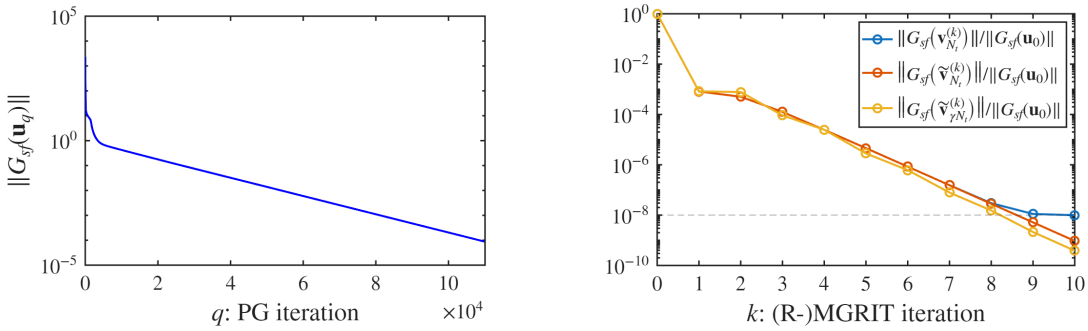
	$\widehat{\ell} = 2$	$\widehat{\ell} = \ell$
$m = 4$	9(9)	9(9)
$m = 16$	8(7)	8(7)
$m = 64$	8(6)	8(6)
$m = 256$	8(6)	
$m = 1024$	7(5)	

(b) $N = 80$, $N_t = 30,016$

7.4.2 Numerical experiments for proximal gradient applied to MP2

Numerical results for MP2-1D

We now consider solving the elastic obstacle problem (EOP) problem, MP2 (5.23), using PG method (5.32). For MGRIT, we choose the AP operator (5.35) to be the coarse-grid operator. We present one MP2-1D and one MP2-2D example, similar to those in subsections 5.7.1 and 5.7.2. For the MP2-1D example, we consider MP2 in (5.23) with $d = 1$ and $N = 256$, with MGRIT parameters $m = 256$ and $\widehat{\ell} = 2$. We again start by presenting results for the sequential and (R-)MGRIT convergence in figure 7.6. In this example, the PG method requires $109,888$ iterations to reach the prescribed halting tolerance, i.e., that the generalized gradient of F, G_{sF} in (5.145), be reduced by 8 orders of magnitude; we extend the number of time points to $N_t = 110,081$ such that $N_T = N_t/m = 431$ is an integer. In figure 7.6, we observe that MGRIT reaches the prescribed residual tolerance in 10 iterations, while R-MGRIT reaches it in 9 iterations both when using the full N_t number of time points and when truncating the time domain to a fifth of the number of time points. We present the C-point gradients of each iteration in figure 7.7.



(a) Sequential convergence

(b) MGRIT/ convergence

Figure 7.6: **(a)**: Convergence of MP2-1D, (5.23), with a standard sequential solve using PG method (5.32); $G_{sf}(\mathbf{u}_q)$ denotes the generalized gradient (5.145) of the qth sequential iterate. **(b)**: Convergence of MP2-1D, (5.23), with MGRIT using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $G_{sF}(\mathbf{v}_{N_t}^{(k)})$ and $G_{sF}(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denote the gradient of the approximate solution at the final time point N_t and iteration k of MGRIT and R-MGRIT, respectively, $G_{sF}(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point γN_t and iteration k of γ-truncated R-MGRIT for $\gamma = 0.2$, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

Numerical results for MP2-2D

For the MP2-2D example, we consider MP2 in (5.23) with $d = 2$ and $N^2 = 64^2$, with MGRIT parameters $m = 64$ and $\widehat{\ell} = 2$. We present the sequential and MGRIT convergence results in figure 7.8, noting that the sequential solver required $13,256$ iterations to converge; for a coarsening factor $m = 64$, we then work with $N_t = 13,312$, such that $N_T = N_t/m = 208$ denotes the number of coarse time intervals. We present C-point gradients in figure 7.9, in which we observe that γ-truncated R-MGRIT (with $\gamma = 0.2$) converged in 9 iterations, while MGRIT and full-time R-MGRIT converged in 10 iterations. As in the MP2-1D example, we observe that γ-truncated R-MGRIT required fewer iterations to reduce the gradient by a factor of 10^{-8} than MGRIT and full-time R-MGRIT.

While the theory from section 7.3 is only presented for linear MGRIT problems, we observe generally similar results for the nonlinear MP2, (5.23), in both one and two dimensions. In particular, in both figures 7.6 and 7.8 we observe that R-MGRIT reduces the gradient by a factor of 10^{-8} in the same 9 iterations required for MGRIT to achieve a similar reduction factor, with R-MGRIT continuing to reduce the gradient in subsequent iterations while MGRIT's final-point generalized gradient stalls. Although the approximation from (7.96) for $\overline{N_T} = \gamma N_T$ with $\gamma = 0.2$ is observed to be less accurate in figures 7.7

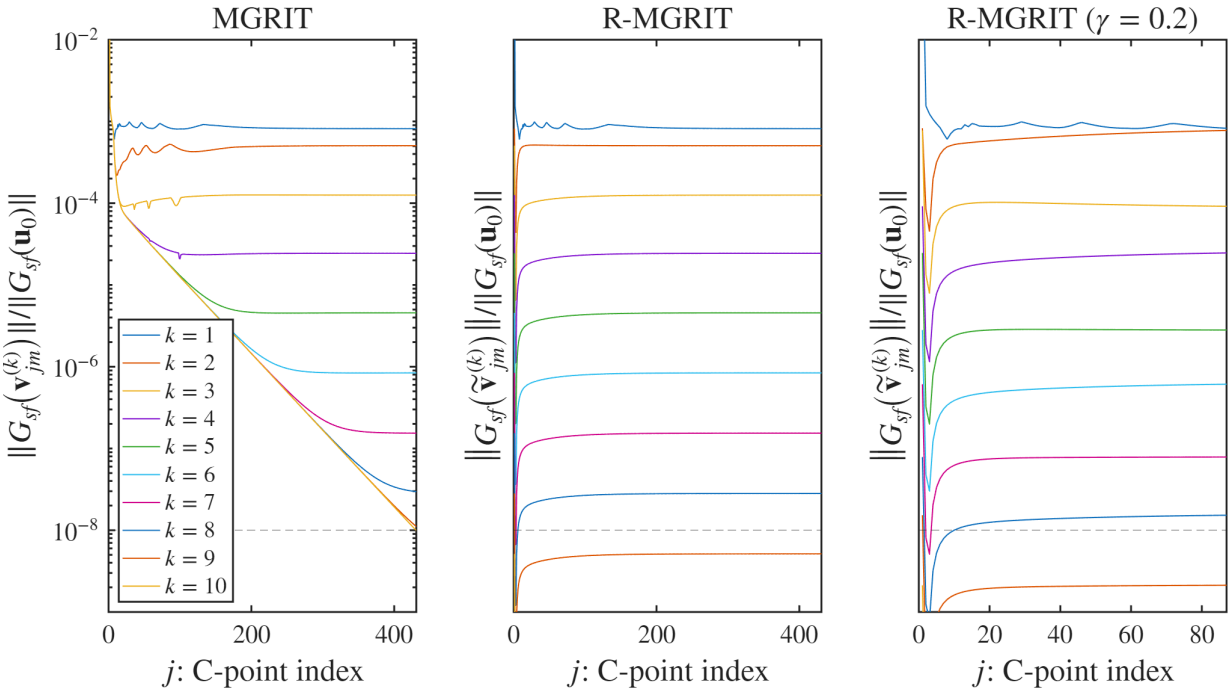
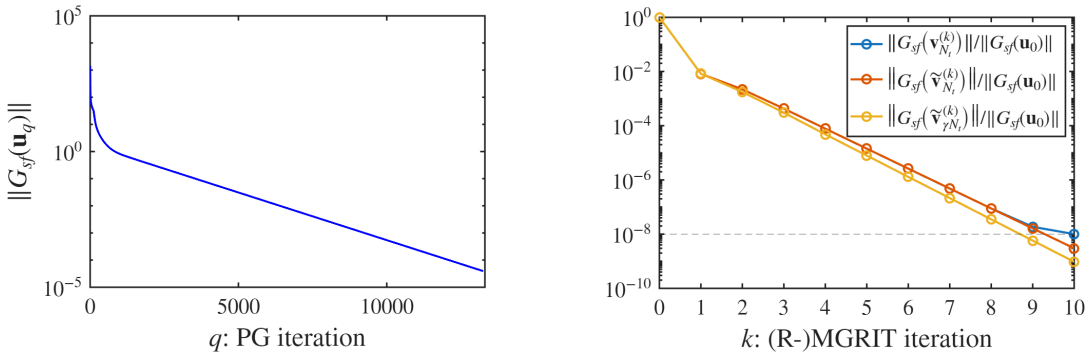


Figure 7.7: 2-norm of the generalized gradient G_{sF} (5.145) of the MP2-1D, (5.23), approximate solutions at coarse time index j and iteration k of MGRIT, R-MGRIT and γ-truncated R-MGRIT (with $\gamma = 0.2$), from left to right, respectively, divided by the 2-norm of the gradient of the initial guess $\mathbf{u}_0$; $G_{sF}(\mathbf{v}_{jm}^{(k)})$ and $G_{sF}(\widetilde{\mathbf{v}}_{jm}^{(k)})$ denote the generalized gradient of the approximate solution at the C-point jm for $j = 0, 1, \ldots, N_T$ (restricted to $j = 0, 1, \ldots, \overline{N_T}$ for truncated R-MGRIT) and iteration k of MGRIT and R-MGRIT, respectively, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

and 7.9 than in figure 7.5, as can be inferred from the rightmost panel of the respective figures, this is advantageous for γ-truncated R-MGRIT, which is able to reduce the final-point generalized gradient (5.145) at a slightly faster rate.

In tables 7.2 and 7.3, we present convergence results for MP2, (5.23), with different coarsening factors m, number of MGRIT levels $\widehat{\ell}$ and different sizes for the matrices A_1 in (2.5) for MP2-1D and A_2 in (2.9) for MP2-2D. As in table 7.1, we only present results for the smallest ($\widehat{\ell} = 2$) and largest ($\widehat{\ell} = \ell$) permissible number of levels. Comparing results for MGRIT, R-MGRIT with full time domain in parentheses and γ-truncated R-MGRIT (with a truncation factor $\gamma = 0.2$), we note that for all pairs of m and $\widehat{\ell}$, γ-truncated R-MGRIT required the fewest iterations for convergence.



(a) Sequential convergence

(b) MGRIT convergence

Figure 7.8: **(a)**: Convergence of MP2-2D, (5.23), with a standard sequential solve using PG method (3.84); $G_{sf}(\mathbf{u}_q)$ denotes the generalized gradient (5.145) of the qth sequential iterate. **(b)**: Convergence of MP2-2D, (5.23), with MGRIT using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $G_{sF}(\mathbf{v}_{N_t}^{(k)})$ and $G_{sF}(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denote the gradient of the approximate solution at the final time point N_t and iteration k of MGRIT and R-MGRIT, respectively, $G_{sF}(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point γN_t and iteration k of γ-truncated R-MGRIT for $\gamma = 0.2$, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

Table 7.2: MP2-1D MGRIT convergence results for different values of n, m and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. In parentheses, results for R-MGRIT; in square brackets, results for R-MGRIT with truncation factor $\gamma = 0.2$. For $m = 4$, we have $\ell = 8$ for $n = 256$ and $\ell = 10$ for $n = 512$. For both $n = 256$ and $n = 512$, we have $\ell = 5$ for $m = 16$, $\ell = 4$ for $m = 64$, $\ell = 3$ for $m = 256$ and $\ell = 2$ for $m = 1024$.

	$\widehat{\ell} = 2$	$\widehat{\ell} = \ell$
$m = 4$	10(9)[10]	12(13)[11]
$m = 16$	9(8)(8)	10(9)[9]
$m = 64$	10(9)[9]	10(11)[9]
$m = 256$	10(9)[9]	10(9)[9]
$m = 1024$	11(10)[9]	

(a) $N = 256$, $N_t = 109,888$.

	$\widehat{\ell} = 2$	$\widehat{\ell} = \ell$
$m = 4$	10(8)[8]	12(9)[10]
$m = 16$	11(9)[9]	11(10)[9]
$m = 64$	9(7)[9]	10(9)[9]
$m = 256$	9(8)[7]	
$m = 1024$	10(9)[9]	

(b) $N = 512$, $N_t = 369,612$.

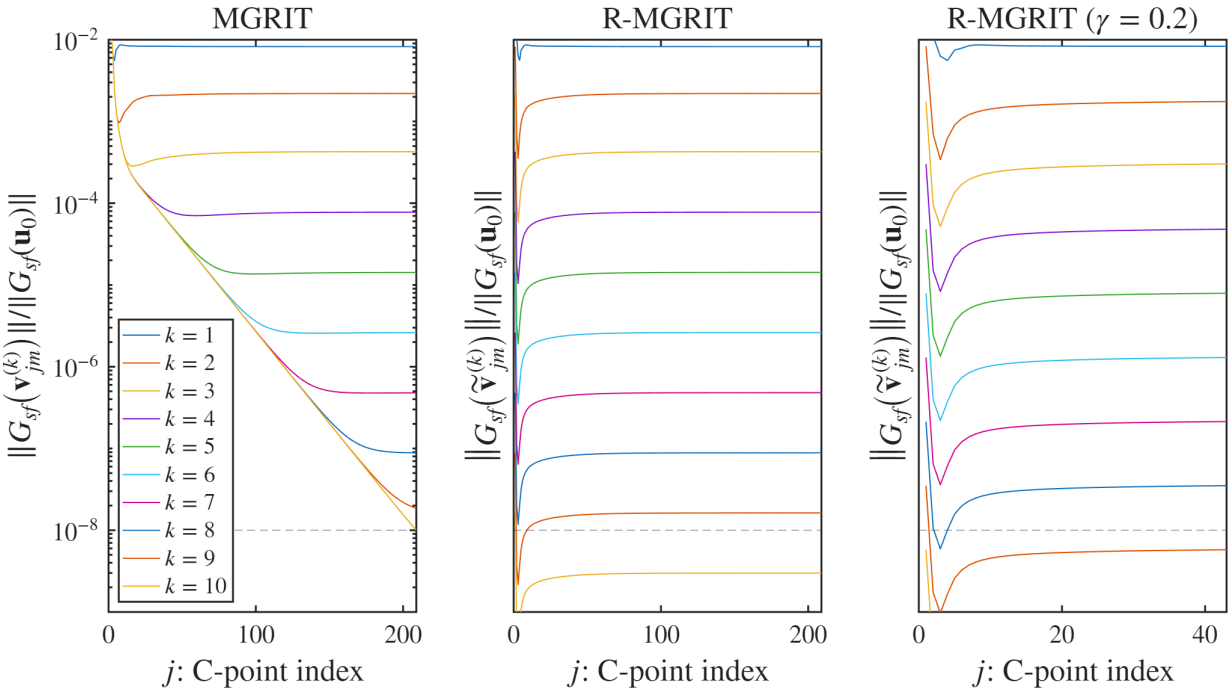


Figure 7.9: 2-norm of the generalized gradient G_{sF} (5.145) of the MP2-2D, (5.23), approximate solutions at coarse time index j and iteration k of MGRIT, R-MGRIT and γ-truncated R-MGRIT (with $\gamma = 0.2$), from left to right, respectively, divided by the 2-norm of the gradient of the initial guess $\mathbf{u}_0$; $G_{sF}(\mathbf{v}_{jm}^{(k)})$ and $G_{sF}(\widetilde{\mathbf{v}}_{jm}^{(k)})$ denote the generalized gradient of the approximate solution at the C-point jm for $j = 0, 1, \ldots, N_T$ (restricted to $j = 0, 1, \ldots, \overline{N_T}$ for truncated R-MGRIT) and iteration k of MGRIT and R-MGRIT, respectively, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

7.5 R-MGRIT parallel speedup estimates

In section 7.4, we empirically observed that γ-truncated R-MGRIT achieves gradient reduction comparable to MGRIT (in the same or fewer number of iterations) with the R-MGRIT time domain truncated to a fifth of the number of time points required for MGRIT convergence to the same accuracy threshold. Because the primary motivation for employing MGRIT is to reduce the wall-clock time required to solve the sequential problem, we now formalize how this temporal truncation translates into practical parallel speedup. In the following, we assume that one R-MGRIT iteration has the same computational cost as one MGRIT iteration as described by the speedup formulas derived in section 5.8; that is, we assume that the additional wall-clock time required for initializing each each R-MGRIT

Table 7.3: MP2-2D MGRIT convergence results for different values of n, m and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. In parentheses, results for R-MGRIT; in square brackets, results for R-MGRIT with truncation factor $\gamma = 0.2$. For both $n^2 = 64^2$ and $n^2 = 128^2$, we have $\ell = 7$ for $m = 4$, $\ell = 4$ for $m = 16$, $\ell = 3$ for $m = 64$ and $\ell = 2$ for $m = 256$.

	$\widehat{\ell} = 2$	$\widehat{\ell} = \ell$
$m = 4$	10(10)[10]	14(13)[13]
$m = 16$	11(10)[10]	12(12)[11]
$m = 64$	10(10)[9]	10(10)[9]
$m = 256$	9(9)[7]	

(a) $N^2 = 64^2$, $N_t = 13,256$.

	$\widehat{\ell} = 2$	$\widehat{\ell} = \ell$
$m = 4$	9(9)[9]	11(11)[10]
$m = 16$	9(8)[8]	9(9)[9]
$m = 64$	9(8)[8]	9(9)[8]
$m = 256$	9(9)[8]	

(b) $N^2 = 128^2$, $N_t = 46,431$.

all-at-once system of the form (7.7), as well as any other additional operations, is negligible relative to the time required for the relaxation and coarse-grid correction steps. Thus, in this section we extend our work from section 5.8 to 2- and 3-level R-MGRIT with a γ-truncated temporal domain, with $\gamma = 1$ denoting the non-truncated R-MGRIT.

7.5.1 2-level γ-truncated R-MGRIT parallel speedup

Recall the notation and discussion from subsection 5.8.1. Now, we consider enforcing a fixed artificial truncation on the number of coarse-grid intervals, $\overline{N_c} < N_c$, for which the (generalized) gradient at the $\overline{N_c}$th C-point is approximately the same as that at the N_cth C-point. For a fixed $\overline{N_c}$ for all N_{it} iterations, for full parallelization we work with $\overline{N_p} = \overline{N_c}$ and the number of fine-grid intervals is $\overline{N_f} = m\overline{N_c}$. Thus, from (5.162) with $\alpha := t_c/t_f$, the speedup formula for this truncated time domain is

$$\overline{S}_2(m) = \frac{N_f}{\overline{N_{it}}\left(\alpha\overline{N_c} + 2m\right)}, \tag{7.97}$$

where $\overline{N_{it}}$ is the number of iterations required for R-MGRIT convergence on a fine grid with $\overline{N_f} = \overline{N_c}m$ intervals. Given some $\gamma \in (0, 1)$, we substitute $\overline{N_c} = \gamma N_f/m = \gamma N_c$ into (7.97), which yields

$$\overline{S_2}(m, \gamma) = \frac{N_f}{\overline{N_{it}}\left(\frac{\gamma\alpha N_f}{m} + 2m\right)} = \frac{1}{\overline{N_{it}}\left(\frac{\gamma\alpha}{m} + \frac{2m}{N_f}\right)}. \tag{7.98}$$

Note that if $\overline{N_{it}} = N_{it}$, this formula is identical to the MGRIT speedup formula (5.162), but with the operator cost ratio α scaled down by γ. By setting the derivative of (7.98) for a fixed γ with respect to m to zero, we find the optimal coarsening factor for the γ-truncated R-MGRIT solve:

$$\frac{\partial \overline{S_2}}{\partial m}(m_\gamma^*, \gamma) = 0 \iff -\frac{\gamma\alpha}{(m_\gamma^*)^2} + \frac{2}{N_f} = 0 \iff m_\gamma^* = \sqrt{\frac{\gamma\alpha N_f}{2}}. \tag{7.99}$$

We note that taking $\gamma = 1$ recovers the standard MGRIT optimal coarsening factor from (5.163), $m^* := m_1^*$. Then, the relation between the truncated R-MGRIT and the MGRIT optimal coarsening factors satisfies

$$\frac{m_\gamma^*}{m^*} = \sqrt{\gamma}. \tag{7.100}$$

By (7.99), we substitute $\frac{\gamma\alpha}{m_\gamma^*} = \frac{2m_\gamma^*}{N_f}$ into (7.98), which yields the optimal speedup

$$\overline{S_2}(m_\gamma^*, \gamma) = \frac{m_\gamma^*}{2\gamma\alpha\overline{N_{it}}} = \frac{\sqrt{N_f}}{\overline{N_{it}}2\sqrt{2\gamma\alpha}}, \tag{7.101}$$

such that the ratio between the γ-truncated R-MGRIT speedup and the standard MGRIT speedup from (5.164) (or equivalently, (7.101) with $\gamma = 1$ and N_{it} iterations) is given by

$$\frac{\overline{S_2}(m_\gamma^*, \gamma)}{S_2(m^*)} = \frac{1}{\sqrt{\gamma}}(N_{it}/\overline{N_{it}}). \tag{7.102}$$

In practice, we expect that $\overline{N_{it}} \leq N_{it}$ for many small γ, as seen in tables 7.1 to 7.3 for $\gamma = 0.2$. Therefore, the speedup is increased by a factor of $N_{it}/(\overline{N_{it}}\sqrt{\gamma})$ while the number of processors required for full parallelization is also reduced proportionally to γ:

$$\overline{N_p^*}/N_p^* = \frac{\gamma N_f/m_\gamma^*}{N_f/m^*} = \sqrt{\gamma}. \tag{7.103}$$

For example, if $\gamma = 1/4$ and $\overline{N_{it}} = N_{it}$, we achieve twice the speedup with half of the number of processors required for the standard MGRIT solve.

7.5.2 3-level γ-truncated R-MGRIT parallel speedup

We now extend the previous 2-level discussion for 3-level methods based on the 3-level speedup estimates from subsection 5.8.2. Similarly to the 2-level case, given some $\gamma \in (0, 1)$

we substitute N_c and N_{it} in (5.166) with $\overline{N_c} = \gamma N_f/m = \gamma N_c$ and $\overline{N_{it}}$, respectively. This yields the full-parallelization 3-level γ-truncated speedup

$$\overline{S_3}(m,\gamma) = \frac{1}{\overline{N_{it}}\left(\frac{\gamma\alpha}{m^2} + \frac{2m}{N_f}\left(1+\frac{3\alpha}{2}\right)\right)}. \tag{7.104}$$

Taking the derivative with respect to m and setting it equal to zero then yields the optimal coarsening factor

$$\frac{\partial \overline{S_3}}{\partial m}(m^*_\gamma,\gamma) = 0 \iff -\frac{2\gamma\alpha}{(m^*_\gamma)^3} + \frac{2}{N_f}\left(1+\frac{3\alpha}{2}\right) = 0 \iff m^*_\gamma = \sqrt[3]{\frac{\gamma N_f}{1/\alpha + 3/2}}. \tag{7.105}$$

Once again letting $m^* \coloneqq m^*_1$ denote the standard MGRIT optimal coarsening factor, we get that the γ-truncated R-MGRIT optimal coarsening factor satisfies

$$\frac{m^*_\gamma}{m^*} = \sqrt[3]{\gamma}. \tag{7.106}$$

Evaluating the truncated 3-level speedup (7.104) at its optimal coarsening factor $m^*_\gamma = \sqrt[3]{\gamma}m^*$, factoring out γ and substituting $\frac{2\alpha}{(m^*)^2} = \frac{2m^*}{N_f}\left(1+\frac{3\alpha}{2}\right)$ by (7.105) yields

$$\overline{S_3}(m^*_\gamma,\gamma) = \frac{1}{\overline{N_{it}}\left(\frac{\gamma\alpha}{(\sqrt[3]{\gamma}m^*)^2} + \frac{2(\sqrt[3]{\gamma}m^*)}{N_f}\left(1+\frac{3\alpha}{2}\right)\right)} = \frac{1}{\overline{N_{it}}\gamma^{1/3}\left(\frac{\alpha}{(m^*)^2} + \frac{2\alpha}{(m^*)^2}\right)}. \tag{7.107}$$

Rearranging the fraction and substituting (7.105) then yields

$$\overline{S_3}(m^*_\gamma,\gamma) = \frac{(m^*)^2}{\overline{N_{it}}3\alpha\gamma^{1/3}} = \frac{\left(m^*_\gamma\right)^2}{\overline{N_{it}}3\alpha\gamma} = \frac{N_f^{2/3}}{3\alpha\overline{N_{it}}\gamma^{1/3}(1/\alpha+3/2)^{2/3}}. \tag{7.108}$$

Finally, taking the ratio of the γ-truncated R-MGRIT optimal speedup (7.108) to the MGRIT optimal speedup (i.e., with $\gamma = 1$ and N_{it} iterations) yields

$$\frac{\overline{S_3}(m^*_\gamma,\gamma)}{S_3(m^*)} = \frac{1}{\sqrt[3]{\gamma}}(N_{it}/\overline{N_{it}}). \tag{7.109}$$

Similarly, it follows that the number of processors required for full-parallelization is reduced by an even sharper factor than in the 2-level case:

$$\overline{N^*_p}/N^*_p = \frac{\gamma N_f/m^*_\gamma}{N_f/m^*} = \gamma\frac{m^*}{m^*_\gamma} = \frac{\gamma}{\sqrt[3]{\gamma}} = \gamma^{2/3}. \tag{7.110}$$

For example, if we truncate the time domain by a factor $\gamma = 1/8$, we only require $(0.125)^{2/3} = 1/4$ of the number of processors compared to the 3-level standard MGRIT solve, while obtaining $(N_{it}/\overline{N_{it}})/(0.125)^{1/3} = 2N_{it}/\overline{N_{it}}$ times the speedup.

7.5.3 R-MGRIT parallel speedup estimates for MP1 and MP2

Similarly to subsection 5.8.3, we now present the computed R-MGRIT parallel speedup estimates for MP1, (5.15), and MP2, (5.23), in the following tables. First, we compute the optimal m^* for $\ell = 2$ and $\ell = 3$ by the formulas derived in subsections 7.5.1 and 7.5.2,respectively, given an empirically estimated $\alpha = \frac{t_c}{t_f}$. We choose $m = m^*$ as the coarsening factor, rounding it to the nearest integer if needed, for the MGRIT solves. By running the MGRIT algorithm of our choice, namely algorithm 6 for MP1 and algorithm 7 for MP2, with $m = m^*$ as the coarsening factor, we empirically obtain N_{it}^* as the number of iterations required for convergence with the theoretically predicted optimal coarsening factors. We then use $m = m^*$ and $N_{it} = N_{it}^*$ in the speedup formulas (7.98) and (7.104). We then do the same for truncated R-MGRIT, for which we consider a truncation factor $\gamma = 0.2$, as in section 5.7, as well as $\gamma = 0.125$, which exemplifies a more extreme truncation strategy.

Remark 42 (MGRIT parallel speedup estimates from chapter 5 vs. chapter 7)**.** We note that results for MGRIT are identical to those in subsection 5.8.3 except for the number of processor N_p, since in the numerical experiments from section 5.7 we did not enforce the condition that the final time point must be a C-point; see remark 31. In particular, in tables 7.4 to 7.6 we note that N_t denotes the number of iterations required for sequential convergence, which matches the values from section 5.7. However, as explained in section 7.4, in practice we extend this number to the next multiple of m to ensure that the final time point is a C-point. Therefore, the actual number of fine-grid points of each solve is possibly slightly larger than the stated value of N_t.

In tables 7.4 to 7.6, we observe that truncating the time domain to a fifth ($\gamma = 0.2$) or an eighth ($\gamma = 0.125$) of the number of time points required for standard MGRIT convergence provides a substantial improvement of the standard MGRIT speedup, in accordance with the theory developed in section 7.5. We recall that we obtain N_t by solving the problem sequentially, and that in a practical setting N_t would have to be estimated based on known convergence properties of the underlying optimization method; see remark 16. However, our results here suggest that R-MGRIT converges in less parallel wall-clock time by employing only a small fraction of N_t, which in turn sidesteps the need to estimate N_t with great accuracy; see remark 34.

We note that truncating more aggressively, i.e., choosing $\gamma = 0.125$ instead of $\gamma = 0.25$, did not significantly change the predicted speedup. In preliminary numerical experiments for both MP1, (5.15), and MP2, (5.23), omitted here for the sake of brevity, we observed that by truncating the time domain with some smaller factor $\gamma < 0.125$, the number of (R-)MGRIT iterations required for converge increases, which often offsets the decreased

Table 7.4: Speedup estimates for MP1, (5.15), different values of n and $\widehat{\ell}$, using GD operator (5.30) on the fine grid and PP operator (5.31) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 3.1$ for $N = 60$, and $\alpha = 3.5$ for $N = 80$. N_p denotes the number of processors required to achieve optimal speedup.

MGRIT

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	167	8	3.37	109
3	21	8	5.92	861

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	229	8	4.09	132
3	26	8	8.04	1431

R-MGRIT ($\gamma = 0.2$)

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	74	6	16.13	50
3	13	6	15.14	279

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	103	6	15.17	60
3	15	8	13.39	252

R-MGRIT ($\gamma = 0.125$)

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	59	8	15.29	40
3	11	8	13.01	207

(a) $N = 60$, $N_t = 18,050$.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	81	8	14.39	48
3	13	8	16.09	290

(b) $N = 80$, $N_t = 30,018$.

Table 7.5: Speedup estimates MP2-1D, (5.23), for different values of n and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 2.4$ for $N = 256$, and $\alpha = 3.9$ for $N = 512$. N_p denotes the number of processors required to achieve optimal speedup.

MGRIT

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	363	10	7.56	303
3	39	10	21.12	2820

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	849	10	10.88	436
3	59	9	33.05	6267

R-MGRIT ($\gamma = 0.2$)

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	162	9	12.21	137
3	23	10	36.73	957

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	380	7	34.79	196
3	34	8	61.75	2176

R-MGRIT ($\gamma = 0.125$)

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	128	11	12.64	109
3	19	10	40.11	724

(a) $N = 256$, $N_t = 109,888$.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	300	12	16.02	156
3	30	9	42.73	1542

(b) $N = 512$, $N_t = 369,612$.

Table 7.6: Speedup estimates for MP2-2D, (5.23), for different values of n and $\widehat{\ell}$, using PG operator (5.32) on the fine grid and AP operator (5.35) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated $\frac{t_c}{t_f}$ values given by $\alpha = 13$ for $N^2 = 64^2$, and $\alpha = 7$ for $N^2 = 128^2$. N_p denotes the number of processors required to achieve optimal speedup.

MGRIT

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	294	9	1.25	46
3	20	10	1.02	664

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	403	9	3.19	116
3	30	9	4.76	1549

R-MGRIT ($\gamma = 0.2$)

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	131	8	3.14	22
3	12	11	1.67	222

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	180	8	8.05	53
3	18	9	8.57	517

R-MGRIT ($\gamma = 0.125$)

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	104	10	3.2	17
3	10	11	1.86	167

(a) $N^2 = 64^2$, $N_t = 13,256$.

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	143	11	7.42	42
3	15	10	8.57	195

(b) $N^2 = 128^2$, $N_t = 46,431$.

computational cost of each iteration and leads to reduced speedup relative to $\gamma = 0.125$ or $\gamma = 0.2$. Furthermore, we observed that taking $0.2 < \gamma < 1$ did not reduce $\overline{N_{it}}$ relative to $\gamma = 0.2$. Note that an optimal truncation factor γ can not be determined *a priori* since we cannot estimate N_{it}^* as a function of γ and the other parameters. However, our numerical results suggest $\gamma = 0.2$ is a good problem-independent choice.

7.6 On a greedy restart strategy

In remark 33, we noted that different strategies for finding the initial guess other than picking the approximate solution at the final time point may be considered. In particular, we now consider choosing the new initial guess to be the approximate solution at the final time point with the smallest gradient over all time points of the most recent R-MGRIT iteration. We denote this variant of algorithm 8 by Greedy Restarted MGRIT (GR-MGRIT), which we define in algorithm 9.

Note that step 4 now requires us to search the gradient at all time points in order to find the time point with the smallest gradient, which is then taken to be the new initial guess. In practice, in the absence of a post-relaxation step (see remark 31), we may restrict this

Algorithm 9 GR-MGRIT($\Phi, \Psi, \mathbf{u}_0, \mathbf{w}, \varepsilon, N_t$).

repeat
  Initialize $\mathbf{u}$ given the initial guess $\mathbf{u}_0 \in \mathbb{R}^N$: $\mathbf{u} = \begin{bmatrix} \mathbf{u}_0 & \cdots & \mathbf{u}_0 \end{bmatrix}^T \in \mathbb{R}^{NN_t}$.
  Run 1 iteration of MGRIT($\Phi, \Psi, \mathbf{u}, \mathbf{w}, \varepsilon$) on the all-at-once system $A(\mathbf{u}) = \mathbf{w}$.
  Set $\mathbf{u}_0 \longleftarrow \mathbf{u}_{t^*}$, where $t^* \coloneqq \arg\min_{0 \le i \le N_t} \|G_{sF}(\mathbf{u}_i)\|$.
until $\|G_{sF}(\mathbf{u}_{N_t})\| / \|G_{sF}(\mathbf{u}_0)\| \le \varepsilon$.

search to only the C-points. Thus, letting $\widehat{\mathbf{u}}_i^{(k)}$ denote the approximate solution at time point $i = 0, \ldots, N_t$ of the kth GR-MGRIT iteration, we take the updated initial guesses of the all-at-once systems of the form (7.7) to be given by

$$\widehat{\mathbf{u}}_0^{(k+1)} \coloneqq \widehat{\mathbf{v}}_{t_k^*}^{(k)}, \quad \text{where} \quad t_k^* = j^{*(k)} m, \quad j^{*(k)} = \arg \min_{0 \le j \le N_T} \|\nabla f(\widetilde{\mathbf{v}}_{jm}^{(k)})\|. \tag{7.111}$$

Remark 43 (Greedy search does not require additional resources)**.** Following the strategy outlined in (7.111), we note that we only need to compute the gradients at the N_T C-points excluding the $j = 0$ C-point, i.e., the previous initial guess, since choosing $j = 0$ would cause algorithm 9 to repeat the previous iteration. Note that these gradient evaluations can be performed in parallel; since we already needed $N_p = N_T$ processors for full parallelization of algorithm 8, this additional search does not require additional resources for full parallelization.

For example, in figures 7.5 and 7.9, we see that the C-point gradient in each R-MGRIT iteration is often minimized at an earlier C-point index; this is specially evident in the γ-truncated R-MGRIT plots, for which the gradient curves feature noticeable dips early on. Based on these figures, we infer that GR-MGRIT will restart from approximate solutions at early C-points with a smaller gradient than the approximate solution at the final time point N_t or γN_t. In the following, we restrict ourselves to the setting of MP1, (5.15), and investigate the relation between these dips in the gradient coarse-grid trajectory and the optimal greedy restart indices.

7.6.1 Analysis of the MP1 optimal greedy restart index

By restricting our greedy search to the C-points, we can analytically evaluate the gradients via corollary 3 and identify the optimal restart point without needing to compute or store the F-points. In particular, for the MP1 operators $\Phi = I - sA_1$ and $\Psi = (I + msA_1)^{-1}$, where A_1 is the 1D Laplacian matrix from (2.5), we can show that the optimal greedy

restart index is well-defined and independent of the number N_T of coarse-grid intervals. In the following, we assume that the size of the Laplacian matrix N is sufficiently large that remark 11 holds, such that we may take its normalized eigenvalues $z_n = s\lambda_n$ as continuous in $(0, 1]$. In the following, we take the C-point indices to be continuous in $[0, N_T]$; in practice, all non-integer continuous indices are rounded to the nearest integer. We also omit the (k) superscripts, which denote the R-MGRIT iteration, for a cleaner notation.

Theorem 6 (Optimal greedy restart index for MP1)**.** *Let $\Phi = I - sA_1$ and $\Psi = (I + msA_1)^{-1}$, where A_1 is the 1D Laplacian matrix from (2.5) and $s = 1/L$. Then, the objective function for the optimal greedy restart C-point index j^* from (7.111) for MP1, (5.15), can be decoupled into a weighted sum of scalar equations for each eigenvalue of A_1, each with a minimizing index j_n^*. Furthermore, the optimal restart index j^* lies between the smallest and largest j_n^* indices and is independent of the number of coarse-grid intervals N_T for sufficiently large N_T.*

Proof. Recall that we seek to minimize the gradient norm at the C-points in (7.111). This is equivalent to minimizing the square of the gradient norm,

$$\min_j \|\nabla f(\widetilde{\mathbf{v}}_{jm})\|_2 = \min_j \|A_1 \widetilde{\mathbf{v}}_{jm}\|_2. \tag{7.112}$$

Recall that given $\Phi = I - sA_1$ and $\Psi = (I + msA_1)^{-1}$, their eigenvalues ϕ_n and ψ_n from (4.99), associated to the normalized eigenvalues $z_n = s\lambda_n \in (0, 1]$ of A_1, are given by

$$\phi_n = 1 - z_n, \quad \psi_n = \frac{1}{1 + mz_n}. \tag{7.113}$$

Expanding the initial guess in terms of the orthonormal eigenvectors $\mathbf{q}_n$ (the columns of Q (4.99)) gives $\mathbf{u}_0 = \sum_{n=1}^{N} u_{0,n}\mathbf{q}_n$. Substituting the explicit C-point formulation from (7.94), we decouple the matrix norm into a sum of N scalar components,

$$\|A_1 \widetilde{\mathbf{v}}_{jm}\|_2^2 = \sum_{n=1}^{N} \lambda_n^2 u_{0,n}^2 g_{j,n}^2, \tag{7.114}$$

where $g_{j,n}$ is the nth modal gradient of the jth C-point approximate solution (7.94) for $n = 1, \ldots, N$,

$$g_{j,n} = \left(1 + \sum_{i=0}^{j-2} \psi_n^i (\phi_n^m - 1)\right) \phi_n^m = \left[1 + \frac{1 - \psi_n^{j-1}}{1 - \psi_n}(\phi_n^m - 1)\right] \phi_n^m. \tag{7.115}$$

Note that $g_{j,n}$ depends only on the current coarse-time index j, the coarsening factor m, and the normalized eigenvalue z_n.

In the following argument, we consider j to be a continuous variable; given $n = 1, \ldots, N$, we treat (7.115) as a function of $j \in [0, N_T]$. By the chain rule, the critical points of (7.115) satisfy

$$\frac{\mathrm{d}}{\mathrm{d}j}\left[g_{j,n}^2\right] = 2g_{j,n}\frac{\mathrm{d}}{\mathrm{d}j}g_{j,n} = 0. \tag{7.116}$$

By (7.113), for $n = 1$ such that $z_n = 1$, we have that $\phi_1 = 0$ such that $g_{j,1} = 0$ for all j. In the following, assume that $n \geq 2$, such that for all $z_n \in (0, 1)$ we have that $\phi_n, \psi_n \in (0, 1)$. Thus,

$$\frac{\mathrm{d}}{\mathrm{d}j}g_{j,n} = -\phi_n^m\left(\frac{\phi_n^m - 1}{1 - \psi_n}\right)\psi_n^{j-1}\ln\psi_n < 0 \tag{7.117}$$

for all finite j. Consequently, the only finite critical point of $g_{j,n}^2$ is the root $g_{j_n^*,n} = 0$. Note that $g_{1,n} = \phi_n^m > 0$. We define the asymptotic modal gradient function

$$g_{\infty,n} \coloneqq \frac{(\phi_n^m - \psi_n)\phi_n^m}{1 - \psi_n} = \lim_{N_T \to +\infty} g_{N_T,n}. \tag{7.118}$$

To evaluate $g_{\infty,n}$, note that $\frac{1}{1-z} > 1 + z$ by the geometric series $\sum_{i=0}^{\infty} z^i$ and that by the Binomial Theorem, $(1 + z)^m > 1 + mz$ for all $m \geq 2$ and $0 \leq z \leq 1$. Hence, $\phi_n^m - \psi_n < 0$, such that $g_{\infty,n} < 0$. Therefore, the existence of $j_n^* \in (0, \infty)$ such that $g_{j_n^*,n} = 0$ root is guaranteed by the Intermediary Value Theorem. Furthermore, since $g_{j,n}^2 \geq 0$, the root is a minimum point. Hence, to find this minimizing root we set (7.115) to zero, which yields

$$1 + \frac{1 - \psi_n^{j-1}}{1 - \psi_n}(\phi_n^m - 1) = 0 \implies \psi_n^{j-1} = \frac{\psi_n - \phi_n^m}{1 - \phi_n^m}. \tag{7.119}$$

Taking the logarithm with respect to base ψ_n and isolating j yields the C-point index root

$$j_n^* = 1 + \frac{\ln\left(\frac{\psi_n - \phi_n^m}{1 - \phi_n^m}\right)}{\ln\psi_n}, \quad n = 2, \ldots, N; \tag{7.120}$$

note that j_n^* is possibly noninteger. For the numerator of j_n^*, note that $\phi_n^m - \psi_n < 0$ implies $\psi_n - \phi_n^m > 0$; since $\psi_n < 1$, it follows that $\psi_n - \phi_n^m < 1 - \phi_n^m \leq 1$. Therefore, $0 < \frac{\psi_n - \phi_n^m}{1 - \phi_n^m} < 1$, such that $-\infty < \ln\left(\frac{\psi_n - \phi_n^m}{1 - \phi_n^m}\right) < 0$. For the denominator, $0 < \psi_n < 1$ implies that $-\infty < \ln\psi_n < 0$. Since j_n^* is a sum of 1 plus a ratio of two finite, strictly

negative numbers, it evaluates to a finite positive real number for each n. Therefore, there exist finite minimum and maximum indices $j^*_{\min}$ and $j^*_{\max}$, respectively, such that

$$j^*_{\min} \leq j^*_n \leq j^*_{\max} < \infty \quad \forall n = 2, \ldots, N. \tag{7.121}$$

In particular, for a sufficiently large N_T, we have that $j^*_{\max} < N_T$.

Now, recall (7.114) and define

$$F(j) = \sum_{n=2}^{N} \lambda_n^2 u_{0,n}^2 g_{j,n}^2 = \sum_{n=2}^{N} w_n g_{j,n}^2 \tag{7.122}$$

with weights $w_n = \lambda_n^2 u_{0,n}^2 \geq 0$; note that we start the sum from $n = 2$ since $g_{j,1} = 0$ for all j. To find the critical points of $F(j)$, we take the derivative with respect to j, which yields

$$\frac{\mathrm{d}}{\mathrm{d}j} F(j) = 2 \sum_{n=1}^{N} w_n g_{j,n} \left(\frac{\mathrm{d}}{\mathrm{d}j} g_{j,n} \right). \tag{7.123}$$

Now, we seek to show that $j^* \coloneqq \arg\min F(j)$, for which we have that $\frac{\mathrm{d}}{\mathrm{d}j} F(j^*) = 0$, exists and lies between $j^*_{\min}$ and $j^*_{\max}$.

Recall that we have established that $g_{j,n} > 0$ for $j < j^*_n$, and $g_{j,n} < 0$ for $j > j^*_n$ and that $\frac{\mathrm{d}}{\mathrm{d}j} g_{j,n} < 0$ for all finite j. For any $j < j^*_{\min}$, we have $j < j^*_n$ for all $n \geq 2$, such that $g_{j,n} > 0$ for all $n \geq 2$. Since $\frac{\mathrm{d}}{\mathrm{d}j} g_{j,n} < 0$, every term in the sum from (7.123) is the product of a positive number and a negative number, such that $\frac{\mathrm{d}}{\mathrm{d}j} F(j) < 0$ for all $j < j^*_{\min}$. For any $j > j^*_{\max}$, we have $j > j^*_n$ for all $n \geq 2$, such that $g_{j,n} < 0$ for all n. Since $\frac{d}{dj} g_{j,n} < 0$, every term in the sum from (7.123) is the product of two negative numbers, such that $\frac{\mathrm{d}}{\mathrm{d}j} F(j) > 0$ for all $j > j^*_{\max}$.

Since $F(j)$ is strictly decreasing for $j < j^*_{\min}$ and strictly increasing for $j > j^*_{\max}$, the global minimum of the continuous extension of $F(j)$ must lie in the closed interval $[j^*_{\min}, j^*_{\max}]$ by the Intermediate Value Theorem. Since $j^*_{\max} < N_T$ for sufficiently large N_T, if follows that $j^* \leq j^*_{\max} < N_T$. Note that (7.114) implies that j^* does not have a closed form because the j^*_n roots (7.120) of each scalar equation $g_{j,n}$ (7.115) are not all identical. □

Although theorem 6 derives the optimal indices j^*_n in (7.120) by treating them as continuous variables, the actual algorithm requires discrete integer values for the coarse indices . Assuming a sufficiently dense spectrum, the optimal discrete C-point index $j \in$

$\{0, \ldots, N_T\}$ that minimizes the nth modal gradient will simply be the integer closest to the continuous minimizer j_n^*.

We illustrate our findings from theorem 6 with an example in figure 7.10. For A_1 (2.5) of size $N = 10$, $N_T = 100$ coarse-grid intervals, coarsening factor $m = 4$ and $\widehat{\ell} = 2$ MGRIT levels, we let $\widetilde{\mathbf{v}}_{jm}$ denote the approximate solution at C-points jm, $j = 0, \ldots, N_T$ after the one MGRIT iteration, starting from an initial guess $\mathbf{u}_0 \in \mathbb{R}^N$ with components randomly sampled from an uniform distribution in $[0, 1]$. Note in figure 7.10 that the $g_{j,n}$ (7.115) coarse-grid trajectories corresponding to each of the 10 λ_n eigenvalues from (2.6) cross the zero-gradient threshold near $j^* = 3$, which is the C-point index at which the squared gradient norm (7.114) of the approximate solution $\widetilde{\mathbf{v}}_{jm}$ is minimized. We note that in this example, $j^*_{\min} = 2$ and $j^*_{max} = 40$, such that all the modal gradient trajectories are minimized in the interior of $[0, N_T]$, as established in theorem 6.

In figure 7.11, by looking at the absolute modal gradients instead, the zero-crossing of $g_{j,n}$ for each n manifests itself as a dip in the $|g_{j,n}|$ absolute gradient trajectory, which briefly goes to zero before increasing again. We then see that minimum of $\|\nabla f(\widetilde{\mathbf{v}}_{jm})\|^2$ aligns almost exactly with the superposition of multiple $|g_{j,n}|$ curves dipping towards zero. We call this phenomenon *transient dip*, and discuss it more generally in remark 44.

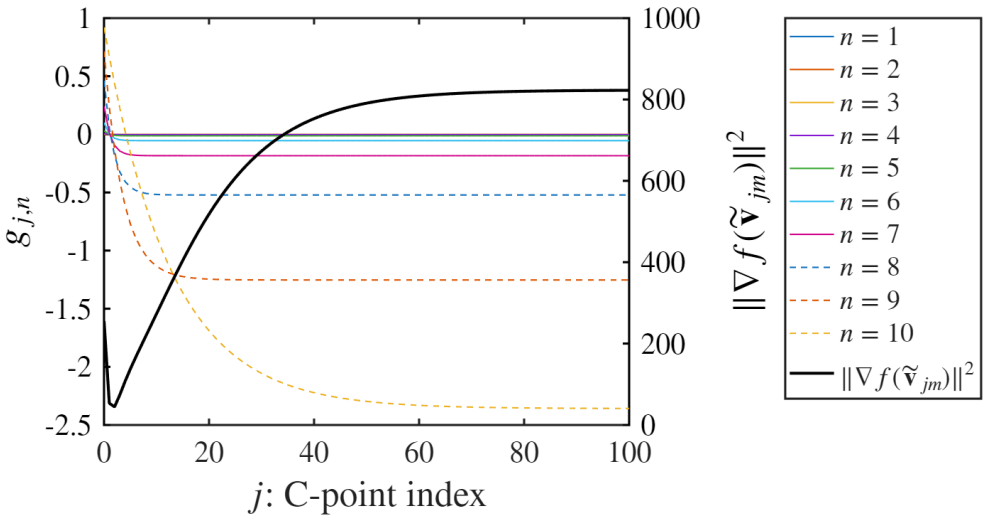


Figure 7.10: Left axis: modal scalar gradients $g_{j,n}$ (7.115) for modes $n = 1, \ldots, 10$ and C-point index $j = 0, \ldots, 100$. **Right axis**: squared gradient norm $\|\nabla f(\widetilde{\mathbf{v}}_{jm})\|^2$ (7.114), where $\nabla f(\widetilde{\mathbf{v}}_{jm})$ denotes the gradient of the C-point approximate solution at C-point index $j = 0, \ldots, 100$.

Remark 44 (Transient Dip)**.** By theorem 6, the absolute gradient trajectory for each mode, described by $|g_{j,n}|$ for $j \in [0, N_T]$, features a dip that occurs when the gradient

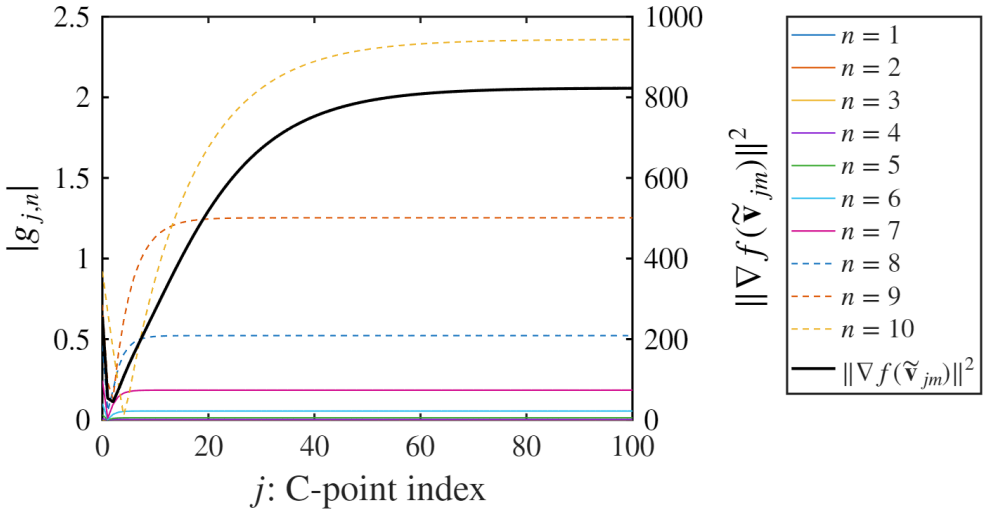


Figure 7.11: Left axis: absolute modal scalar gradients $|g_{j,n}|$ (7.115) for modes $n = 1, \ldots, 10$ and C-point index $j = 0, \ldots, 100$. **Right axis**: squared gradient norm $\|\nabla f(\widetilde{\mathbf{v}}_{jm})\|^2$ (7.114), where $\nabla f(\widetilde{\mathbf{v}}_{jm})$ denotes the gradient of the C-point approximate solution at C-point index $j = 0, \ldots, 100$.

$g_{j,n}$ (7.115) drops to zero at the C-point index j_n^* (7.120) as it changes sign from positive to negative, before it asymptotically climbs back toward the non-zero steady-state limit $g_{\infty,n}^2$. Recall that the squared gradient norm (7.114) is an aggregate of many such modes. Because each mode reaches its continuous root j_n^* at a different iteration index, their individual transient dips do not align perfectly. Consequently, the norm of the gradient of the approximate solution does not hit zero. Instead, the superposition of dips of neighbouring roots creates an aggregate effect, resulting in non-zero global minimum, as seen in figures 7.10 and 7.11, which occurs at an interior C-point index j^*. While MGRIT and R-MGRIT ignore this non-monotonicity by unconditionally evaluating the gradient at the final coarse index N_T, GR-MGRIT is able to exploit the transient dip in the gradient norm and restart from the gradient minimum, which by theorem 6 occurs at some interior index $0 \leq j^* < N_T$ for a sufficiently large N_T.

7.6.2 Estimating the MP1 optimal greedy restart index

By (7.111), we trivially have that an iteration of GR-MGRIT provides at least the same gradient reduction as an iteration of R-MGRIT if both start from the same initial guess, since by definition $\|\nabla f(\mathbf{v}_{t_k^*})\| \leq \|f(\mathbf{v}_{N_t})\|$. However, in the worst case-scenario, this ad-

ditional gradient reduction is too small to make up for the additional cost of the greedy index search. Thus, for GR-MGRIT to provide a better parallel speedup than R-MGRIT (in particular, with γ-truncated time domain), two things need to be true: *1.* the number of GR-MGRIT iterations for convergence must be smaller than R-MGRIT's, and *2.* the greedy restart index search (7.111) must be as cheap as possible. First, we propose a simple strategy for estimating the optimal greedy restart index under the assumptions of remark 11.

Remark 45 (Dominant mode heuristic for estimating the optimal greedy restart index)**.** Let $n^* := \arg\max_n w_n$ denote the index of the spectral mode with the largest initial gradient weight $w_n := \lambda_n u_{0,n}$, and let z_{n^*} be its corresponding normalized eigenvalue. Assuming the dominant weight strongly outweighs the others, i.e. $w_{n^*} \gg w_n$ for $n \neq n^*$, the behaviour of the total gradient (7.114) is heavily governed by the the n^*th modal gradient:

$$\|A_1 \widetilde{\mathbf{v}}_{jm}\|_2^2 \approx w_{n^*} g_{j,n^*}^2. \tag{7.124}$$

Hence, by minimizing only the dominant gradient, by (7.120) we obtain an inexpensive estimate $j_{n^*}^* \approx j^*$ for the optimal restart index whilst avoiding the additional cost of the greedy search (7.111).

To see how this work in practice, once again consider the example from figure 7.11. Recall from (2.6) that λ_n is strictly decreasing from $n = 1$ to $n = N$. Since the initial guess $\mathbf{u}_0$ is obtained from a uniform distribution, we have that $w_n \propto \lambda_n$, such that the largest eigenvalues dominate the sum in (7.114). Then, it is evident from figure 7.11 that by minimizing the $g_{j,n}$ for $n = 2$, for example, we would obtain $j_2^* \approx j^*$. However, while this holds for an initial MGRIT iteration, as R-MGRIT progresses and the spectrum of the updated initial guesses is filtered by the relaxation and coarse-grid correction stages, (7.124) may no longer hold.

More interesting, however, is the observation that the simplest strategy for taking advantage of the transient dip is to choose a fixed coarse restart index $\overline{N_T} \ll N_T$, since any substantial transient dip must occur at a time point before the approximate solution approaches its asymptotic limit; this is evident from the figures in section 7.4 and figure 7.11, for example. Not only is the γ-truncated R-MGRIT able to accurately approximate the GR-MGRIT gradient reduction without incurring the costs of the GR-MGRIT optimal restart index search (7.111), but we recall from section 7.5 that truncating the time domain yields in a substantially increased parallel wall-clock time speedup.

7.6.3 Numerical experiments for R-MGRIT vs. GR-MGRIT

To empirically verify our claims the γ-truncated R-MGRIT is an accurate approximation for GR-MGRIT, as well as test how the GR-MGRIT strategy performs in a nonlinear setting, we briefly present exploratory numerical experiments for comparing R-MGRIT to GR-MGRIT applied to our model problems from section 5.2. For MP1, (5.15), with the same parameter specifications as in the numerical experiments from section 7.4, we observe in figure 7.12 that GR-MGRIT does not reduce the number of MGRIT iterations required for convergence over R-MGRIT, with all requiring 6 iterations for both full-time and γ-truncated solves with $\gamma = 0.2$. In figure 7.13, we see that the transient dips in the gradient, when they exist, are practically negligible, such that very little additional gradient reduction is obtained by GR-MGRIT over R-MGRIT.

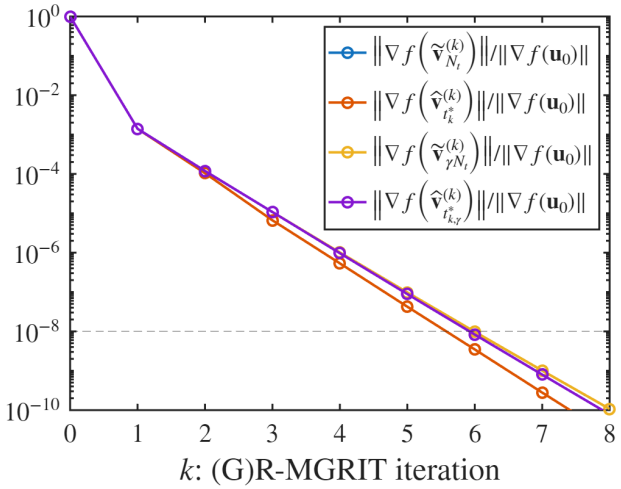


Figure 7.12: Convergence of MP1, (5.15), with (G)R-MGRIT using GD (5.30) on the fine grid and PP operator (5.31) on the coarse grid; $\nabla f(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denotes the gradient of the approximate solution at the final time point N_t and iteration k of R-MGRIT and $\nabla f(\widehat{\mathbf{v}}_{t_k^*}^{(k)})$ denotes the gradient of the approximate solution at the optimal C-point t_k^* (7.111) and iteration k of GR-MGRIT, respectively, with $\nabla f(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ and $\nabla f(\widehat{\mathbf{v}}_{t_{k,\gamma}^*}^{(k)})$ denoting their analogues for γ-truncated R-MGRIT and GR-MGRIT, respectively, with $\gamma = 0.2$; $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

Now, conside the nonlinear MP2-1D, (5.23), again with parameters as specified in the numerical experiments from section 7.4. We note that GR-MGRIT performs worse than R-MGRIT in the absence of post-relaxation, as seen in figure 7.14. In figure 7.15, we see that restarting from the C-point with smallest gradient does not reduce the generalized

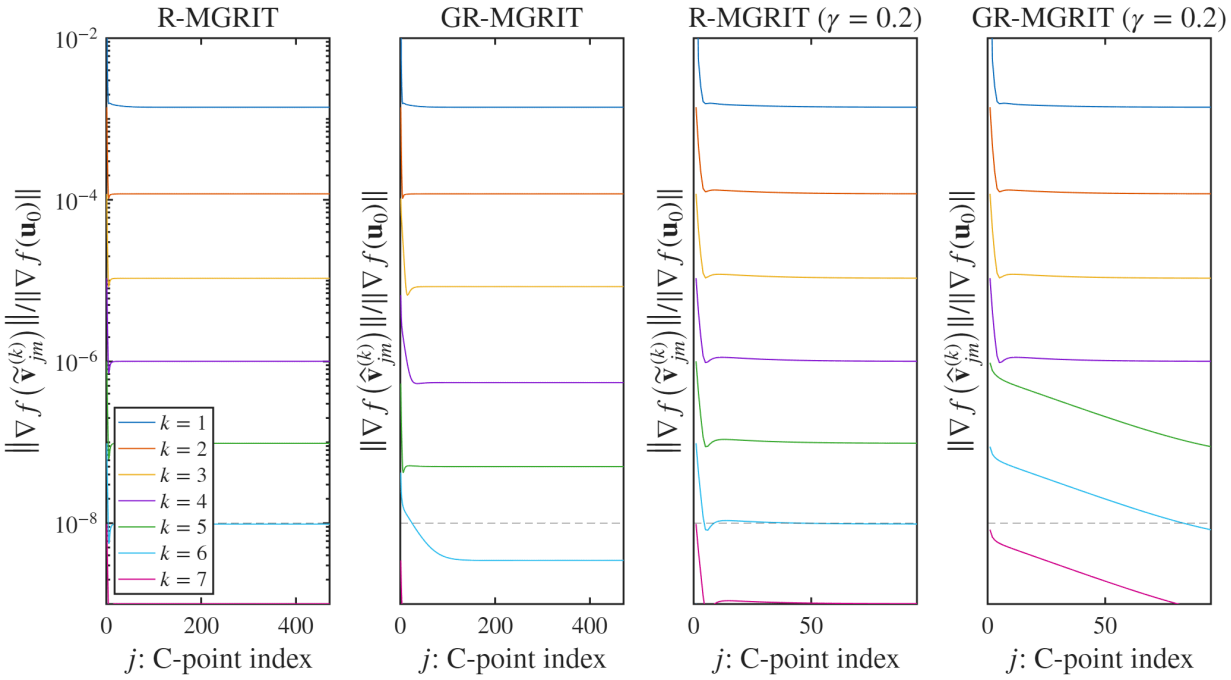


Figure 7.13: 2-norm of the gradient of the MP1, (5.15), approximate solutions at coarse time index j and iteration k of R-MGRIT, GR-MGRIT, γ-truncated R-MGRIT and γ-truncated GR-MGRIT (both with $\gamma = 0.2$), from left to right, divided by the 2-norm of the gradient of the initial guess $\mathbf{u}_0$; $\nabla f(\widetilde{\mathbf{v}}_{jm}^{(k)})$ and $\nabla f(\widehat{\mathbf{v}}_{jm}^{(k)})$ denote the gradient of the approximate solution at the C-point jm for $j = 0, 1, \ldots, N_T$ (restricted to $j = 0, 1, \ldots, \overline{N_T}$ for truncated (G)R-MGRIT) and iteration k of R-MGRIT and GR-MGRIT, respectively, and $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

gradient beyond approximately 10^{-4} of the initial gradient. We relate these results to figure 5.5, in which we observed the residual to be nonsmooth at the early C-points, one of which GR-MGRIT chooses to yield the new initial guess, and smooth at the final time point N_t, which is chosen by R-MGRIT.

Thus, we consider applying a post-relaxation step to both R-MGRIT and GR-MGRIT solves. With the additional relaxation steps, GR-MGRIT is now able to converge towards a minimizer of (5.23), as expected given the discussion from subsection 5.5.2. This is achieved by modifying the greedy optimal restart index search from (7.111) to now search for the smallest gradient out of the N_T F-points immediately preceding a C-point; that is, the optimal index is now given by $\arg\min_{1 \leq j \leq N_T} \|G_{sF}(\widetilde{\mathbf{v}}_{jm-1}^{(k)})\|$. By restricting the search to at the last F-point of each coarse-grid interval, we satisfy the desirable property from

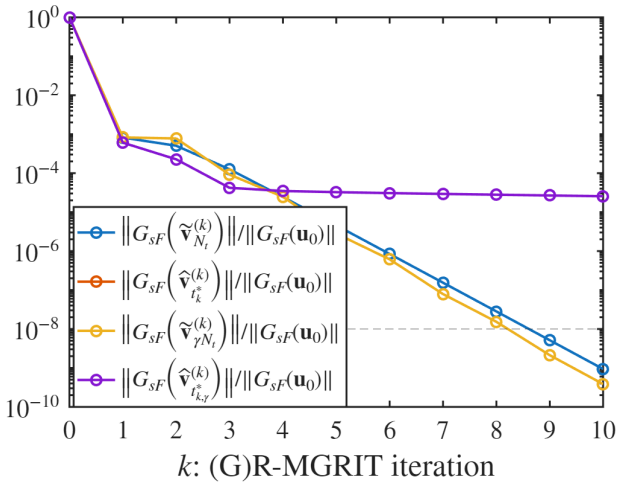


Figure 7.14: Convergence of MP2-1D, (5.23), with (G)R-MGRIT using PG (5.32) on the fine grid and AP operator (5.35) on the coarse grid; $G_{sF}(\widetilde{\mathbf{v}}_{N_t}^{(k)})$ denotes the generalized gradient (5.145) of the approximate solution at the final time point N_t and iteration k of R-MGRIT and $G_{sF}(\widehat{\mathbf{v}}_{t_k^*}^{(k)})$ denotes the generalized gradient of the approximate solution at the optimal C-point t_k^* (7.111) and iteration k of GR-MGRIT, respectively, with $G_{sF}(\widetilde{\mathbf{v}}_{\gamma N_t}^{(k)})$ and $G_{sF}(\widehat{\mathbf{v}}_{t_{k,\gamma}^*}^{(k)})$ denoting their analogues for γ-truncated R-MGRIT and GR-MGRIT, respectively, with $\gamma = 0.2$; $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

remark 43 that the search does not increase processor requirements for full-parallelization while taking full advantage of the additional gradient reduction yielded by the post F-relaxation.

However, by evaluating the R-MGRIT trajectory at $N_t - 1$, the final F-point, we see that R-MGRIT performs almost as well when taking advantage of a post-relaxation, as seen in figure 7.16. Notably, in figure 7.17, where we present the generalized gradient at the F-points immediately preceding a C-point, we see that the γ-truncated R-MGRIT and GR-MGRIT curves are practically identical across all iterations. Therefore, for our model problems, there is no advantage to be gained by using GR-MGRIT over R-MGRIT, since the additional gradient reduction does not translate into a reduction in the number of iterations required to reduce the (generalized) gradient by a factor of $\varepsilon = 10^{-8}$. In particular, we numerically verify that by choosing an adequate γ parameter such as $\gamma = 0.2$, we are able to approximate the gradient reduction of GR-MGRIT without the additional costs of the greedy index search (7.111).

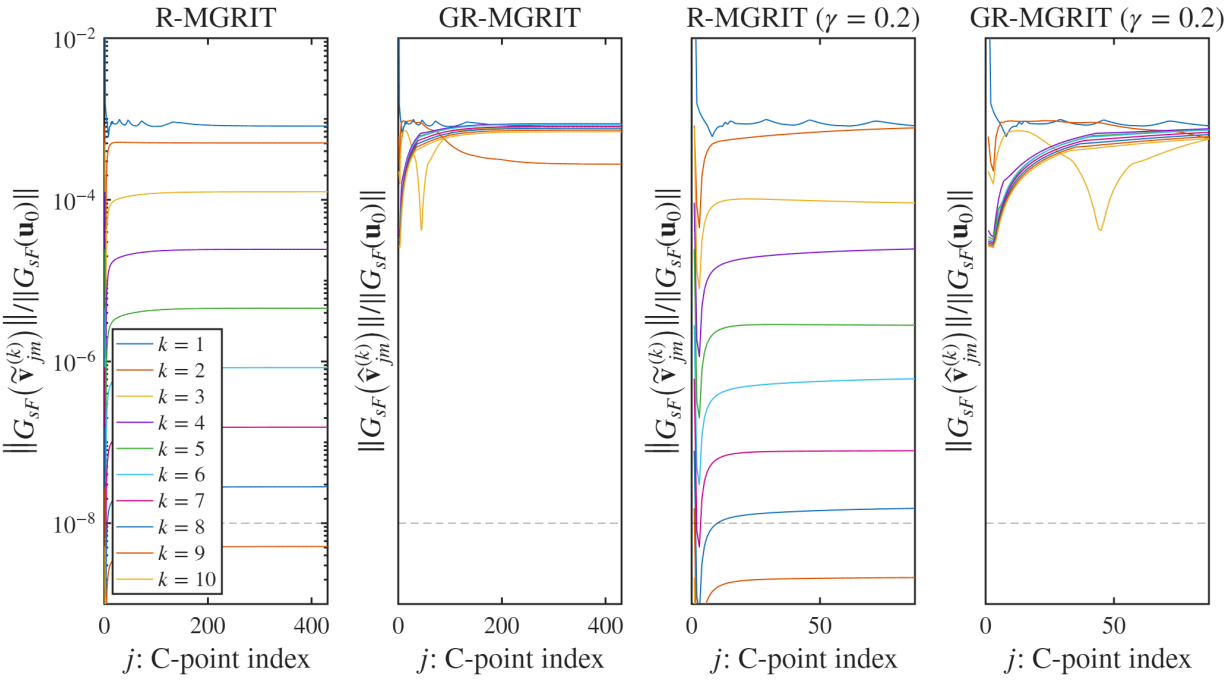


Figure 7.15: 2-norm of the generalized gradient (5.145) of the MP2-1D, (5.23), approximate solutions at coarse time index j and iteration k of R-MGRIT, GR-MGRIT, γ-truncated R-MGRIT and γ-truncated GR-MGRIT (both with $\gamma = 0.2$), from left to right, divided by the 2-norm of the generalized gradient of the initial guess $\mathbf{u}_0$; $G_{sF}(\widetilde{\mathbf{v}}_{jm-1}^{(k)})$ and $G_{sF}(\widehat{\mathbf{v}}_{jm-1}^{(k)})$ denote the generalized gradient of the approximate solution at the F-point $jm-1$ for $j = 1, \ldots, N_T$ (restricted to $j = 1, \ldots, \overline{N_T}$ for truncated (G)R-MGRIT) and iteration k of R-MGRIT and GR-MGRIT, respectively, and $G_{sF}(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

7.7 Conclusions

We have expanded on our parallel-in-iteration framework with a restarting, multi-trajectory variant, with which we are able to avoid the late-iteration gradient stalling often observed in MGRIT solves and possibly obtain a slight decrease in the number of iterations required for convergence. By applying this restarting scheme for a fixed fraction of the original time domain, we are able to obtain a multiplicative increase in the parallel wall-clock time speedup with a proportional reduction in the resource requirements for full parallelization. We prove for a relevant linear setting that both standard and restarting MGRIT algorithms converge asymptotically to identical final-point solutions which are quickly approximated in finite time, which further justifies truncating the time domain in order to save unnecessary

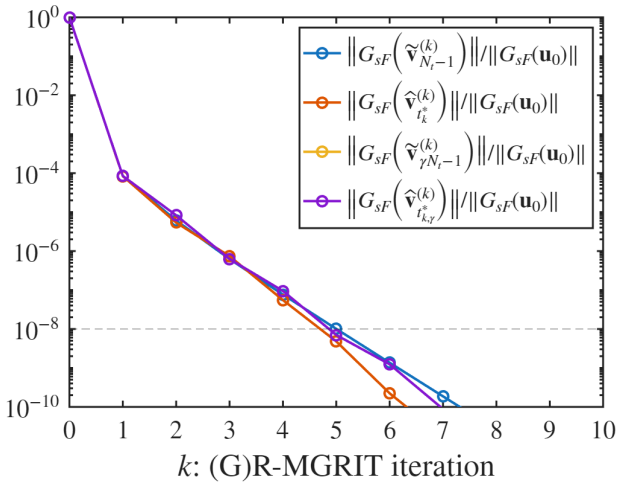


Figure 7.16: Convergence of MP2-1D, (5.23), with (G)R-MGRIT using PG (5.32) on the fine grid and AP operator (5.35) on the coarse grid, with an additional post F-relaxation; $G_{sF}(\widetilde{\mathbf{v}}^{(k)}_{N_t})$ denotes the generalized gradient (5.145) of the approximate solution at the final time point N_t and iteration k of R-MGRIT and $G_{sF}(\widehat{\mathbf{v}}^{(k)}_{t^*_k})$ denotes the gradient of the approximate solution at the optimal C-point t^*_k (7.111) and iteration k of GR-MGRIT, respectively, with $G_{sF}(\widetilde{\mathbf{v}}^{(k)}_{\gamma N_t})$ and $G_{sF}(\widehat{\mathbf{v}}^{(k)}_{t^*_{k,\gamma}})$ denoting their analogues for γ-truncated R-MGRIT and GR-MGRIT, respectively, with $\gamma = 0.2$; $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

work. We considered a greedy variant of our restarting scheme, but show that our restarting scheme with truncated time domain attains a similar final-point gradient reduction in a linear setting without requiring the additional costs incurred by the greedy scheme, while avoiding convergence issues that plague the greedy scheme in a nonlinear setting. For future work, we suggest investigating an optimal choice of truncation factor and initialization strategies that can possibly improve the results presented here.

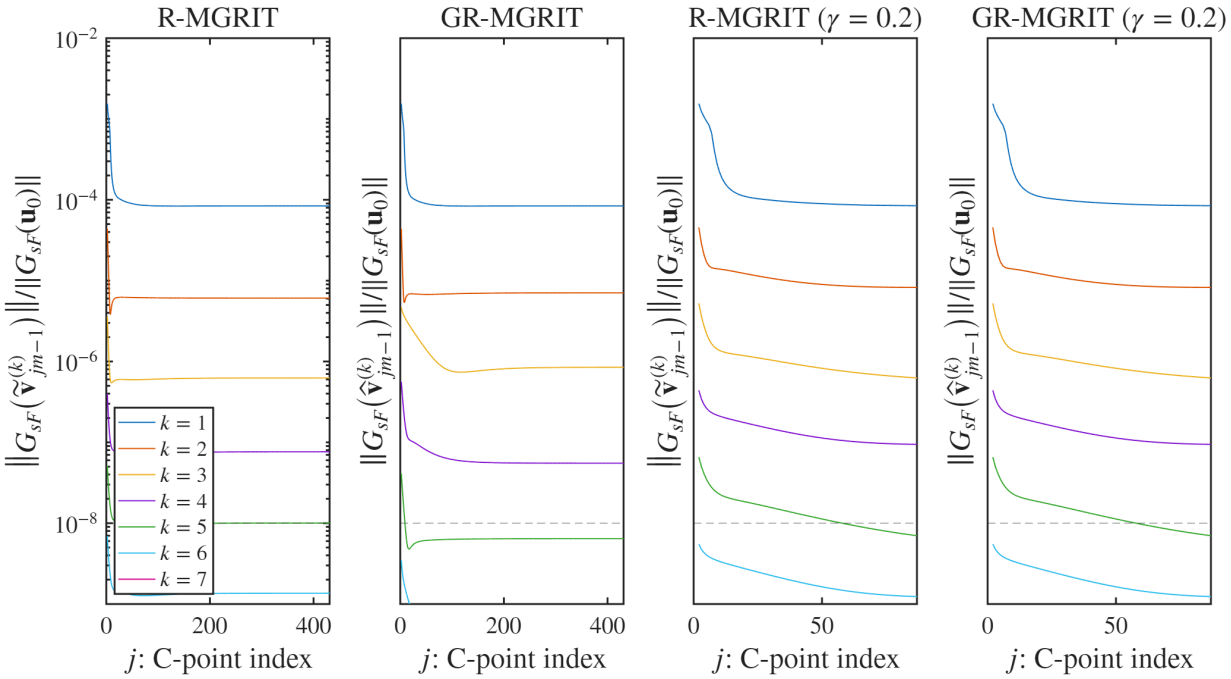


Figure 7.17: 2-norm of the generalized gradient (5.145) of the MP2-1D, (5.23), approximate solutions after post-relaxation at F-points of index $jm-1$ and iteration k of R-MGRIT, GR-MGRIT, γ-truncated R-MGRIT and γ-truncated GR-MGRIT (both with $\gamma = 0.2$), from left to right, divided by the 2-norm of the generalized gradient of the initial guess $\mathbf{u}_0$; $G_{sF}(\widetilde{\mathbf{v}}^{(k)}_{jm-1})$ and $G_{sF}(\widehat{\mathbf{v}}^{(k)}_{jm-1})$ denote the generalized gradient of the approximate solution at the F-point $jm-1$ for $j = 1, \ldots, N_T$ (restricted to $j = 1, \ldots, \overline{N_T}$ for truncated (G)R-MGRIT) and iteration k of R-MGRIT and GR-MGRIT, respectively, and $G_{sF}(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$.

Chapter 8

Final remarks

In chapter 5, we have successfully developed a parallel-in-iteration (P-in-I) framework using Multigrid Reduction-in-Time (MGRIT) for parallelizing over Gradient Descent (GD) and Proximal Gradient Descent (PG) iterations for solving two model problems, the linear and smooth Model Problem 1 (MP1), (5.15), and the nonlinear and nonsmooth Model Problem 2 (MP2), (5.23), which illustrate the performance of our framework for problems belonging to the general classes of optimization problems (5.1) for MP1 and (5.2) for MP2. In particular, we expect our framework to work well for similar problems where the Laplacian matrices from (2.5) and (2.9) are substituted by symmetric positive definite (SPD) matrices with a similar eigenstructure to Laplacian matrices. We refer to [89, 54] for MGRIT convergence analysis results related to the eigenvalues of the MGRIT operators.

Furthermore, in chapter 7 we have extended our framework with a Multi-trajectory (MT) approach, which allowed for reducing the wall-clock time of a P-in-I solve of our model problems while requiring fewer processors for a fully-parallel solve. By building on this framework, we hope to improve our current performance estimates, as well as solve many other optimization problems of interest for mathematicians and scientists of different fields. We note that in this thesis, we have not been successful in accelerating momentum-accelerated algorithms, as discussed in chapter 6, and that competing with the sequential performance of these methods remains the next challenge for our framework.

We conclude this thesis by briefly exploring some suggestions for future work. We discuss some methods related to the ones investigated in this thesis in section 8.1, and other problems of our interest in section 8.2. The goal of this section is to describe why our P-in-I framework cannot be used as is for these problems

8.1 Other optimization algorithms for consideration

Here, we briefly consider applying our P-in-I framework for different optimization problems and methods. While we have thus far only considered GD and Proximal Point (PP) for solving MP1, (5.15), and PG and Alternating Proximals (AP) for MP2, (5.23) (and at least in principle, their momentum-accelerated variants), we seek to develop a more general framework that accepts possibly many other optimization algorithms. The key requirement for our framework is that we should be able to approximate a fixed number of steps of the explicit method with one step of a corresponding implicit method.

Based on the discussion section 5.4, ideally we work with a pair of explicit and implicit methods that can be interpreted as explicit and implicit discretizations of the same Ordinary Differential Equation (ODE), or to behave similarly to the same ODE in the limit of arbitrarily small time steps. However, many relevant optimization methods currently do not have well-established ODE interpretations, such that this approach often does not bear fruit. Furthermore, while our general heuristic for gradient-based methods is to replace gradient steps by proximal steps, for more complex algorithms our heuristic may not yield a natural candidate for an implicit method if the cumulative work of a fixed number of successive steps of the explicit method is not easily describable or interpretable. Therefore, finding the ideal implicit approximations for the explicit algorithms of our choice is often a complicated task.

We briefly comment on a few relevant algorithms related to GD and PG and present some considerations on using these methods for our framework.

8.1.1 Gradient descent-related methods with variable stepsizes

In chapters 5 to 7, we only considered the GD method (5.30), and related methods such as PP (5.31) and Nesterov's Accelerated Gradient Descent (NAGD) (6.1) with fixed stepsize for all iterations, namely $s = 1/L$. Now, we consider a variant of (3.49) with iteration given by

$$\mathbf{u}_{k+1} = \mathbf{u}_k - s_k \nabla f(\mathbf{u}_k), \quad k = 0, 1, \dots \tag{8.1}$$

for minimizing a convex smooth function f, as in (3.48). The classical method of the (8.1) form is the Steepest Descent (SD) method, where

$$s_k = \min_s f(\mathbf{u}_k - s\nabla f(\mathbf{u}_k)), \quad k = 0, 1, \dots \tag{8.2}$$

is the optimal stepsize for the kth iteration. This iteration-wise optimal stepsize is typically found via line search; for minimizing a quadratic function of the form (5.24), this is

equivalent to computing

$$s_k = \frac{\langle \nabla f(\mathbf{u}_k), \nabla f(\mathbf{u}_k) \rangle}{\langle A \nabla f(\mathbf{u}_k), \nabla f(\mathbf{u}_k) \rangle}, \quad k = 0, 1, \ldots. \tag{8.3}$$

Should the $s_1, \ldots, s_{N_t}$ stepsizes be known in advance, we can use them to create our fine grid for an all-at-once system solve of the form (4.3), where s_i is the spacing between the fine-grid time points of indices $i-1$ and i. We note that MGRIT implementations with variable spacing are not new; for example, [55] considers variable coarsening factors for defining a coarse grid with non-uniform spacing. However, in practice, we cannot know the values of s_k without first running the SD method sequentially. While this is possible, and is in fact what we do in chapter 5, as noted in remark 16 this is not a generally useful practice; it is unlikely we would need to solve the same problem twice, or two problems with the exact same sequence of stepsizes.

Similarly, we may interpret the Newton-Raphson (NR) method (see subsection 2.5.1), which for solving a problem of the form (3.48) has its iteration update given by

$$\mathbf{u}_{k+1} = \mathbf{u}_k - [\nabla^2 f(\mathbf{u}_k)]^{-1} \nabla f(\mathbf{u}_k), \quad k = 0, 1, \ldots, \tag{8.4}$$

as a special case of (8.1) where the stepsize is no longer scalar, but instead obtained from inverting the Hessian $\nabla^2 f(\mathbf{u}_k)$. Even if the Hessians $\nabla^2 f(\mathbf{u}_k)$ are somehow known in advance or approximated by a fixed Hessian, such as in the simplified NR method (2.113), there is no intuitive notion of how to approximate m NR steps in one step of a similar implicit method. Even more complex gradient-based methods exist, such as Conjugate Gradient (CG), for which this fundamental problem of determining an accurate coarse-grid operator is considerably more difficult. Applying our P-in-I framework for such methods may require some complicated, problem and method-specific work, after which we may still find that our framework does not provide a worthwhile improvement in wall-clock time parallel speedup relative to the simpler GD solve we considered in chapters 5 and 7. Therefore, we leave the investigation of applying our framework to other GD-related methods for future work.

Remark 46 (Gradient descent with variable but predetermined stepsizes)**.** In general, it would be easier to work with methods of the form (8.1) where the sequence $\{s_k\}_{k=0,1,\ldots}$ of stepsizes can be computed (or estimated) independently of the iterations of (8.1), such as is the case for the NAGD momentum-parameter sequence $\beta_k = \frac{k-1}{k+2}$; see section 6.4. Then, while we would still need to solve N_T problems to determine an appropriate coarse-grid approximation of each coarse-grid interval, for methods of the form (8.1) this is intuitively similar to the fixed-stepsize approximation discussed in subsection 5.5.1. For example, to approximate the first m fine-grid steps of (8.1) variant with stepsizes $s_1, \ldots, s_m$, we initially consider applying one PP iteration with stepsize $\widehat{s} = \sum_{i=1}^{m} s_i$.

8.1.2 Proximal-gradient related methods

As previously mentioned in subsection 3.3.2, we choose AP (3.91) when we need to approximate a fixed number of PG steps (3.84). However, there are other implicit splitting algorithms which converge to an exact minimizer of $F = f + g$. In particular, we briefly present two methods related to PG and AP, known as the Douglas-Rachford (DR) and Peaceman-Rachford (PR) methods. Before we do so, we must define *reflectants* of functions, which are related to proximals (3.35).

Definition 38 (Reflectant)**.** *The reflectant (or reflected proximal) of f is defined by*

$$\mathcal{R}_f := 2\mathcal{P}_f - I, \tag{8.5}$$

where $\mathcal{P}_f$ is the proximal of f (3.35).

Assume that $F := f + g$ for convex, possibly nondifferentiable functions f and g. Then, we note that the reflectants of f and g are directly related to the set of minimizers of $f + g$.

Fact 44. *[7, Propositions 31.3 and 31.4] The reflectants $\mathrm{R}_f, \mathrm{R}_g$ are nonexpansive, and so is their composition $\mathrm{R}_g \circ \mathrm{R}_f$. Furthermore,*

$$\mathcal{P}_f\left(\mathrm{Fix}\left(\mathrm{R}_g \circ \mathrm{R}_f\right)\right) = Z, \tag{8.6}$$

where $Z := \mathrm{zer}(\partial f + \partial g)$ (3.81); see fact 37.

The DR algorithm [6, 7, 63, 25], starting from some $\mathbf{u}_0 \in \mathbb{R}^n$, is described by the update [7, Lemma 31.6]

$$\begin{aligned} \mathbf{u}_{k+1} &:= (I - \mathcal{P}_f + \mathcal{P}_g \circ (2\mathcal{P}_f - I))(\mathbf{u}_k) \\ &= \frac{1}{2}\left(I + \mathcal{R}_g \circ \mathcal{R}_f\right)(\mathbf{u}_k), \quad k = 0, 1, \ldots. \end{aligned} \tag{8.7}$$

The DR algorithm not only converges to a minimizer of $F := f + g$; it converges for the sum of any convex functions, without requiring differentiability. Note, however, that it is not the sequence generated by (8.7) that directly converges to the minimizer, but rather the sequence of the proximals of the DR iterates.

Fact 45. *[7, Theorem 31.7] Suppose that $Z \neq \varnothing$, where Z is defined in (3.81). Then, the DR sequence generated by (8.7) starting from some $\mathbf{u}_0 \in \mathbb{R}^n$ converges to $\widetilde{\mathbf{u}}$, where $\mathcal{P}_f(\widetilde{\mathbf{u}}) \in Z$.*

In particular, the DR method may be interpreted as special case of the PP method [25]. Now, assume that $F = f + g$ where f and g are both possibly nondifferentiable, with f strongly convex. Then, we consider the PR algorithm [6, 7, 63], starting from some $\mathbf{u}_0 \in \mathbb{R}^n$, which is described by the update

$$\mathbf{u}_{k+1} \coloneqq \mathcal{R}_g \circ \mathcal{R}_f(\mathbf{u}_k), \quad k = 0, 1, \dots. \tag{8.8}$$

Similarly to DR, the proximals of the PR sequence generated by (8.8) converge to a minimizer of $F \coloneqq f + g$.

Fact 46. *[7, Theorem 32.3] Suppose that $Z \neq \varnothing$, where Z is defined in (3.81) for some strongly convex function f. Then, Z is single-valued, i.e., $Z = \{\mathbf{u}^*\}$, and the PR sequence generated by (8.8) starting from some $\mathbf{u}_0 \in \mathbb{R}^n$ converges to $\widetilde{\mathbf{u}}$, where $\mathcal{P}_f(\widetilde{\mathbf{u}}) = \mathbf{u}^*$.*

Recall that the AP method (3.91) converges to a minimizer of $\operatorname{env}(f) + g$ rather than $f + g$; see subsection 3.3.2. Therefore, in the sense of being implicit methods capable of converging to the minimizer of convex functions without requiring L-smoothness (or even differentiability), we have that DR and PR (in the strongly convex case) are more suitable implicit methods for solving problems of the form (3.79), of which MP2, (5.23), is a special case. However, recall from section 5.4 that AP is the natural implicit equivalent of PG from an ODE perspective; not only does this choice allows us to test our heuristic from remark 12 in a challenging setting, given that the fine- and coarse-grid operators have different fixed point sets, but in preliminary numerical experiments we found AP to yield a better coarse-grid approximation for PG than DR and PR. We illustrate this discussion with a simple 2D example, in which we minimize the function

$$F(\mathbf{x}) = \min_{\mathbf{x} \in \mathbb{R}^2} \frac{1}{2}\langle A\mathbf{x}, \mathbf{x}\rangle - \langle \mathbf{b}, \mathbf{x}\rangle + \lambda \|\mathbf{x}\|_1, \tag{8.9}$$

with $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$, $A = \begin{bmatrix} 2 & 1/2 \\ 1/2 & 2 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$ and $\lambda = 0.5$; note that this is similar to MP2, (5.23), although with a different SPD matrix A and a simpler L^1-penalty term. In figure 8.1, we plot the trajectories of the PG, AP, DR and PR methods starting from the same initial guess $\mathbf{x}_0 = \begin{bmatrix} -9 & 8 \end{bmatrix}^T$. For PG, we take $s = 1/L$ and plot every 2 steps, while for the other methods we take a stepsize $s = 2/L$ and plot every step of the implicit algorithms.

In figure 8.1, we see that while DR and PR converge faster to the solution $\mathbf{x}^*$, they diverge considerably from the PG trajectory, which is better approximated by AP. However, as expected, the AP trajectory does not converge to $\mathbf{x}^*$, the minimizer of $F \coloneqq f + g$, but to a nearby point, the minimizer of $\operatorname{env}(f) + g$, where $f \coloneqq \frac{1}{2}\langle A\mathbf{x}, \mathbf{x}\rangle - \langle \mathbf{b}, \mathbf{x}\rangle$ and

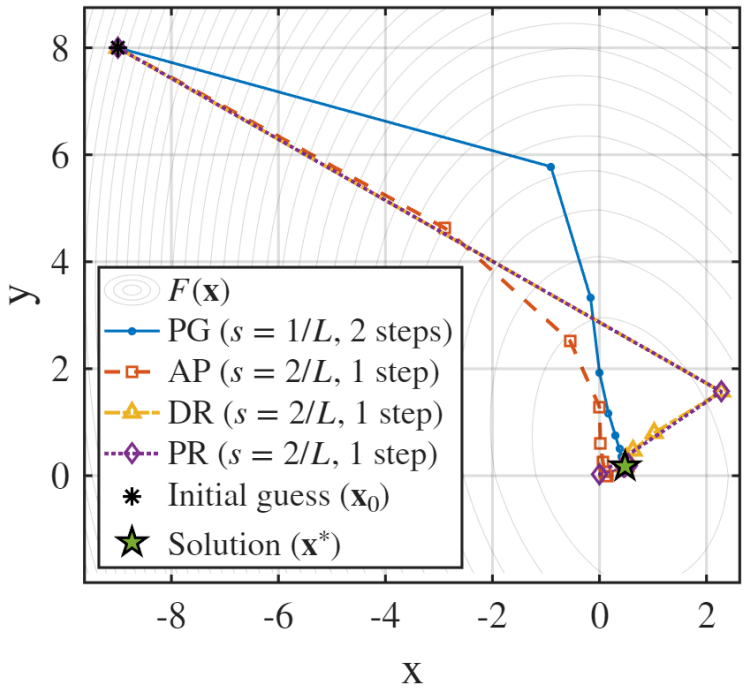


Figure 8.1: Comparison of PG (3.84), AP (3.91), DR (8.7) and PR (8.8) trajectories for minimizing $F(\mathbf{x})$ from (8.9), starting from the same initial guess. Here, x and y denote the coordinates of the $\mathbf{x} \in \mathbb{R}^2$ vectors generated by each optimization method.

$g := \lambda\|\mathbf{x}\|_1$. This figure also reinforces that in the MGRIT context of solving MP2, (5.23), P-in-I with the PG (5.32) fine-grid operator and the AP (5.35) coarse-grid operator, only a small number of PG relaxation steps is necessary to correct the AP coarse-grid trajectory towards the minimizer of F. Hence, we leave the consideration of these methods in our P-in-I framework for future work.

8.2 Other optimization problems

We now briefly present a few additional problems of possible interest for future work. These are widely known and studied problems, for which a successful P-in-I implementation could prove itself to be practically useful. A key limitation of our framework as presented in this thesis is that we require the matrices in our problems (e.g., the 1D and 2D Laplacians from (2.5) and (2.9)) to be square; furthermore, most of our analysis assume the matrices are SPD. For many relevant problems, especially those arising in statistics and machine

learning contexts, this is rarely the case: if the matrix in question describe observations of certain variables, we may have millions of rows and a handful of columns, for example; we discuss this in subsection 8.2.1. Furthermore, in subsection 8.2.2 we discuss optimization problems that are not given in terms of matrices, which require analysis of a different kind than that discussed in this thesis. Finally, in subsection 8.2.3 we consider working with constrained optimization problems directly, unlike in the case of MP2, (5.23), where we approximately or exactly satisfy the problems constraints via a penalty term.

8.2.1 Statistics and machine learning problems

Many problems in statistics and machine learning are related to problems of estimating the value of one unobservable variable based on the values of observable variables after multiple observations. Let $\mathbf{y} \in \mathbb{R}^n$ be the vector of observations of a dependent binary variable y, and let $X \in \mathbb{R}^{m \times n} = \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_m \end{bmatrix}$, where $\mathbf{x}_j \in \mathbb{R}^n$ for $j = 1, \ldots, m$, be a dataset of independent continuous variables $x_1, \ldots, x_m$ which we want to use to predict the value of the binary variable y. That is, we have observations of the form

$$\left[\begin{array}{c|ccc} (\mathbf{y})_1 & (\mathbf{x}_1)_1 & \cdots & (\mathbf{x}_m)_1 \\ \vdots & \vdots & \ddots & \vdots \\ (\mathbf{y})_n & (\mathbf{x}_1)_n & \cdots & (\mathbf{x}_m)_n \end{array}\right] = \left[\begin{array}{c|c} (\mathbf{y})_1 & (\mathbf{x})_1 \\ \vdots & \vdots \\ (\mathbf{y})_n & (\mathbf{x})_n \end{array}\right] = \left[\begin{array}{c|c} \mathbf{y} & X \end{array}\right]. \tag{8.10}$$

We consider this setting and notation for the following problems.

Ordinary Least Squares and LASSO

The Ordinary Least Squares (OLS) (or Linear Least Squares) problem is a special case of (3.48); i.e., convex smooth unconstrained minimization problem given by

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|X\mathbf{w} - \mathbf{y}\|_2^2 \tag{8.11}$$

where $\mathbf{w}$ denotes a vector of weights. If X^TX is invertible, the OLS problem has a closed-form solution given by $\mathbf{w}^* = (X^TX)^{-1}X\mathbf{y}$. Among the issues of OLS is the fact that it can be prone to overfitting, by which we mean that the weight vector $\mathbf{w}$ is excessively tailored to the specific X matrix, such that replacing X with another observation matrix $\widehat{X}$ may lead to a large residual $\widehat{X}\mathbf{w}^* - \mathbf{y}$, where $\mathbf{w}^*$ is the OLS solution for X. Furthermore, the prediction mechanism from X to $\mathbf{y}$ can be hard to interpret if $\mathbf{w}$ is not sparse.

To handle these issues, (8.11) is often replaced by the nonsmooth convex problem known as Least Absolute Shrinkage and Selection Operator (LASSO):

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|X\mathbf{w} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_1, \tag{8.12}$$

where $\lambda > 0$ is some predetermined penalty parameter on the weights. While (8.12) has no closed-form solution, it can be solved using splitting algorithms such as PG (3.84) with stepsize $s = 1/L$, $L = \|X\|_2^2$. Note that (8.12) is a special case of (3.79) for an objective function $F \coloneqq f + g$ defined in terms of $f \coloneqq \frac{1}{2}\|X\mathbf{w} - \mathbf{y}\|_2^2$, the convex smooth part, and $g \coloneqq \lambda\|\mathbf{w}\|_1$, the nondifferentiable convex part. Note that the relation between the OLS and LASSO problems, (8.11) and (8.12), respectively, is analogous to that between MP1, (5.15), and MP2, (5.23). In particular, if X is SPD with a suitable eigenstructure, then we expect our P-in-I framework to work very similarly to the numerical experiments from sections 5.7 and 7.4. However, for general OLS and LASSO problems, we first need to be able to extend our framework for possibly non-square matrices, as previously mentioned.

Logistic Regression

Now, assume that the variable y we want to estimate is binary, such that $(\mathbf{y})_i \in \{0, 1\}$ for all $i = 1, \ldots, n$ in (8.10). Therefore, instead of finding a weight vector $\mathbf{w}$ that gives the best approximation $X\mathbf{w} \approx \mathbf{y}$, as in the OLS and LASSO cases, we want to find the $\mathbf{w}$ that best predicts the probability distribution of y in terms of X. Recall that the sigmoid function is given by $\sigma(\mathbf{z}) = \frac{1}{1+\exp(-\mathbf{z})}$, and note that $\sigma(\mathbf{z}) \in (0, 1)$ for all $\mathbf{z}$. Then, we define

$$h(\mathbf{w}, \mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{x})}, \tag{8.13}$$

which we interpret as $P(y = 1 \mid \mathbf{x})$; i.e., the conditional probability of $y = 1$ occurring given $\mathbf{x}$. Logistic Regression (LR) then consists of minimizing the *cross-entropy loss function*

$$J(\mathbf{w}) = -\left(\sum_{i=1}^{m} y_i \log h\left(\mathbf{x}_i\right) + (1 - y_i) \log\left(1 - h\left(\mathbf{x}_i\right)\right)\right), \tag{8.14}$$

which has unique minimum but no closed form solution. Note that the gradient of the cross-entropy function is given by

$$\nabla J(\mathbf{w}) = X^T(h(\mathbf{w}, X) - \mathbf{y}). \tag{8.15}$$

Therefore, we can minimize $J(\mathbf{w})$ with GD; for this LR problem, the GD iteration (3.49), starting from $\mathbf{w}_0 = \mathbf{0}$, is given by

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha X^T(h(\mathbf{w}_k, X) - \mathbf{y}), \quad k = 0, 1, \ldots, \tag{8.16}$$

where $\alpha > 0$ is the stepsize. We can compute or estimate $L = \|X\|_2^2$ in order to find the Courant-Friedrichs-Lewy (CFL) limit $2/L$ for the stepsize α. Then, on the coarse grid, remark 12 recommends we work with PP iterations (3.75) of the form $\mathbf{w}_{k+1} = \mathcal{P}_{\alpha J}(\mathbf{w}_k)$, where

$$\mathcal{P}_{\alpha J}(\mathbf{v}) = \arg\min_{\mathbf{w}} \left(J(\mathbf{w}) + \frac{1}{2\alpha}\|\mathbf{v} - \mathbf{w}\|^2 \right) \tag{8.17}$$

has no closed form and must be computed numerically. While this approach is consistent with our work for MP1, (5.15), we note that the convergence of GD for logistic regression as a function of the stepsize α depends on a property of the data matrix $\left[\begin{array}{c|c} \mathbf{y} & X \end{array}\right]$ from (8.10), known as *linear separability*. When $\left[\begin{array}{c|c} \mathbf{y} & X \end{array}\right]$ satisfies this property, (8.16) may converge even with $s > 2/L$; on the other hand, if the data is not linearly separable, we may observe a cycling behaviour even for $s \ll 2/L$. Therefore, even if restricting our analysis to square X matrices, which are more naturally suitable to our framework, logistic regression introduces many special considerations beyond the scope of this thesis.

8.2.2 The convex feasibility problem (CFP)

We consider a general problem similar in form to (3.79). For a simple example, let C_1 and C_2 be two convex sets; see definition 9. Then, the Convex Feasibility Problem (CFP) consists of finding a point in the intersection between these sets:

$$\text{find } \mathbf{u}^* \in C_1 \cap C_2. \tag{8.18}$$

Besides DR and PR, another classical method is the Method of Alternating Projections (MAP) [6, 7], which is related to PG and AP:

$$\mathbf{u}_{k+1} = \mathcal{P}_{\iota_{C_2}} \circ \mathcal{P}_{\iota_{C_1}}(\mathbf{u}_k), \tag{8.19}$$

where ι_{C_i} denotes the indicator function of C_i. In particular, we refer to proximals of the form $\mathcal{P}_{\iota_C}$, where C is a closed, convex set, as *projections* onto C. One special property of projections is that $\mathcal{P}_{\iota_C}(\mathbf{u}) \in C$ for all $\mathbf{u}$. Therefore, by alternating projections, we alternate between points in C_1 and C_2, whilst simultaneously approximating the intersection between them. Having reached the intersection $C_1 \cap C_2$, the projections will no longer update the MAP sequence, such that any point $\mathbf{u}^* \in C_1 \cap C_2$ is a fixed point of (8.19).

In particular, by (3.82), problems of the form (3.79) are equivalent to (8.18) for $\mathbf{u}^* = \mathbf{0}$, $C_1 = \operatorname{zer}(\nabla f)$ and $C_2 = \operatorname{zer}(\partial g)$. Therefore, our analysis for MP2, (5.23), may serve as a starting point for exploring the application of our P-in-I framework to the CFP, although the weaker connection to time integration problems make it so that a careful problem-specific analysis must be done.

8.2.3 Minimizing sums of functions with linear constraints

As another example of a problem which can be considered in future work, we consider

$$\min_{\mathbf{u}\in\mathbb{R}^n, \mathbf{v}\in\mathbb{R}^m} \quad f(\mathbf{u}) + g(\mathbf{v}) \quad \text{s.t.} \quad A\mathbf{u} + B\mathbf{v} = \mathbf{c} \tag{8.20}$$

for $\mathbf{c} \in \mathbb{R}^q$ and linear operators $A : \mathbb{R}^n \to \mathbb{R}^q$ and $B : \mathbb{R}^m \to \mathbb{R}^q$. In particular, we may consider (8.20) as a generalization of (3.79) where f and g now have possibly different domains. Furthermore, note that linear constraint $A\mathbf{u} + B\mathbf{v} = \mathbf{c}$ can be made inactive by choosing a trivially satisfied condition, such as by letting A and B be the zero operators in $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively, with $\mathbf{c} = \mathbf{0}$, such that (8.20) also describes unconstrained minimization problems.

A classical method for solving (8.20) is the Alternating Direction Method of Multipliers (ADMM). For a penalty parameter $\rho > 0$, we define the *augmented Lagrangian*

$$\mathcal{L}_\rho(\mathbf{u}, \mathbf{v}, \mathbf{w}) = f(\mathbf{u}) + g(\mathbf{v}) + \langle A\mathbf{u} + B\mathbf{v} - \mathbf{z}, \mathbf{w}\rangle + \frac{\rho}{2}\|A\mathbf{u} + B\mathbf{v} - \mathbf{z}\|_2^2. \tag{8.21}$$

Then, the ADMM update for $k = 0, 1, \ldots$ consists of the following three steps (in order):

$$\mathbf{u}_{k+1} = \arg\min_{\mathbf{u}} \mathcal{L}_\rho(\mathbf{u}, \mathbf{v}_k, \mathbf{w}_k); \tag{8.22}$$

$$\mathbf{v}_{k+1} = \arg\min_{\mathbf{u}} \mathcal{L}_\rho(\mathbf{u}_{k+1}, \mathbf{v}, \mathbf{w}_k); \tag{8.23}$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \rho(A\mathbf{u}_{k+1} + B\mathbf{v}_{k+1} - \mathbf{c}). \tag{8.24}$$

In particular, for $\rho = 1$, $m = n = q$, $A = I$, $B = -I$ and $\mathbf{c} = 0$, the ADMM update can be written in terms of proximals as

$$\mathbf{u}_{k+1} = \mathcal{P}_f(\mathbf{v}_k - \mathbf{w}_k); \tag{8.25}$$

$$\mathbf{v}_{k+1} = \mathcal{P}_g(\mathbf{u}_{k+1} + \mathbf{w}_k); \tag{8.26}$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{u}_{k+1} - \mathbf{v}_{k+1}. \tag{8.27}$$

Letting $\mathbf{z}_k = \mathbf{u}_k + \mathbf{w}_{k-1}$ for all $k \geq 1$, we have that $\mathbf{v}_k = \mathcal{P}_g(\mathbf{z}_k)$ and $\mathbf{u}_{k+1} = \mathcal{P}_f \circ \mathcal{R}_g(\mathbf{z}_k)$, such that

$$\mathbf{z}_{k+1} = \mathbf{z}_k - \mathcal{P}_g(\mathbf{z}_k) - \mathcal{P}_f \circ \mathcal{R}_g(\mathbf{z}_k); \tag{8.28}$$

note that $\{\mathbf{z}_k\}_{k\in\mathbb{N}}$ is exactly the sequence of DR iterates given by (8.7). Therefore, ADMM can be interpreted as a special case of DR; it can also be shown that the reverse is true, i.e., DR is a special case of ADMM [25, 84].

8.3 Conclusion

In this final chapter, we present a few interesting problems and methods for consideration in future developments of our parallel-in-iteration framework. The two main obstacles detailed here are handling optimization problems involving non-square matrices and taking advantage of (or competing with) adaptive schemes that optimize certain parameter choices given the trajectory of the corresponding iterative method. The success of our framework for handling such issues may benefit from the ever-evolving developments of parallel-in-time methods for time integration problems.

References

[1] Andersen Ang, Hans De Sterck, and Stephen Vavasis. MGProx: A nonsmooth multigrid proximal gradient method with adaptive restriction for strongly convex optimization. *SIAM Journal on Optimization*, 34(3):2788–2820, August 2024.

[2] Joshua D. Angrist and Jörn-Steffen Pischke. *Mostly Harmless Econometrics: An Empiricist's Companion*. Princeton University Press, Princeton, USA, December 2008.

[3] G. H. M. Araújo, O. A. Krzysik, and H. De Sterck. Parallel-in-iteration optimization using multigrid reduction in time. *ETNA - Electronic Transactions on Numerical Analysis*, 66:32–63, 2026.

[4] Hedy Attouch, Zaki Chbani, Juan Peypouquet, and Patrick Redont. Fast convergence of inertial dynamics and algorithms with asymptotic vanishing viscosity. *Mathematical Programming*, 168(1–2):123–175, March 2016.

[5] Chenglong Bao, Liang Chen, Jiahong Li, and Zuowei Shen. Accelerated gradient methods with gradient restart: Global linear convergence, 2024. https://arxiv.org/abs/2401.07672.

[6] Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer International Publishing, 2017.

[7] Heinz H. Bauschke and Walaa M. Moursi. *An Introduction to Convexity, Optimization, and Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, USA, January 2023.

[8] Amir Beck. *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB*. Society for Industrial and Applied Mathematics, October 2014.

[9] Amir Beck. *First-Order Methods in Optimization*. Society for Industrial and Applied Mathematics, October 2017.

[10] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, January 2009.

[11] Achi Brandt and Colin W. Cryer. Multigrid algorithms for the solution of linear complementarity problems arising from free boundary problems. *SIAM Journal on Scientific and Statistical Computing*, 4(4):655–684, December 1983.

[12] William L Briggs. *A multigrid tutorial*. Society for Industrial and Applied Mathematics, Philadelphia, Pa., 2 edition, 2000.

[13] L. A. Caffarelli. The obstacle problem revisited. *The Journal of Fourier Analysis and Applications*, 4(4–5):383–402, July 1998.

[14] Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. "Convex until proven guilty": Dimension-free acceleration of gradient descent on non-convex functions. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 654–663. PMLR, 06–11 Aug 2017.

[15] Yair Censor. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University PressNew York, NY, January 1998.

[16] Antonin Chambolle and Thomas Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25:161–319, May 2016.

[17] Andrew J. Christlieb, Colin B. Macdonald, and Benjamin W. Ong. Parallel high-order integrators. *SIAM Journal on Scientific Computing*, 32(2):818–835, January 2010.

[18] Joshua Christopher, Robert D. Falgout, Jacob B. Schroder, Stephen M. Guzik, and Xinfeng Gao. A space-time parallel algorithm with adaptive mesh refinement for computational fluid dynamics. *Computing and Visualization in Science*, 23(1-4), 2020.

[19] H. De Sterck, R. D. Falgout, O. A. Krzysik, and J. B. Schroder. Efficient multigrid reduction-in-time for method-of-lines discretizations of linear advection. *Journal of Scientific Computing*, 96(1), May 2023.

[20] H. De Sterck, S. Friedhoff, O. A. Krzysik, and S. P. MacLachlan. Multigrid reduction-in-time convergence for advection problems: A Fourier analysis perspective. *Numerical Linear Algebra with Applications*, 32(1), October 2024.

[21] Hans De Sterck, Robert D. Falgout, Stephanie Friedhoff, Oliver A. Krzysik, and Scott P. MacLachlan. Optimizing multigrid reduction-in-time and parareal coarse-grid operators for linear advection. *Numerical Linear Algebra with Applications*, 28(4), March 2021.

[22] Hans De Sterck, Robert D. Falgout, and Oliver A. Krzysik. Fast multigrid reduction-in-time for advection via modified semi-Lagrangian coarse-grid operators. *SIAM Journal on Scientific Computing*, 45(4):A1890–A1916, July 2023.

[23] Hans De Sterck, Stephanie Friedhoff, Alexander J. M. Howse, and Scott P. MacLachlan. Convergence analysis for parallel-in-time solution of hyperbolic systems. *Numerical Linear Algebra with Applications*, 27(1), November 2019.

[24] V. A. Dobrev, Tz. Kolev, N. A. Petersson, and J. B. Schroder. Two-level convergence theory for multigrid reduction in time (MGRIT). *SIAM Journal on Scientific Computing*, 39(5):S501–S527, January 2017.

[25] Jonathan Eckstein and Dimitri P. Bertsekas. On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1-3):293–318, April 1992.

[26] Matthew Emmett and Michael Minion. Toward an efficient parallel in time method for partial differential equations. *Communications in Applied Mathematics and Computational Science*, 7(1):105–132, March 2012.

[27] R. D. Falgout, S. Friedhoff, Tz. V. Kolev, S. P. MacLachlan, and J. B. Schroder. Parallel time integration with multigrid. *SIAM Journal on Scientific Computing*, 36(6):C635–C661, January 2014.

[28] R. D. Falgout, T. A. Manteuffel, B. O'Neill, and J. B. Schroder. Multigrid reduction in time for nonlinear parabolic problems: A case study. *SIAM Journal on Scientific Computing*, 39(5):S298–S322, January 2017.

[29] Gabriele Farina. Lecture 12: Gradient descent and descent lemmas. Lecture notes for MIT 6.7220 / 15.084: Nonlinear Optimization, February 2025.

[30] Hartmut Foerster, Klaus Stüben, and Ulrich Trottenberg. *Non-standard multigrid techniques using checkered relaxation and intermediate grids*, page 285–300. Elsevier, 1981.

[31] Stephanie Friedhoff. *Design and analysis of multigrid methods for parabolic problems*. PhD thesis, Tufts University, May 2014.

[32] Michael P. Friedlander and Paul Tseng. Exact regularization of convex programs. *SIAM Journal on Optimization*, 18(4):1326–1350, January 2008.

[33] Martin J. Gander. A waveform relaxation algorithm with overlapping splitting for reaction diffusion equations. *Numerical Linear Algebra with Applications*, 6(2):125–145, March 1999.

[34] Martin J. Gander. *50 Years of Time Parallel Time Integration*, page 69–113. Springer International Publishing, Cham, Switzerland, 2015.

[35] Martin J. Gander and Stefan Güttel. Paraexp: A parallel integrator for linear initial-value problems. *SIAM Journal on Scientific Computing*, 35(2):C123–C142, January 2013.

[36] Martin J. Gander, Laurence Halpern, and Frédéric Nataf. Optimal schwarz waveform relaxation for the one dimensional wave equation. *SIAM Journal on Numerical Analysis*, 41(5):1643–1681, January 2003.

[37] Martin J. Gander, Felix Kwok, and Bankim C. Mandal. Dirichlet–neumann waveform relaxation methods for parabolic and hyperbolic problems in multiple subdomains. *BIT Numerical Mathematics*, 61(1):173–207, August 2020.

[38] Martin J. Gander, Felix Kwok, and Julien Salomon. PARAOPT: A parareal algorithm for optimality systems. *SIAM Journal on Scientific Computing*, 42(5):A2773–A2802, January 2020.

[39] Martin J. Gander, Felix Kwok, and Hui Zhang. Multigrid interpretations of the parareal algorithm leading to an overlapping variant and MGRIT. *Computing and Visualization in Science*, 19(3-4):59–74, 2018.

[40] Martin J. Gander, Thibaut Lunet, Daniel Ruprecht, and Robert Speck. A unified analysis framework for iterative parallel-in-time algorithms. *SIAM Journal on Scientific Computing*, 45(5):A2275–A2303, 2023.

[41] Martin J. Gander and Stefan Vandewalle. Analysis of the parareal time-parallel time-integration method. *SIAM Journal on Scientific Computing*, 29(2):556–578, January 2007.

[42] Martin J. Gander, Shu-Lin Wu, and Tao Zhou. Time parallelization for hyperbolic and parabolic problems. *Acta Numerica*, 34:385–489, 2025.

[43] Bolin Gao and Lacra Pavel. On the properties of the softmax function with application in game theory and reinforcement learning, 2017. https://arxiv.org/abs/1704.00805.

[44] Guillaume Garrigos and Robert M. Gower. Handbook of convergence theorems for (stochastic) gradient methods, 2024. https://arxiv.org/abs/2301.11235.

[45] Pontus Giselsson and Stephen Boyd. Monotonicity and restart in fast gradient methods. In *53rd IEEE Conference on Decision and Control*, page 5058–5063. IEEE, December 2014.

[46] Gene H. Golub, Per Christian Hansen, and Dianne P. O'Leary. Tikhonov regularization and total least squares. *SIAM Journal on Matrix Analysis and Applications*, 21(1):185–194, January 1999.

[47] Gene H Golub and Charles F Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, 4 edition, February 2013.

[48] Robert M. Gower. Sketch and project: Randomized iterative methods for linear systems and inverting matrices, 2016. https://arxiv.org/abs/1612.06013.

[49] D. Guibert and D. Tromeur-Dervout. Parallel adaptive time domain decomposition for stiff systems of odes/daes. *Computers &; Structures*, 85(9):553–562, May 2007.

[50] Osman Güler. New proximal point algorithms for convex minimization. *SIAM Journal on Optimization*, 2(4):649–664, November 1992.

[51] Stefanie Günther, Lars Ruthotto, Jacob B. Schroder, Eric C. Cyr, and Nicolas R. Gauger. Layer-parallel training of deep residual neural networks. *SIAM Journal on Mathematics of Data Science*, 2(1):1–23, January 2020.

[52] Jens Hahne, Björn Polenz, Iryna Kulchytska-Ruchka, Stephanie Friedhoff, Stefan Ulbrich, and Sebastian Schöps. Parallel-in-time optimization of induction motors. *Journal of Mathematics in Industry*, 13(1), June 2023.

[53] Yunhui He and Scott MacLachlan. Two-level Fourier analysis of multigrid for higher-order finite-element discretizations of the laplacian. *Numerical Linear Algebra with Applications*, 27(3), February 2020.

[54] Andreas Hessenthaler, Ben S. Southworth, David Nordsletten, Oliver Röhrle, Robert D. Falgout, and Jacob B. Schroder. Multilevel convergence analysis of multigrid-reduction-in-time. *SIAM Journal on Scientific Computing*, 42(2):A771–A796, January 2020.

[55] Alexander J. Howse, Hans De Sterck, Robert D. Falgout, Scott MacLachlan, and Jacob Schroder. Parallel-in-time multigrid with adaptive spatial coarsening for the linear advection and inviscid burgers equations. *SIAM Journal on Scientific Computing*, 41(1):A538–A565, January 2019.

[56] Hamed Karimi, Julie Nutini, and Mark Schmidt. *Linear Convergence of Gradient and Proximal-Gradient Methods Under the Polyak-Łojasiewicz Condition*, page 795–811. Springer International Publishing, 2016.

[57] Donghwan Kim and Jeffrey A. Fessler. Adaptive restart of the optimized gradient method for convex optimization. *Journal of Optimization Theory and Applications*, 178(1):240–263, May 2018.

[58] O. A. Krzysik, H. De Sterck, R. D. Falgout, and J. B. Schroder. Parallel-in-time solution of hyperbolic PDE systems via characteristic-variable block preconditioning. *SIAM Journal on Scientific Computing*, 0(0):S337–S363, 0.

[59] Felix Kwok, Julien Salomon, and Djahou N. Tognon. Convergence of ParaOpt for general Runge–Kutta time discretizations. *SIAM Journal on Numerical Analysis*, 64(3):926–956, 2026.

[60] Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, January 2016.

[61] Huan Li and Zhouchen Lin. Restarted nonconvex accelerated gradient descent: No more polylogarithmic factor in the in the $o(\epsilon^{(-7/4)})$ complexity. *Journal of Machine Learning Research*, 24(157):1–37, 2023.

[62] Jacques-Louis Lions, Yvon Maday, and Gabriel Turinici. Resolution d'EDP par un schema en temps parareel. *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, 332(7):661–668, April 2001.

[63] P. L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16(6):964–979, 1979.

[64] Yvon Maday, Mohamed-Kamel Riahi, and Julien Salomon. *Parareal in Time Intermediate Targets Methods for Optimal Control Problems*, page 79–92. Springer Basel, Basel, Switzerland, 2013.

[65] Jan Mandel. A multilevel iterative method for symmetric, positive definite linear complementarity problems. *Applied Mathematics & Optimization*, 11(1):77–95, February 1984.

[66] O. L. Mangasarian. Sufficiency of exact penalty minimization. *SIAM Journal on Control and Optimization*, 23(1):30–37, January 1985.

[67] Thomas A. Manteuffel, Steffen Münzenmaier, John Ruge, and Ben Southworth. Nonsymmetric reduction-based algebraic multigrid. *SIAM Journal on Scientific Computing*, 41(5):S242–S268, January 2019.

[68] Thomas A. Manteuffel, John Ruge, and Ben S. Southworth. Nonsymmetric algebraic multigrid based on local approximate ideal restriction (ℓair). *SIAM Journal on Scientific Computing*, 40(6):A4105–A4130, January 2018.

[69] Michael Minion. A hybrid parareal spectral deferred corrections method. *Communications in Applied Mathematics and Computational Science*, 5(2):265–301, December 2010.

[70] Yurii Nesterov. A method for solving the convex programming problem with convergence rate $o(1/k^2)$. *Proceedings of the USSR Academy of Sciences*, 269:543–547, 1983.

[71] Yurii Nesterov. *Introductory Lectures on Convex Optimization*. Springer US, 2004.

[72] J. Nievergelt. Parallel methods for integrating ordinary differential equations. *Communications of the ACM*, 7(12):731–733, December 1964.

[73] Benjamin W. Ong and Jacob B. Schroder. Applications of time parallelization. *Computing and Visualization in Science*, 23(1–4), September 2020.

[74] Brendan O'Donoghue and Emmanuel Candès. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, 15(3):715–732, July 2013.

[75] Neal Parikh. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

[76] Boris T Polyak. *Introduction to optimization (translations series in mathematics and engineering)*. Optimization Software, May 1987.

[77] B.T. Polyak. Gradient methods for the minimisation of functionals. *USSR Computational Mathematics and Mathematical Physics*, 3(4):864–878, January 1963.

[78] B.T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, January 1964.

[79] James Renegar and Benjamin Grimmer. A simple nearly optimal restart scheme for speeding up first-order methods. *Foundations of Computational Mathematics*, 22(1):211–256, March 2021.

[80] Manfred Ries, Ulrich Trottenberg, and Gerd Winter. A note on MGR methods. *Linear Algebra and its Applications*, 49:1–26, February 1983.

[81] J-F Rodrigues. *Obstacle problems in mathematical physics*. North-Holland Mathematics Studies. North-Holland, May 2014.

[82] Orlando Romero, Mouhacine Benosman, and George J. Pappas. Ode discretization schemes as optimization algorithms. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, page 6318–6325. IEEE, December 2022.

[83] D. Ruprecht and R. Krause. Explicit parallel-in-time integration of a linear acoustic-advection system. *Computers & Fluids*, 59:72–83, April 2012.

[84] Ernest K. Ryu and Wotao Yin. *Large-Scale Convex Optimization: Algorithms & Analyses via Monotone Operators*. Cambridge University Press, November 2022.

[85] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, January 2003.

[86] Lisa Schönenberger and Hans-Georg Beyer. Optimal restart strategies for parameter-dependent optimization algorithms, 2025.

[87] V. V. Shaidurov. *Multigrid Methods for Finite Elements*. Springer Netherlands, 1995.

[88] Oleg V. Shylo and Oleg A. Prokopyev. *Restart Strategies*, page 1–16. Springer International Publishing, 2015.

[89] Ben S. Southworth. Necessary conditions and tight two-level convergence bounds for parareal and multigrid reduction in time. *SIAM Journal on Matrix Analysis and Applications*, 40(2):564–608, January 2019.

[90] Weijie Su, Stephen Boyd, and Emmanuel J. Candès. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. *Journal of Machine Learning Research*, 17(153):1–43, 2016.

[91] J.A.K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, June 1999.

[92] Yunpeng Tai. A survey of regression algorithms and connections with deep learning, 2021.
https://arxiv.org/abs/2104.12647.

[93] Giang Tran, Hayden Schaeffer, William M. Feldman, and Stanley J. Osher. An l^1 penalty method for general obstacle problems. *SIAM Journal on Applied Mathematics*, 75(4):1424–1444, January 2015.

[94] Ulrich Trottenberg, Cornelius W Oosterlee, and Anton Schuller. *Multigrid*. Academic Press, San Diego, CA, November 2000.

[95] Stefan Ulbrich. *Preconditioners Based on "Parareal" Time-Domain Decomposition for Time-Dependent PDE-Constrained Optimization*, page 203–232. Springer International Publishing, 2015.

[96] Richard S. Varga. *Matrix Iterative Analysis*. Springer Berlin Heidelberg, 2000.

[97] Radoslav Vuchkov, Eric C. Cyr, and Denis Ridzal. Multigrid-in-time preconditioners for KKT systems, 2024.
https://arxiv.org/abs/2405.04808.

[98] Chunxiao Wu and Justin W.L. Wan. Multigrid methods with Newton-Gauss-Seidel smoothing and constraint preserving interpolation for obstacle problems. *Numerical Mathematics: Theory, Methods and Applications*, 8(2):199–219, May 2015.

[99] Hongyuan Zha. Computing the generalized singular values/vectors of large sparse or structured matrix pairs. *Numerische Mathematik*, 72(3):391–417, January 1996.

APPENDIX

Appendix A

Schur complement

Definition 39 (Schur complement)**.** *Let* $M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \in \mathbb{R}^{(p+q)\times(p+q)}$ *be a block matrix with blocks* $A \in \mathbb{R}^{p\times p}$*,* $B \in \mathbb{R}^{p\times q}$*,* $C \in \mathbb{R}^{q\times p}$ *and* $D \in \mathbb{R}^{q\times q}$*. If* D *is invertible, we define*

$$M/D \coloneqq A - BD^{-1}C \tag{A.1}$$

as the Schur complement of the block D*. Similarly, if* A *is invertible, we define*

$$M/A \coloneqq D - CA^{-1}B \tag{A.2}$$

as the Schur complement of A*.*

In the following remarks, we discuss how the Schur complement arises from two different perspectives, both times showing how a closed form for the inverse of M, M^{-1}, may be obtained from the Schur complement.

Remark 47 (The Schur complement and linear systems of equations)**.** Let M be as defined in definition 39, and consider a linear system of equations given by

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}, \tag{A.3}$$

which is equivalent to the following linear system of equations:

$$A\mathbf{x} + B\mathbf{y} = \mathbf{u}, \tag{A.4a}$$
$$C\mathbf{x} + D\mathbf{y} = \mathbf{v}. \tag{A.4b}$$

Assume that A is invertible. Then, by (A.4a), we get

$$\mathbf{x} = A^{-1}(\mathbf{u} - B\mathbf{y}). \tag{A.5}$$

Substituting (A.5) into (A.4b) yields

$$(D - CA^{-1}B)\mathbf{y} = \mathbf{v} - CA^{-1}\mathbf{u}. \tag{A.6}$$

Substituting $M/A := D - CA^{-1}B$ from (A.2), and multiplying both sides by $(M/A)^{-1}$ yields

$$\mathbf{y} = (M/A)^{-1}\left(\mathbf{v} - CA^{-1}\mathbf{u}\right). \tag{A.7}$$

Substituting (A.7) into (A.4a) then yields

$$\mathbf{x} = \left(A^{-1} + A^{-1}B(M/A)^{-1}CA^{-1}\right)\mathbf{u} - A^{-1}B(M/A)^{-1}\mathbf{v}. \tag{A.8}$$

Then, by (A.5) and (A.7), we may express the vector of components $\mathbf{x}$ and $\mathbf{y}$ in terms of u and v by the matrix-vector product below:

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} A^{-1} + A^{-1}B(M/A)^{-1}CA^{-1} & -A^{-1}B(M/A)^{-1} \\ -(M/A)^{-1}CA^{-1} & (M/A)^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}. \tag{A.9}$$

By (A.3), we have that the right-hand side matrix above must be an inverse of M. That is, we find that

$$\begin{aligned} M^{-1} &= \begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} \\ &= \begin{bmatrix} A^{-1} + A^{-1}B(M/A)^{-1}CA^{-1} & -A^{-1}B(M/A)^{-1} \\ -(M/A)^{-1}CA^{-1} & (M/A)^{-1} \end{bmatrix} \\ &= \begin{bmatrix} I_p & -A^{-1}B \\ & I_q \end{bmatrix} \begin{bmatrix} A^{-1} & \\ & (M/A)^{-1} \end{bmatrix} \begin{bmatrix} I_p & \\ -CA^{-1} & I_q \end{bmatrix}. \end{aligned} \tag{A.10}$$

Similarly, by starting from (A.4b), we find M^{-1} in terms of M/D, the Schur complement of D (A.1).

Remark 48 (The Schur complement and Gaussian elimination)**.** Assume D is invertible such that the Schur complement of D, (A.1), is well-defined. Performing block-Gauss elimination on M, we first find

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \to \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} I_p & \\ -D^{-1}C & I_q \end{bmatrix} = \begin{bmatrix} A - BD^{-1}C & B \\ & D \end{bmatrix}. \tag{A.11}$$

Substituting (A.1) and performing another round of Gaussian elimination, we obtain

$$\begin{bmatrix} M/D & B \\ & D \end{bmatrix} \to \begin{bmatrix} I_p & -BD^{-1} \\ 0 & I_q \end{bmatrix} \begin{bmatrix} M/D & B \\ & D \end{bmatrix} = \begin{bmatrix} M/D & \\ & D \end{bmatrix}. \tag{A.12}$$

Therefore, we obtain the following decomposition for M in terms of the Schur complement M/D:

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} I_p & BD^{-1} \\ & I_q \end{bmatrix} \begin{bmatrix} M/D & \\ & D \end{bmatrix} \begin{bmatrix} I_p & \\ D^{-1}C & I_q \end{bmatrix}. \tag{A.13}$$

From the above block-matrix product, we can find M^{-1} in terms of M/D:

$$\begin{aligned} M^{-1} &= \begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} \\ &= \left(\begin{bmatrix} I_p & BD^{-1} \\ & I_q \end{bmatrix} \begin{bmatrix} (M/D)^{-1} & \\ & D \end{bmatrix} \begin{bmatrix} I_p & \\ D^{-1}C & I_q \end{bmatrix} \right)^{-1} \\ &= \begin{bmatrix} I_p & \\ -D^{-1}C & I_q \end{bmatrix} \begin{bmatrix} ((M/D)^{-1})^{-1} & 0 \\ 0 & D^{-1} \end{bmatrix} \begin{bmatrix} I_p & -BD^{-1} \\ 0 & I_q \end{bmatrix} \\ &= \begin{bmatrix} (M/D)^{-1} & -(M/D)^{-1}BD^{-1} \\ -D^{-1}C(M/D)^{-1} & D^{-1} + D^{-1}C(M/D)^{-1}BD^{-1} \end{bmatrix}. \end{aligned} \tag{A.14}$$

Similarly, if A is invertible, we can obtain a decomposition of M in terms of M/A (A.2), in terms of which we find another formulation of M^{-1}.