Imperial College London
Department of Computing

Multilevel Algorithms for the Optimization of Structured Problems

Chin Pang Ho

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Computing of Imperial College and
the Diploma of Imperial College, 2016

Declaration

I, Chin Pang Ho, declare that the research presented in this thesis is my own work, except where acknowledged. No part of this thesis has been submitted before for any degree or examination at this or any other university.

Abstract

Although large scale optimization problems are very difficult to solve in general, problems that arise from practical applications often exhibit particular structure. In this thesis we study and improve algorithms that can efficiently solve structured problems. Three separate settings are considered.

The first part concerns the topic of singularly perturbed Markov decision processes (MDPs). When a MDP is singularly perturbed, one can construct an aggregate model in which the solution is asymptotically optimal. We develop an algorithm that takes advantage of existing results to compute the solution of the original model. The proposed algorithm can compute the optimal solution with a reduction in complexity without any penalty in accuracy.

In the second part, the class of empirical risk minimization (ERM) problems is studied. When using a first order method, the Lipschitz constant of the empirical risk plays a crucial role in the convergence analysis and stepsize strategy of these problems. We derive the probabilistic bounds for such Lipschitz constants using random matrix theory. Our results are used to derive the probabilistic complexity and develop a new stepsize strategy for first order methods. The proposed stepsize strategy, Probabilistic Upper-bound Guided stepsize strategy (PUG), has a strong theoretical guarantee on its performance compared to the standard stepsize strategy.

In the third part, we extend the existing results on multilevel methods for unconstrained convex optimization. We study a special case where the hierarchy of models is created by approximating first and second order information of the exact model. This is known as Galerkin approximation, and we named the corresponding algorithm Galerkin-based Algebraic Multilevel Algorithm (GAMA). Three case studies are conducted to show how the structure of a problem could affect the convergence of GAMA.

Acknowledgements

I would like to express my deepest gratitude to my supervisor Dr. Panos Parpas for making this thesis possible. My journey of getting PhD admission was a long trek over 2010-2012. By August 2012, I already accepted that a single PhD offer is impossible for me, and contacting Panos was my last trial. To this day, I still remember the moment when Panos offered me a position: It was one day after I have bought my one-way ticket back to Hong Kong. I have to admit that I did once hesitate whether joining the computing department is suitable for me, but I am glad that I made the right decision and had my ticket refunded. With all other applications rejected, I am extremely grateful that Panos made the unpopular decision and believed that I could handle research in computational optimization, which I had no knowledge of. As an international student, it is impossible to not mention how thankful I am to Panos, for funding 3.5 years of my PhD studies with his two research grants, despite being a (very) junior faculty himself in 2012. Panos is my first teacher in optimization, and he has introduced many exciting research to me during our countless meetings. He has always encouraged me to enjoy doing research and do what I believe. He served as a great example for me too; his calmness and patience, especially when faced with emotional PhD student (i.e. me, and hopefully just me), are always something that I need to learn. Working with Panos is not easy though. During the iterative process of research, he has always put a sufficient condition[1] as a stopping criteria while I was aiming to use a necessary condition[2]. This four years have been arduous and uncomfortable, but yet interesting, challenging, memorable, and ruthlessly pushing the limit of my patience, knowledge, and sanity. By looking at the input-output ratio, I cannot resist questioning myself whether this precious opportunity from Panos was given to right person. This question perhaps cannot be answered with certainty, but in the following years of my career I hope to prove that it might not be a complete mistake.

I am heartily thankful to my second supervisor, Professor Berc Rustem, who is also the head of the Computational Optimization Group (COG). Berc has always tried to encourage me to enjoy my life as a PhD student as well as doing research. I truly appreciate it when he told me that the next chapter of life is not very far away when I was lost in the darkness with very dim streetlight.

[1] Sufficient conditions do not guarantee convergence.
[2] Necessary conditions do not guarantee optimality.

As a rookie in optimization, I was very fortunate to join the COG which comprises faculty members whose expertise covers a large spectrum of optimization. Dr. Daniel Kuhn, Dr. Ruth Misener, and Dr. Wolfram Wiesemann have enlightened me on stochastic optimization, global optimization, and robust optimization through seminars and discussion during lunch, group meetings, and pub drinks. They have also set up themselves as very good and perhaps unachievable examples for PhD students. Special thanks go to Wolfram, who has been giving me invaluable advice for my career plan, helping me with the empirical behavioral sensitivity analysis (in pubs), and inviting me to all of his activities during INFORMS Annual Meeting 2014; otherwise I would be totally lost in such a huge conference. I also highly appreciate the help from Wolfram, Daniel, Dr. Phebe Vayanos, and Professor Kalyan Talluri when I applied for the Imperial Junior Research Fellowship.

I would like to thank Professor Huifu Xu and Wolfram for their time spending on my thesis and viva, and I avoid trying to realize how much effort they put in to figure out all unnecessary typos and unclear proofs. I was extremely guilty when Professor Xu showed me what algebraic trick I used for the proof of Theorem 5.18.

It has been an eye-opening experience coming to the computing department, and I would like to thank my collaborators April Xi Chen, Liang Chen, and Yuanwei Li for inviting me to work in their fascinating PhD projects. I believe the true value of optimization comes from its practicality, and working with them has been an amazing experience. They have also been my dinner buddies, together with Luo Mai, Lukas Rupprecht, and Jeremy Riviere. Engaging in heated discussions with all of them is more than enjoyable, and gives me good exposure to many aspects of computer science. Additional thanks go to Yuanwei who has been a very understanding, helpful, and reliable flatmate.

It is very easy to feel isolated as a PhD student, but this does not apply to me because of my fellows: Radu Baltean-Lugojan, Juan Campos Salazar, Jeremy Cohen, Raul Castro Fernandez, Christos Gavriel, Micheal Hadjiyiannis, Layal Hakim, Grani Adiwena Hanasusanto, Sei Howe, Bidan Huang, Iakovos Kakouris, Eva Kalyvianaki, Alexandros Koliousis, Georgia Kouyialis, Miten Mistry, Dan O'Keeffe, Simon Olofsson, Andreas Pamboris, Zhan Qiu, Napat Rujeerapaiboon, Mali Shen, Christine Simpson, Jacintha Mack Smith, Quang Kha Tran, Weikun Wang, Pijika Watcharapichat, Poonam Yadav, Liang Zhao, and many others. I am not able to make a list with finite items to mention how much I have learned from all of them. I would like to

thank the supportive staffs: Amani El-Kholy, Teresa Ng, and Geoff Bruce. Many thanks and appreciation go to Ryan Vu Ngoc Duy Luong and Vahan Hovhannisyan for their encouragement during April 2015.

This separate paragraph is used to thank Vladimir Roitch, who has been extremely supportive with very few words. To show my respect to Vlad, here is my feeling: he is very nice![3]

I left Hong Kong in July 2006 as a way-below-average student, who was not able to continue A-level education in Hong Kong. The changes in these 10 years were dramatic and would not happen without the help from the following people:

- Teresa June Mun Mark: Teresa helped me when I applied to the U.S. high school exchange program in 2005-2006. It is clear that my English level was not well qualified for the program, and I would not have made it without her help.
- Janet and Guillermo Munoz: Janet and Guillermo hosted me in their family in Oxnard, CA, for my first three weeks in the U.S. in 2006, and they recommended "Clint" to be my American name. I am very thankful for the eight pages (if not more) long encouragement from Janet when I moved to Lott, TX.
- Suzanne Woodill and Brandy Glenn: Suzanne and Grandy were my host parents in Lott, TX, during the academic year 2006-2007. I am heartily thankful to them for giving me a year of difficult life and being perfect counterexamples in everything.
- Marilyn and Steve Holland, Kandy and Chris Nasso: They were the source of warmth during the year in Lott. I am grateful for everything they provided me with.
- Veronica and Art Ayres: Veronica and Art hosted me in their family in Port Angeles, WA, in 2007-2008. It was a very nice and warm experience, and one of the best years of my vagrant life.
- Professor Gerald N. Estberg: Jerry is a Professor Emeritus in the University of San Diego, who settled at Port Angeles for his retirement. I am very fortunate to be his research student during my second year of undergraduate. Jerry was my first teacher in programming and many other things in life. Because of Jerry, I am very honored and embarrassed

[3]Word count: 4. Characters excluding spaces: 13. Appreciation: Uncountable.

to be the (unofficial) third academic generation of Professor Kenneth G. Wilson, who is a world-famous physicist, Nobel Prize winner, and supercomputing pioneer. I also like to express my sincere gratitude to Jerry for providing countless recommendation letters when I was applying to PhD programs.

- Larry Smith: Larry is a wonderful math lecturer at Peninsula College who is the first person telling me that "operations research is very interesting".

- Dr. Thomas Richthammer and Dr. Yves van Gennip: Thomas and Yves are my favourite math lecturers and role models when I was in UCLA. I owe my sincere gratitude to them for providing countless recommendation letters when I was applying to PhD programs.

- Professor Radek Erban and Dr. Mark Flegg: Radek and Mark are my supportive MSc supervisors in Oxford, and they introduced to me to the fascinating research in mathematical biology and algorithms.

- Joe Chee Hoe Fong, Shuohao Liao, and Andrew Xiaodong Sui: They are my best friends and classmates at Oxford. It is hard to imagine how I could ever survive a year in Oxford without them.

- Stephen Yee Man Chung and Raymond Chi Kan Cheung: Stpehen and Raymond are my close friends who are long gone. They often remind me how fortunate I am whenever I feel frustrated. Raymond dreams to study computer science at UCLA, and I am pleased to partially fulfil his dream twice.

I would also like to thank my fiancee Sylvia Suet Yee Cheng and her family. Sylvia's family kindly offered to support my PhD studies without knowing what I was doing and where I was heading. I am deeply indebted to Sylvia, for all her love, support, and understanding over the last 7 years. I met Sylvia in 2007, when she was one of the very few people who actually believed in me. I cannot overstate my appreciation for all the colors that she has brought into my life, and I am looking forward to the following years that I can spend my life with her.

I don't know how to express my gratitude to my family for their unconditional tolerance, love, and upbringing. I am proud to be the (academic) role model of my sister, Hazel Wing Yung Ho, and I hope she can be humble and persistent on the way of pursuing her own dreams. I am grateful to my parents for providing this oversea experience. My mom, Mo Ching Ho,

has always encouraged me to do whatever I believe in. Without her encouragement, I doubt I would ever follow my heart to major in mathematics in the first place. Her calmness, tenacity, and tolerance are also something beyond my imagination. My dad, Sau Ming Ho, has been giving me invaluable advice for my future plans. He shows me how a diligent working attitude can change the fate of the entire family. His intelligence and visions are something that I can only dream to have. There is no doubt that these good genes were not passed on to me well, but nothing stops me from keep trying my very best, for everyone I care.

In dedication to my family for their unconditional love, trust, tolerance, and support.

'I do not know what I may appear to the world, but to myself I seem to have been only like a boy playing on the sea-shore, and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me.'

Sir Isaac Newton

Contents

List of Tables

List of Figures

Chapter 1

Introduction

I don't know anything, but I do know that everything is interesting if you go into it deeply enough.

Richard Feynman

1.1 Motivation and Objectives

Due to the complexity of practical applications, the ability to solve large scale optimization problems is crucial. Examples can be found across many disciplines: machine learning [SNW12], finance [Pri07], logistics [YCYA12], energy systems [PW14], molecular conformation [Wu96]. As one of the fundamental challenges in computational science, solving large scale optimization is computationally demanding, and much efforts have been made to reduce this burden.

In general, it is fair to say that solving an arbitrary optimization problem could be very difficult, or it simply could not be done [BV04]. Problems arising from practical applications, however, often have particular structure. In what follows, we will discuss several important applications and their underlying structure.

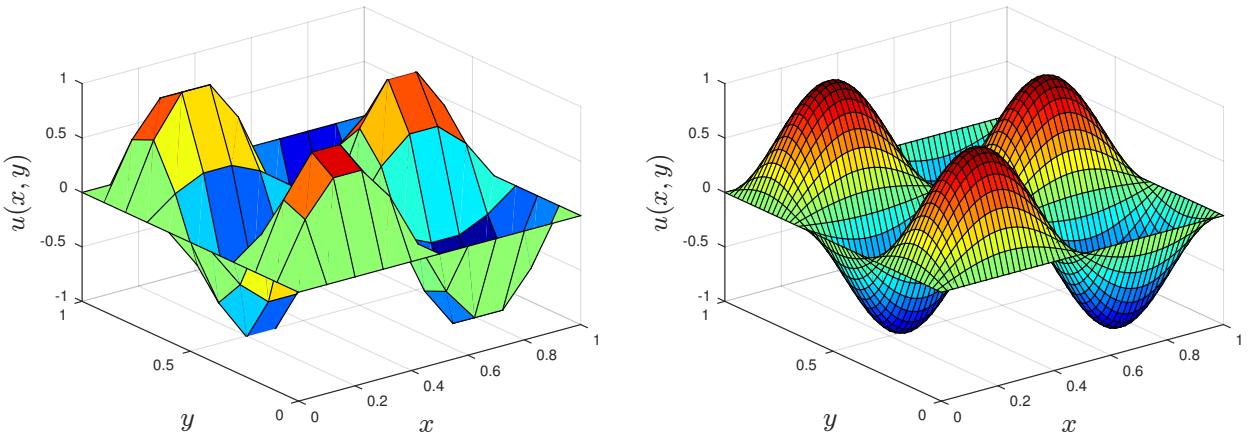


Figure 1.1: The solutions, $u(x, y)$'s, of the Poisson's equation (1.1) with different mesh sizes.

Geometric Structure

Many optimization problems exhibit geometric structure, in which the geometry of the solution can be approximated.

One classical example is the class of infinite-dimensional optimization problems, which include many problems in optimal control. These problems usually could not be solved exactly, and one needs to approximate and discretize the original problem using finite differences or finite elements. The dimension of the discretized problem depends on the mesh size during the discretization. In general, smaller mesh size results in higher dimensional problems, although the solution would be more accurate. Figure 1.1 shows an example using a two-dimensional Poisson's equation,

$$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = 13\pi^2 \sin(2\pi x)\sin(3\pi y), \quad \text{in} \quad \Omega = [0,1]^2, \tag{1.1}$$

where,

$$u = 0, \quad \text{on} \quad \partial\Omega.$$

Notice that the "two-dimensional" in (1.1) refers to the dimensions of the continuous variables in the original Poisson's equation, i.e. x and y. Once the Poisson's equation is discretized, the decision variable of the corresponding optimization problem is not in two dimensions, and the number of dimensions is the same as the number of grid points on the chosen mesh. Figure

1.1 shows the solutions of a Poisson's equation using different mesh sizes. One can see that although using a large mesh size would yield an inaccurate solution, the geometry of the solution is highly similar to the solution which is computed by using a smaller mesh size. Thus, the geometric structure of the solution is preserved across the different mesh sizes.

Another example can be found in image processing. For natural images, it is expected that the same image with different resolutions would have the same structure for every small region of the image. That is, neighbouring pixels usually have similar image intensities. Therefore, one can use a high resolution image to construct a low resolution image that is visually similar. For applications such as image de-blurring, images with lower resolution would yield an optimization model that is in lower dimension [PLRR].

Multiscale Structure

Apart from geometric structure, many practical applications on complex systems present multiscale structure, i.e. there exists an important feature in which its magnitude spans across multiple scales, such as time and space scales.

One important example could be found in energy systems, where many long term sequential decisions are made based on the dynamics of an environment. This environment depends on the long term decisions made as well as the short-term operations such as unit commitment and generation dispatch. The interactions between long term environment and short term operations display a multiple time scales dynamics, i.e. dynamics that evolve significantly in both short and long time horizons [PW14].

The resource allocation problem in cloud provisioning is another example [MKC13, KRC+15]. Cloud provisioners often receive job requests that have needs for a certain amount of cloud resource within a time period. These requests could ask for cloud resource that ranges from the scale of 10 units to 10000 units, and so the resource requirement of this problem exhibits a multiscale structure. On the other hand, the deadlines of these job requests also have multiscale structure since they could be in the range of seconds to hours, or even days.

Statistical Structure

Consider an example in stochastic optimization,

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathbb{E}_{\mathbb{Q}}[f(\mathbf{x}, \boldsymbol{\xi})], \tag{1.2}$$

where $\boldsymbol{\xi} \in \mathbb{R}^P$ is a random vector which follows the distribution $\mathbb{Q}$, and f is convex in $\mathbf{x}$. Therefore, the above optimization model minimizes the expected value of f with decision variable $\mathbf{x}$. In practice, however, the distribution $\mathbb{Q}$ is often unknown. Instead, one is usually given a sample of data, i.e. $\{\boldsymbol{\xi}_i\}_{i=1}^m$. In such cases, one common way is to solve the sample average of the above optimization model, i.e.

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{x}), \tag{1.3}$$

where $f_i(\mathbf{x}) = f(\mathbf{x}, \boldsymbol{\xi}_i)$, for $i = 1, 2, \ldots, m$.

Equation (1.3) is called the Sample Average Approximation (SAA), and SAA has a long history of developments. See [BL11, KSHdM02] for more details. One famous example of using SAA is the empirical risk minimization, which is a general form of many popular regression models, including linear regression and logistics regression.

Notice that SAA (1.3) displays a unique statistical and mathematical structure. Firstly, SAA always has the form of a sum of functions. Secondly, the difference between each f_i is due to the data points $\boldsymbol{\xi}_i$'s, in which they follow the same probability distribution. These two features of SAA have motivated a lot of development in optimization algorithms, including stochastic gradient descent [Bot12, PJ92, TH12, Bot98] and mini-batch algorithms [CR16].

In this thesis, we aim to take advantages of problem structure to advance computational performance of optimization algorithms. The structure of this thesis will be provided in the next section.

1.2 Thesis Outline and Contributions

We consider optimization problems with different structures, and develop efficient algorithms based on this additional information. The problems we consider cover a large spectrum of interesting applications, ranging from stochastic optimal control to machine learning to infinite-dimensional optimization. Except for Section 4, the main approach used is multilevel optimization methods, which follow the idea of multigrid methods for solving (non-)linear equations of discretizations arising from partial differential equations. This thesis expands the capabilities and knowledge in the field of multilevel optimization methods.

Apart from Chapter 2 and Chapter 6, which we provide the background materials and the conclusions, this thesis is divided into three parts. Each part corresponds to a class of structured optimization problems.

In Chapter 3, we consider singularly perturbed Markov Decision Processes (MDPs), which exhibit multiple time-scale structure. An existing result shows that a singularly perturbed MDP could be approximated by an aggregate model. The solution of the aggregate model was shown to be asymptotically optimal [YZ13]. By making use of this result, a multilevel algorithm is developed by replacing some parts of the computation using the coarse model. We show that the complexity of the proposed algorithm is superior to the standard value iteration for this class of problems. The contents of this chapter appeared in the following paper:

1. C. P. Ho, and P. Parpas. *Singularly Perturbed Markov Decision Processes: A Multiresolution Algorithm.* SIAM Journal on Control and Optimization 52:6, 3854-3886, 2014.

In Chapter 4, we consider the computational complexity and stepsize strategy of regularized empirical risk minimization problems. The worst-case complexity for this problem follows from standard results in convex optimization theory [BT09, Nes15]. Some algorithms in this class of problems are considered to be "dimension-free" because the convergence analysis of these algorithms is independent of the size of the problem. This above argument is based on the assumption that the Lipschitz constant of the problem is independent of the dimensionality. We

show that the dimensionality of the model is, however, hidden within the Lipschitz constant. Standard random matrix theory is used to derive the probabilistic bounds of the Lipschitz constant. The derived bounds are also used to develop a stepsize strategy for better computational performance. The contents of this chapter are currently being prepared for publication with the following working title:

2. C. P. Ho, and P. Parpas. *Empirical Risk Minimization: Probabilistic Complexity and Stepsize Strategy.* In preparation.

In Chapter 5, we consider the unconstrained convex optimization problem. We provide a broader view on the general multilevel framework, and we show a connection between this framework and standard optimization methods. A special case of this general framework is further studied, and we call it **G**alerkin-based **A**lgebraic **M**ultilevel **A**lgorithm (GAMA). The Galerkin model is highly related to algebraic multigrid methods, in which the hierarchy of the models is generated by using the algebraic information of the models instead of using the geometric structure. In the view of optimization, GAMA is equivalent to performing Newton's method in reduced dimensions. We prove that GAMA has a local rate of composite convergence, which is a linear combination of linear and quadratic convergence. By considering three case studies, we show how the structure of the problems could affect the convergence of multilevel methods. The contents of this chapter are currently being prepared for publication with the following working title:

3. C. P. Ho, and P. Parpas. *Multilevel Optimization Methods: Convergence and Problem Structure.* In preparation.

During the 4 years of doctoral studies, I was fortunate to have opportunities to collaborate with different researchers and PhD students at Imperial College London. Below is a list of publications which do not directly contribute to this thesis.

4. X. Chen, C. P. Ho, R. Osman, P. Harrison, and W. Knottenbelt. *Understanding, Modelling and Improving the Performance of Web Applications in Multi-core Virtualised Envi-*

ronments. Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering (ICPE) 197-207, 2014.

5. C. P. Ho, and P. Parpas. *On Using Spectral Graph Theory to Infer the Structure of Multiscale Markov Processes.* 2015 Proceedings of the Conference on Control and its Applications 228-235, 2015.

6. L. Chen, T. Tong, C. P. Ho, R. Patel, D. Cohen, A. C. Dawson, O. Halse, O. Geraghty, P. E.M. Rinne, C. J. White, T. Nakornchai, P. Bentley, and D. Rueckert. *Identification of Cerebral Small Vessel Disease Using Multiple Instance Learning.* Medical Image Computing and Computer-Assisted Intervention MICCAI 2015, Springer International Publishing, 9349, 523-530, 2015.

7. Y. Li, C. P. Ho, N. Chahal, R. Senior, and M.-X. Tang. *Myocardial Segmentation of Contrast Echocardiograms Using Random Forests Guided by Shape Model.* Accepted for Medical Image Computing and Computer-Assisted Intervention MICCAI, 2016.

Chapter 2

Background Theory

It is not knowledge, but the act of learning, not possession but the act of getting there, which grants the greatest enjoyment.

Carl Friedrich Gauss

In this chapter, we will provide background material for the thesis. The chapter is divided into three sections.

In the first section, we review first order and second order algorithms which solve unconstrained continuous convex optimization problems. We consider the conventional setting that the objective function is (twice) continuously differentiable and Lipschitz continuous, and we introduce four classical algorithms: gradient descent, block-coordinate descent, Newton's method, and quasi-Newton methods. We also consider the composite convex program, in which the objective function is a sum of a continuously differentiable function and a simple function. The definition of simple function will be provided later in the section. One state-of-the-art algorithm for such problems is known as "FISTA", which stands for Fast Iterative Shrinkage-Thresholding Algorithm. Algorithmic details and theoretical performance of all five algorithms will be reviewed.

In the second section, some existing results in machine learning and statistics are provided, and these results will be used later in this thesis. We first summarize some concentration bounds in both random variables and random matrices. We then introduce the Nyström method, which is used to compute low rank approximations of positive semi-definite matrices.

In the third section, we review background materials for Markov decision processes (MDPs). We cover the basis of continuous-time Markov chains (CTMCs), Markov decision processes (MDPs), and the two common computational methods for MDPs: value iteration and linear programming.

We emphasize that each topic covered in this chapter has its own long history in research, and this chapter is far from a complete review of all of them. The goal of this chapter is to cover necessary knowledge that will lead to a smoother reading experience for the rest of this thesis. The material presented in this chapter is based on [BT09, BT13, BV04, Ros06, Pow11, Tro15, PGD+15, Git11, NW06].

2.1 Unconstrained Continuous Convex Optimization

In this section we are interested in the unconstrained continuous convex program,

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}), \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function. Note that we require different additional properties of a convex function at different stages of this thesis. Below is a list of common properties.

Definition 2.1 *A continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is said to have a L-Lipschitz continuous gradient if*

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n.$$

Suppose f is also twice continuously differentiable. Then the above definition is equivalent to

$$-L\mathbf{I} \preceq \nabla^2 f(\mathbf{x}) \preceq L\mathbf{I}, \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

Definition 2.2 *A twice continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is said to have a M-Lipschitz continuous Hessian if*

$$\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})\| \leq M\|\mathbf{x} - \mathbf{y}\|, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n.$$

Definition 2.3 *A convex differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be strongly convex if there exists a positive constant μ such that for each $\mathbf{x} \in \mathbb{R}^n$,*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + \mu\|\mathbf{y} - \mathbf{x}\|^2, \quad \forall \mathbf{y} \in \mathbb{R}^n.$$

Suppose f is also twice continuously differentiable. Then the above definition is equivalent to

$$\mu\mathbf{I} \preceq \nabla^2 f(\mathbf{x}), \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

2.1.1 First Order Methods

We begin with the introduction of first order methods. Starting with the case that the objective function is differentiable and has a L-Lipschitz continuous gradient, gradient descent method and block-coordinate descent method are introduced. We then consider objective functions that have the form of a composite function, i.e. a sum of a differentiable function and a (non-smooth) simple function. In such setting, we introduce one standard first order method - Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [BT09]. FISTA can be seen as an extension of gradient descent method with two main differences: (i) Nesterov's acceleration techniques [Nes04] is applied for FISTA. (ii) It accommodates for composite functions.

Gradient Descent

Consider the objective function

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}), \tag{2.2}$$

where f is a continuously differentiable function and has L-Lipschitz continuous gradient. Gradient descent is an iterative method for (2.2). For any initial guess $\mathbf{x}_0$, it updates the incumbent by

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k, \quad k = 0, 1, 2, \ldots,$$

where $\alpha_k \in \mathbb{R}$ and $\mathbf{d}_k \in \mathbb{R}^n$ are the stepsize and direction at the k^{th} iteration, respectively.

The idea of gradient descent is to use the negative gradient as the direction,

$$\mathbf{d}_k = -\nabla f(\mathbf{x}_k),$$

and this particular choice of direction is a descent direction, since

$$\nabla f(\mathbf{x}_k)^T \mathbf{d}_k = -\|\nabla f(\mathbf{x}_k)\|^2 < 0, \quad \forall \nabla f(\mathbf{x}_k) \neq 0.$$

There are many methods to compute stepsize α_k. When L is known or could be estimated, then one choice could be $\alpha_k = 1/L$ since

$$\begin{aligned} f\left(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)\right) &\leq f(\mathbf{x}_k) + \nabla f(\mathbf{x}_k)^T \left(\frac{-1}{L}\nabla f(\mathbf{x}_k)\right) + \frac{L}{2}\left\|\frac{1}{L}\nabla f(\mathbf{x}_k)\right\|^2 \\ &= f(\mathbf{x}_k) - \frac{1}{L}\|\nabla f(\mathbf{x}_k)\|^2 + \frac{1}{2L}\|\nabla f(\mathbf{x}_k)\|^2 \\ &= f(\mathbf{x}_k) - \frac{1}{2L}\|\nabla f(\mathbf{x}_k)\|^2. \end{aligned}$$

That is, the next incumbent $\mathbf{x}_{k+1}$ has a smaller function value as long as $\nabla f(\mathbf{x}_k) \neq 0$. However, in many cases L is unknown, and one needs to use a large enough constant $\tilde{L}_k$ such that,

$$f\left(\mathbf{x}_k + \frac{1}{\tilde{L}_k}\mathbf{d}_k\right) \leq f(\mathbf{x}_k) - \frac{1}{2\tilde{L}_k}\|\nabla f(\mathbf{x}_k)\|^2. \tag{2.3}$$

Algorithm 2.1 Gradient descent

Input parameters: Initial guess $\mathbf{x}_0 \in \mathbb{R}^n$, $\eta > 1$. Choice of Option 1 or 2.
for $k = 1, 2, \cdots$ **do**
Compute the direction $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$.
Select α_k using

- Option 1: $\alpha_k = 1/L$.
- Option 2: Find the smallest $q \in \mathbb{N}$ such that for $\tilde{L}_k = \eta^q \tilde{L}_{k-1}$, $\tilde{L}_k$ satisfies (2.3). Set $\alpha_k = 1/\tilde{L}_k$.

Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$.
end for

Algorithm 2.1 provides the details of gradient descent method. The following theorem states the theoretical performance of gradient descent.

Theorem 2.4 ([Nes04]) *Suppose Algorithm 2.1 is performed, then*

$$f(\mathbf{x}_k) - f(\mathbf{x}_\star) \leq \frac{\hat{L}\|\mathbf{x}_0 - \mathbf{x}_\star\|^2}{2k},$$

where $\hat{L} = L$ if Option 1 is chosen, and $\hat{L} = \eta L$ if Option 2 is chosen.

From Theorem 2.4, one can see the function value converges to the minimum at a rate in $\mathcal{O}(1/k)$. We emphasize that gradient descent method is not the best among its kind. In particular, Nesterov proposed the optimal scheme, which accelerates gradient descent method to the rate in $\mathcal{O}(1/k^2)$ [Nes04].

Block-coordinate Descent

When using Algorithm 2.1 to solve (2.2), gradient evaluations and vector operations are needed. However, these operations could be computationally expensive or even intractable for large scale problems. To this end, block-coordinate descent methods were proposed to relax this computational burden. Similar to gradient descent method, block-coordinate descent can be

accelerated using the Nesterov's acceleration techniques [BT13]. However, for the purpose of this thesis, we shall focus on the non-accelerated version.

The basic idea of block-coordinate descent is to decompose vector operations. We denote matrices $\mathbf{U}_i \in \mathbb{R}^{n\times n_i}$, $i = 1, \ldots, p$, for which

$$[\mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_p] = \mathbf{I},$$

where n_i's are positive integers such that $\sum_{i=1}^{p} n_i = n$. The above notation considers a division of p blocks, and each i represents the i^{th} block. Using the notation of $\mathbf{U}_i$, we define the i^{th} block of the gradient

$$\nabla_i f(\mathbf{x}) \triangleq \mathbf{U}_i^T \nabla f(\mathbf{x}).$$

For block-coordinate descent, we further assume that f is block-wise Lipschitz continuous, i.e. there exist constants L_i, $i = 1, 2, \ldots, p$ such that

$$\|\nabla_i f(\mathbf{x} + \mathbf{U}_i\mathbf{h}_i) - \nabla_i f(\mathbf{x})\| \leq L_i\|\mathbf{h}_i\|, \quad \forall \mathbf{h}_i \in \mathbb{R}^{n_i}.$$

Following the above notation, we can define the i^{th} block of gradient update in the k^{th} iteration

$$\mathbf{x}_k^i = \mathbf{x}_k^{i-1} - \frac{1}{L_i}\mathbf{U}_i\nabla_i f(\mathbf{x}_k^{i-1}).$$

The above update uses the stepsize $1/L_i$. When L_i is unknown, one can apply the same technique as in gradient descent, i.e. finding a large enough $\tilde{L}_i$ such that

$$f\left(\mathbf{x}_k^{i-1} - \frac{1}{\tilde{L}_i}\mathbf{U}_i\nabla_i f(\mathbf{x}_k^{i-1})\right) \leq f(\mathbf{x}_k^{i-1}) - \frac{1}{2\tilde{L}_i}\|\nabla_i f(\mathbf{x}_k^{i-1})\|^2. \tag{2.4}$$

Algorithm 2.1 states the algorithmic procedure of block-coordinate descent method. The theoretical performance is provided in the following theorem, and it uses following notations.

$$\mathcal{X}_\star \triangleq \left\{\mathbf{x}_\star : \mathbf{x}_\star \in \arg\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})\right\}, \quad \text{and} \quad \mathcal{R}(\mathbf{x}_0) \triangleq \max_{\mathbf{x}\in\mathbb{R}^n}\max_{x_\star\in\mathcal{X}_\star}\{\|\mathbf{x} - \mathbf{x}_\star\| : f(\mathbf{x}) \leq f(\mathbf{x}_0)\}.$$

Algorithm 2.2 Block-coordinate gradient descent

Input parameters: Initial guess $\mathbf{x}_0 \in \mathbb{R}^n$, (For option 2: $L_j^0, j = 1, \ldots, p$). Choice of Option 1 or 2.

for $k = 1, 2, \cdots$ **do**

Set $\mathbf{x}_k^0 = \mathbf{x}_k$ and update recursively

$$\mathbf{x}_k^i = \mathbf{x}_k^{i-1} - \alpha_k^i \mathbf{U}_i \nabla_i f(\mathbf{x}_k^{i-1}), \quad i = 1, \ldots, p,$$

where α_k^i is computed by

- Option 1: $\alpha_k^i = 1/L_i$
- Option 2: Find the smallest $q \in \mathbb{N}$ such that for $\tilde{L}_i = \eta^q \tilde{L}_i^0$, $\tilde{L}_i$ satisfies (2.4). Set $\alpha_k = 1/\tilde{L}_i$.

Set $\mathbf{x}_{k+1} = \mathbf{x}_k^p$.

end for

Theorem 2.5 ([BT13]) *Suppose Algorithm 2.2 is performed. Then for* $k = 0, 1, \ldots,$

$$f(\mathbf{x}_k) - f(\mathbf{x}_\star) \leq \begin{cases} 4L_{\max}(1 + p^3\kappa^2)\mathcal{R}^2(\mathbf{x}_0)\dfrac{1}{k + (8/p)} & \text{if Option 1,} \\ 4\eta L_{\max}(1 + pL^2/(L_{\min}^0)^2)\mathcal{R}^2(\mathbf{x}_0)\dfrac{1}{k + (8/p)} & \text{if Option 2,} \end{cases}$$

where $L_{\max} = \max_i L_i$, $\kappa = (\max_i L_i)/(\min_i L_i)$, $L_{\min}^0 = \min_i L_i^0$.

From Theorem 2.5, one can see the drawback of block-coordinate descent method because p is inversely proportional to the performance of the algorithm. That is, the more blocks we make, the worse rate of convergence block-coordinate descent would have. Therefore, for problems in which vector operations and gradient evaluations are not computationally expensive, gradient descent would be preferable.

FISTA

We now let the objective function to be in the form of a composite function. That is,

$$\min_{\mathbf{x} \in \mathbb{R}^n} \{F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})\}, \tag{2.5}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function with L-Lipschitz continuous gradient, and $g : \mathbb{R}^n \to \mathbb{R}$ is a continuous convex function which is possibly nonsmooth but simple. The definition of simple function means that it results in computationally inexpensive proximal projection steps, which will be formally defined later in this section.

One of the standard optimization algorithms for (2.5) is FISTA. FISTA is a modification of ISTA, Iterative Shrinkage-Thresholding Algorithm, with the additional Nesterov's acceleration technique applied [Nes04]. For both FISTA and ISTA, the proximal projection step is taken in each iteration

$$p_L(\mathbf{y}) \triangleq \arg\min\{Q_L(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in \mathbb{R}^n\},$$

where

$$Q_L(\mathbf{x}, \mathbf{y}) \triangleq f(\mathbf{y}) + \langle \mathbf{x} - \mathbf{y}, \nabla f(\mathbf{y}) \rangle + \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2 + g(\mathbf{x}).$$

We point out the the proximal projection step admits a unique minimizer. Recall that g is assumed to be a simple function. By simple we mean that the choice of g would yield to cheap computation in the "arg min" procedure at the proximal step $p_L(\cdot)$. One special case of g is the weighted ℓ_1 norm. Suppose $g(\mathbf{x}) = \omega\|\mathbf{x}\|_1$ with positive constant ω, the proximal step would become

$$p_L(\mathbf{y}) = \mathcal{T}_{\omega/L}\left(\mathbf{x} - \frac{1}{L}\nabla f(\mathbf{x})\right),$$

where $\mathcal{T}_\alpha$ is called the shrinkage operator, and

$$\mathcal{T}_\alpha(\mathbf{x})_i = (|x_i| - \alpha)_+ \mathrm{sgn}(x_i).$$

One can see that in this special case the proximal step is no more than a usual gradient descent step plus the shrinkage operator, which is computationally inexpensive.

The details of FISTA with constant stepsize are provided in Algorithm 2.3. Notice that Algorithm 2.3 requires the Lipschitz constant L. When L is not known, one has to ensure at each iteration, a large enough $\tilde{L}$ is chosen. In particular, at the k^{th} iteration with incumbent $\mathbf{x}_k$, $\tilde{L}$

Algorithm 2.3 FISTA with constant stepsize

Input parameters: Lipschitz constant L, initial guess $\mathbf{x}_0 \in \mathbb{R}^n$.
Initialization: Set $\mathbf{y}_1 = \mathbf{x}_0$ and $t_1 = 1$.
for $k = 1, 2, \cdots$ **do**

$$\begin{aligned} \mathbf{x}_k &= p_L(\mathbf{y}_k), \\ t_{k+1} &= \frac{1+\sqrt{1+4t_k^2}}{2}, \\ \mathbf{y}_{k+1} &= \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}_k - \mathbf{x}_{k-1}). \end{aligned}$$

end for

Algorithm 2.4 FISTA with backtracking

Input parameters: Initial guess $\mathbf{x}_0 \in \mathbb{R}^n$, $L_0 > 0$, $\eta > 1$
Initialization: Set $\mathbf{y}_1 = \mathbf{x}_0$ and $t_1 = 1$.
for $k = 1, 2, \cdots$ **do**
Find the smallest $q \in \mathbb{N}$ such that for $\tilde{L} = \eta^q L_{k-1}$,

$$F(p_{\tilde{L}}(\mathbf{x}_k)) \leq Q_{\tilde{L}}(p_{\tilde{L}}(\mathbf{x}_k), \mathbf{x}_k).$$

Set $L_k = \eta^q L_{k-1}$ and compute

$$\begin{aligned} \mathbf{x}_k &= p_{L_k}(\mathbf{y}_k), \\ t_{k+1} &= \frac{1+\sqrt{1+4t_k^2}}{2}, \\ \mathbf{y}_{k+1} &= \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}_k - \mathbf{x}_{k-1}). \end{aligned}$$

end for

needs to be large enough to satisfy

$$F(p_{\tilde{L}}(\mathbf{x}_k)) \leq Q_{\tilde{L}}(p_{\tilde{L}}(\mathbf{x}_k), \mathbf{x}_k).$$

Algorithm 2.4 provides the details of FISTA with backtracking stepsize strategy. We should mention that the stepsize strategy in Algorithm 2.4 is the original approach proposed in [BT09], but not the only one. See for example [Nes15].

Theorem 2.6 *Suppose Algorithm 2.3 or 2.4 is performed. Then for any $k \geq 1$,*

$$F(\mathbf{x}_k) - F(\mathbf{x}_\star) \leq \frac{2\hat{L}\|\mathbf{x}_0 - \mathbf{x}_k\|^2}{(k+1)^2},$$

where $\mathbf{x}_\star = \arg\min_{\mathbf{x}} F(\mathbf{x})$, $\hat{L} = L$ if the constant stepsize strategy is chosen, and $\hat{L} = \eta L$ if the backtracking stepsize strategy is chosen. η is the user-defined parameter in Algorithm 2.4.

Both versions of FISTA have optimal theoretical performance guarantees. Compared to the non-accelerated version ISTA which has the rate $\mathcal{O}(1/k)$, FISTA converges with the rate $\mathcal{O}(1/k^2)$, as stated in Theorem 2.6.

We emphasize that research in first order algorithms is a popular topic. Gradient descent, block-coordinate descent, and FISTA are just three standard algorithms. We refer readers to [RT16, HPZ15, DBLJ14, HL15, Nes04, Nes15, LPRR16, BTMN01] for the developments on this line of research, including Nesterov's acceleration technique, mirror descent, and parallel coordinate descent.

2.1.2 Second Order Methods

In the rest of this section, some background material on second order algorithms is provided. We solely consider second order methods that solve problems in the following form

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}), \tag{2.6}$$

where f is a twice continuously differentiable function. We further assume that f is a strongly convex function with parameter μ, and f has L-Lipschitz continuous gradient and M-Lipschitz continuous Hessian.

The above setting is more restrictive compared to (2.2) because of the extra assumptions on strong convexity and twice differentiability. Countless research studies have been conducted on solving (2.6), and it is impossible to mention every one of them. We refer readers to

[DM77, DES82, Ber95, NW06] for more details. However, it is fair to say that a good portion of those algorithms are variants of Newton's method. In what follows, we will provide the details of Newton's method and one of its variants, quasi-Newton method.

Newton's method

Similar to the gradient and block-coordinate descent, Newton's method is an iterative method. The core idea of Newton's method is based on second order approximation of f at the current incumbent $\mathbf{x}_k$, i.e.

$$f(\mathbf{x}_k + \mathbf{d}) \approx f(\mathbf{x}_k) + \langle \nabla f(\mathbf{x}_k), \mathbf{d} \rangle + \frac{1}{2}\mathbf{d}^T \nabla^2 f(\mathbf{x}_k)\mathbf{d},$$

and the direction $\mathbf{d}_k$ is computed by minimizing the right hand side of the above equation over $\mathbf{d}$. Equivalently,

$$\mathbf{d}_k = -[\nabla^2 f(\mathbf{x}_k)]^{-1} \nabla f(\mathbf{x}_k).$$

There are many methods to compute stepsize α_k, but for the purpose of this chapter, we only consider the Armijo's rule, i.e., we require that α_k satisfies

$$f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) \leq f(\mathbf{x}_k) + \rho_1 \alpha_k \nabla f(\mathbf{x}_k)^T \mathbf{d}_k. \tag{2.7}$$

where $\rho_1 \in (0, 0.5)$ is a user-defined parameter. The Armijo's rule ensures the stepsize yields a sufficient reduction in function value. In particular, the next function value $f(\mathbf{x}_{k+1})$ must be less than $f(\mathbf{x}_k)$, since $\mathbf{d}_k$ is a descent direction and $\nabla f(\mathbf{x}_k)^T \mathbf{d}_k \leq 0$.

Algorithm 2.5 is one standard version of Newton's method using Armijo's rule as stepsize strategy. In the following two theorems, we state the theoretical performance of Newton's method.

Theorem 2.7 ([BV04]) *Suppose Algorithm 2.5 is performed and* $\|\nabla f(\mathbf{x}_k)\| \geq \eta$, *for some* $\eta > 0$, *then*

$$f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k) \leq -\rho_1 \beta_{ls} \eta^2 \frac{\mu}{L^2}.$$

Algorithm 2.5 Newton's method with Armijo's rule

Input parameters: Initial guess $\mathbf{x}_0 \in \mathbb{R}^n$, $\rho_1 \in (0, 1/2)$, $\beta_{ls} \in (0, 1)$
for $k = 1, 2, \cdots$ **do**
Compute the direction

$$\mathbf{d}_k = -[\nabla^2 f(\mathbf{x}_k)]^{-1} \nabla f(\mathbf{x}_k).$$

Find the smallest $q \in \mathbb{N}$ such that for $\alpha_k = \beta_{ls}^q$,

$$f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) \leq f(\mathbf{x}_k) + \rho_1 \alpha_k \nabla f(\mathbf{x}_k)^T \mathbf{d}_k.$$

Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$.
end for

Theorem 2.8 ([BV04]) *Suppose Algorithm 2.5 is performed and* $\|\nabla f(\mathbf{x}_k)\| \leq 3(1 - 2\rho_1)\frac{\mu^2}{M}$, *then*

$$\|\nabla f(\mathbf{x}_{k+1})\| \leq \frac{M}{2\mu^2} \|\nabla f(\mathbf{x}_k)\|^2.$$

Theorem 2.7 and 2.8 describe the performance of Newton's method at different stages. Suppose when the current incumbent $\mathbf{x}_k$ is far from the solution $\mathbf{x}_\star$, Theorem 2.7 guarantees that each iteration of Newton's method would result in a reduction in function value, and thus moving closer to $\mathbf{x}_\star$. This stage is called the damped Newton phase. Once the current incumbent $\mathbf{x}_k$ is sufficiently close to $\mathbf{x}_\star$, i.e. $\|\nabla f(\mathbf{x}_k)\|$ is sufficiently small, Theorem 2.8 guarantees that the norm of the gradient would be reduced quadratically in each iteration. This stage of the iterative process is called the quadratically convergent phase.

We emphasize that the quadratically convergent phase is the main reason that Newton's method outperforms many other algorithms. However, the drawback of using Newton's method is clear: one needs to solve a $n \times n$ system of linear equations in each iteration. For large scale problems with large n, Newton's method is intractable. Despite its obvious limitation, Newton's method is one of the best algorithms for moderate size optimization problems, and it serves as a base case for many further developments in unconstrained continuous optimization, including the two well-known types of algorithms: inexact Newton method and quasi-Newton method.

Quasi-Newton methods

Quasi-Newton methods were developed to overcome the major drawbacks of Newton's method: evaluation of Hessians and expensive iteration cost. The main idea of quasi-Newton methods is to approximate Newton steps using just first order information. In what follows, we provide the background materials of the BFGS method, which is the most popular quasi-Newton method. BFGS is named after Broyden, Fletcher, Goldfarb, and Shanno. We then discuss the drawback of BFGS, and how it could be relaxed by the limited-memory BFGS (L-BFGS). L-BFGS is one of the state-of-the-art methods for unconstrained optimization.

Consider the problem (2.6), at the k^{th} iteration, the direction of using Newton's method is computed by minimizing the following model,

$$\mathbf{d}_k = \arg\min_{\mathbf{d}\in\mathbb{R}^n} f(\mathbf{x}_k) + \langle \nabla f(\mathbf{x}_k), \mathbf{d}\rangle + \frac{1}{2}\mathbf{d}^T \nabla^2 f(\mathbf{x}_k)\mathbf{d}.$$

The basic idea of BFGS is to consider an approximation of the Hessian using the model,

$$m_k(\mathbf{d}) = f(\mathbf{x}_k) + \langle \nabla f(\mathbf{x}_k), \mathbf{d}\rangle + \frac{1}{2}\mathbf{d}^T \mathbf{B}_k \mathbf{d},$$

for a symmetric positive definite matrix $\mathbf{B}_k$. Notice that $\mathbf{B}_k$ is analogous to $\nabla^2 f(\mathbf{x}_k)$, as shown above. When using the BFGS method, the approximation of the Hessian is not computed afresh at every iteration, but rather it is updated using the previous estimate in the last iteration. In order to do so, one has to impose the updating rules at each iteration. In the case of BFGS, it is based on the idea of secant equation,

$$\nabla m_{k+1}(-\alpha_k \mathbf{d}_k) = \nabla f(\mathbf{x}_k).$$

The secant equation is based on the observation that $\nabla m_{k+1}(0) = \nabla f(\mathbf{x}_{k+1})$. If the secant equation is satisfied, then the model m_{k+1} is a good interpolation of the objective function f, and it has gradients that are matched at the points $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$. The above secant equation

can be re-written as,

$$\mathbf{B}_{k+1}\mathbf{s}_k = \mathbf{y}_k,$$

where $\mathbf{s}_k \triangleq \mathbf{x}_{k+1} - \mathbf{x}_k$ and $\mathbf{y}_k \triangleq \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)$. Using the idea of the above secant equation, we then can approximate the inverse of the Hessian $[\nabla^2 f(\mathbf{x}_{k+1})]^{-1}$ by,

$$\mathbf{s}_k = \mathbf{H}_{k+1}\mathbf{y}_k, \tag{2.8}$$

where $\mathbf{H}_{k+1} = \mathbf{B}_{k+1}^{-1}$ and so it is analogous to $[\nabla^2 f(\mathbf{x}_{k+1})]^{-1}$.

Equation (2.8) states the constraints when one updates the inverse of the Hessian. In the BFGS method, one updates the $\mathbf{H}_{k+1}$ based on the following optimization problem [NW06].

$$\begin{aligned} \min_{\mathbf{H}} \quad & \|\mathbf{H} - \mathbf{H}_k\|_{\tilde{\mathbf{G}}_k} \\ \text{subject to} \quad & \mathbf{H} = \mathbf{H}^T,\ \mathbf{H}\mathbf{y}_k = \mathbf{s}_k, \end{aligned} \tag{2.9}$$

where $\|\mathbf{A}\|_{\mathbf{W}} \triangleq \|\mathbf{W}^{1/2}\mathbf{A}\mathbf{W}^{1/2}\|_F$ for all matrix that satisfy $\mathbf{W}\mathbf{s}_k = \mathbf{y}_k$, and

$$\tilde{\mathbf{G}}_k = \int_0^1 \nabla^2 f(\mathbf{x}_k + \tau\alpha_k\mathbf{d}_k)\ \mathrm{d}\tau.$$

The optimization problem (2.9) has a unique analytical solution

$$\mathbf{H}_{k+1} = (\mathbf{I} - \rho_k\mathbf{s}_k\mathbf{y}_k^T)\mathbf{H}_k(\mathbf{I} - \rho_k\mathbf{y}_k\mathbf{s}_k^T) + \rho_k\mathbf{s}_k\mathbf{s}_k^T, \tag{2.10}$$

where $\rho_k = 1/\mathbf{y}_k^T\mathbf{s}_k$. This above update is the core step in BFGS, and it is used to update the approximation of inverse Hessian at each iteration.

Algorithm 2.6 provides the algorithmic procedure of BFGS method. The theoretical advantage of BFGS is stated in the following theorem.

Theorem 2.9 *Suppose Algorithm 2.6 is performed, then the generated sequence* $\{\mathbf{x}_k\}$ *converges*

Algorithm 2.6 BFGS method

Input parameters: Initial guess $\mathbf{x}_0 \in \mathbb{R}^n$, $\rho_1 \in (0, 1/2)$, $\beta_{ls} \in (0, 1)$, $\mathbf{H}_0 \in \mathbb{R}^{n\times n}$ which is symmetric and positive definite.

for $k = 0, 1, 2, \cdots$ **do**

Compute the direction

$$\mathbf{d}_k = -\mathbf{H}_k \nabla f(\mathbf{x}_k).$$

Find the smallest $q \in \mathbb{N}$ such that for $\alpha_k = \beta_{ls}^q$,

$$f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) \leq f(\mathbf{x}_k) + \rho_1 \alpha_k \nabla f(\mathbf{x}_k)^T \mathbf{d}_k.$$

Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$.

Update $\mathbf{H}_{k+1}$ using BFGS update (2.10).

end for

to the minimizer $\mathbf{x}_\star$ *of* f*. In particular, it converges at a superlinear rate. That is,*

$$\|\mathbf{x}_{k+1} - \mathbf{x}_\star\| \leq o(\|\mathbf{x}_{k+1} - \mathbf{x}_\star\|), \quad \text{as } k \to \infty.$$

The above definition of superlinearly convergence was taken in [DES82].

From Algorithm 2.6, one can see that BFGS requires a large amount of storage for the inverse Hessian $\mathbf{H}_k$ and requires a matrix-vector multiplication at each iteration. These requirements are not ideal when n is large. To reduce these computational burdens, limited-memory BFGS method was developed.

L-BFGS avoids the storage of the inverse Hessian by the following observation. From equation (2.10), one can rewrite the BFGS update as

$$\mathbf{H}_{k+1} = \mathbf{V}_k^T \mathbf{H}_k \mathbf{V}_k + \rho_k \mathbf{s}_k \mathbf{s}_k^T,$$

where $\mathbf{V}_k = \mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^T$. By recursively applying the above updating equation m times, for some positive integer m, L-BFGS could store the approximation of $\mathbf{H}_k$ implicitly by just storing the m lastest pairs of $\mathbf{s}_i$ and $\mathbf{y}_i$.

Algorithm 2.7 states the details of how L-BFGS computes the direction $\mathbf{d}_k$ using two-loop recursion. Notice that $\mathbf{H}_k^0$ in Algorithm 2.7 is often chosen to be the identity matrix. In this case one can see that L-BFGS has the computational advantage of only performing vector

Algorithm 2.7 L-BFGS direction update (at the k^{th} iteration)

Input: $\nabla f(\mathbf{x}_k)$, $\{\mathbf{s}_i, \mathbf{y}_i\}_{i=k-m}^{k-1}$, and $\mathbf{H}_k^0 \in \mathbb{R}^{n\times n}$ which is symmetric and positive definite.
Set $\mathbf{q}_{k-1} = \nabla f(\mathbf{x}_k)$.
for $i = k-1, k-2, \ldots, k-m$ **do**
  Update $\beta_i = \rho_i \mathbf{s}_i^T \mathbf{q}_i$.
  Update $\mathbf{q}_{i-1} = \mathbf{q}_i - \beta_i \mathbf{y}_i$.
end for
Set $\tilde{\mathbf{d}}_{k-m} = \mathbf{H}_k^0 \mathbf{q}_{k-m-1}$.
for $i = k-m, k-m+1, \ldots, k-1$ **do**
  Update $\zeta_i = \rho_i \mathbf{y}_i^T \tilde{\mathbf{d}}_i$.
  Update $\tilde{\mathbf{d}}_{i+1} = \tilde{\mathbf{d}}_i - \mathbf{s}_i(\beta_i - \zeta_i)$.
end for
Output: direction $\mathbf{d}_k = -\tilde{\mathbf{d}}_k$.

operations, and matrix operations are completely avoided. This is done by computing the approximation of the direction $-[\nabla^2 f(\mathbf{x}_k)]^{-1}\nabla f(\mathbf{x}_k)$ at once instead of having the two separate steps - approximating the inverse Hessian and then computing the direction.

We refer the readers to [BV04, NW06, Ber95] for more details regrading second order methods such as Newton's method, inexact Newton method, and quasi-Newton method.

2.2 Machine Learning and Statistics

In this section, we review some existing results in machine learning and statistics that serve as tools used in the thesis.

2.2.1 Concentration Bounds

Concentration bounds or concentration inequalities are used to analyze the likelihood of a random variable larger or smaller than some value. In other words, concentration bounds can give the output range of a random variable, with high probability.

In what follows, we will provide some concentration bounds for both scalar random variables and random matrices.

Theorem 2.10 ([Tro11]) *Let $\mathcal{Q}$ be a finite set of positive numbers, and suppose*

$$\max_{q\in\mathcal{Q}} q \leq B.$$

Sample $\{q_1, q_2, \ldots, q_l\}$ uniformly from $\mathcal{Q}$ without replacement. Compute

$$s = l \cdot \mathbb{E}(q_1).$$

Then

$$\mathbb{P}\left\{\sum_j q_j \leq (1-\sigma)s\right\} \leq \left(\frac{e^{-\sigma}}{(1-\sigma)^{1-\sigma}}\right)^{s/B} \qquad \textit{for} \quad \sigma \in [0,1), \quad \textit{and}$$
$$\mathbb{P}\left\{\sum_j q_j \geq (1+\sigma)s\right\} \leq \left(\frac{e^{\sigma}}{(1+\sigma)^{1+\sigma}}\right)^{s/B} \qquad \textit{for} \quad \sigma \geq 0.$$

Proof See Theorem 2.1 from Tropp [Tro11].

The above concentration bounds are called Chernoff bounds. We point out that $\mathbb{E}(q_1) = \sum_{q_j\in\mathcal{Q}} q_j/|\mathcal{Q}|$, since we assume sampling are conducted uniformly without replacement. It shows that, with high probability, the sum of samples over a finite set is bounded by $\mathcal{O}(l/|\mathcal{Q}|)$, where l is the size of the subset. We refer reader to [CL06] for other settings apart from sampling uniformly without replacement.

In this thesis, we are only interested in the largest eigenvalue of structured random matrices, which forms the basis of Chapter 4. Below we provide some results from [Tro12] that are used in Chapter 4.

Lemma 2.11 *For a sequence $\{\mathbf{Q}_k : k = 1, 2, \cdots, m\}$ of random matrices,*

$$\lambda_{\max}\left(\sum_k \mathbb{E}[\mathbf{Q}_k]\right) \leq \mathbb{E}\left[\lambda_{\max}\left(\sum_k \mathbf{Q}_k\right)\right].$$

We shall mention that the above lemma is a result of Jensen's inequality.

Lemma 2.12 *Suppose that* $\mathbf{Q}$ *is a random positive semi-definite matrix that satisfies* $\lambda_{\max}(\mathbf{Q}) \leq 1$. *Then*

$$\mathbb{E}\left[e^{\theta \mathbf{Q}}\right] \preccurlyeq \mathbf{I} + (e^{\theta} - 1)(\mathbb{E}\left[\mathbf{Q}\right]), \quad \textit{for } \theta \in \mathbb{R},$$

where $\mathbf{I}$ *is the identity matrix in the correct dimension.*

Lemma 2.13 *Consider a sequence* $\{\mathbf{Q}_k : k = 1, 2, \cdots, m\}$ *of independent, random, self-adjoint matrices with dimension* n. *For all* $t \in \mathbb{R}$,

$$\mathbb{P}\left\{\lambda_{\max}\left(\sum_{k=1}^{m} \mathbf{Q}_k\right) \geq t\right\} \leq n \inf_{\theta > 0} \exp\left(-\theta t + m \ \log \lambda_{\max}\left(\frac{1}{m}\sum_{k=1}^{m} \mathbb{E}e^{\theta \mathbf{Q}_k}\right)\right).$$

One may find that the above inequality is not meaningful when t is small, i.e. the right hand side is greater than 1 when t is small. However, as we will see in Chapter 4, in this thesis we are interested in the cases where t is sufficiently large.

2.2.2 The Nyström Method

We shall first emphasize that originally Nyström method was developed as a numerical method to approximate eigenfunctions [Nys30]. However, the main idea of Nyström method was used to approximate Gram matrix for machine learning applications, and then this name was also used as a computational method for low rank approximation of Gram matrices [WS01]. We clarify that this thesis only considers the latter case.

In the era of big data, storage of matrices becomes a limitation for some applications. Fortunately, many matrices that are formed from data exhibit the low rank structure. For a positive semi-definite matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, its best low rank approximation can be recognized as the following optimization problem

$$\min_{\mathbf{A}_q \in \mathbb{R}^{N \times N}} \|\mathbf{A} - \mathbf{A}_q\|_2, \quad \text{s.t.} \quad \text{rank}(\mathbf{A}_q) = q. \tag{2.11}$$

The solution of the above problem, $\mathbf{A}_{q,\star}$, is a matrix with rank q. It is obvious that when

$q = N$, then $\mathbf{A}_{q,\star} = \mathbf{A}$. When one restricts $q << N$, then $\mathbf{A}_{q,\star}$ is the best rank q approximation of the original matrix $\mathbf{A}$. We mention that in the problem (2.11), $\|\cdot\|_2$ is chosen to be measure of the distance between $\mathbf{A}$ and $\mathbf{A}_q$. In general, any measure or norm can be used, but the two most common choices are $\|\cdot\|_2$ and $\|\cdot\|_F$.

Whether $\|\cdot\|_2$ or $\|\cdot\|_F$ is used, an analytical solution of (2.11) is available due to the Eckart-Young-Mirsky theorem [EY36, Mir60]. Denote the eigenvalue decomposition of $\mathbf{A}$ as follows,

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T = \begin{pmatrix} \mathbf{U}_1 & \mathbf{U}_2 \end{pmatrix} \begin{pmatrix} \mathbf{\Sigma}_1 & \\ & \mathbf{\Sigma}_2 \end{pmatrix} \begin{pmatrix} \mathbf{U}_1 & \mathbf{U}_2 \end{pmatrix}^T \tag{2.12}$$

where $\mathbf{U}_1 \in \mathbb{R}^{N\times q}$, $\mathbf{U}_2 \in \mathbb{R}^{N\times(N-q)}$, $\mathbf{\Sigma}_1 \in \mathbb{R}^{q\times q}$, and $\mathbf{\Sigma}_2 \in \mathbb{R}^{(N-q)\times(N-q)}$. We also assume that eigenvalues in $\mathbf{\Sigma}$ are sorted in descending order. Then $\mathbf{A}_{q,\star}$ has the form,

$$\mathbf{A}_{q,\star} = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{U}_1^T. \tag{2.13}$$

We clarify that Eckart-Young-Mirsky theorem applies for the setting in which $\mathbf{A}$ does not need to be positive semi-definite nor a square matrix, but for the purposes of this chapter we only consider the case where $\mathbf{A}$ is positive semi-definite.

From (2.13) one can see that computing the exact solution $\mathbf{A}_{q,\star}$ is computationally expensive when N is large, because it requires the eigenvalue decomposition on $\mathbf{A}$. To this end, the Nyström method is developed to compute the approximation of $\mathbf{A}_{q,\star}$ and thus a good low rank approximation of $\mathbf{A}$.

The details of Nyström method are provided in Algorithm 2.8, which can also be found in [DM05]. Notice that $\mathbf{A}_q$ can be also recognized in the following form,

$$\mathbf{A}_q = \mathbf{A}\mathbf{S}[\mathbf{S}^T\mathbf{A}\mathbf{S}]^+\mathbf{S}^T\mathbf{A}, \tag{2.14}$$

where $\mathbf{S} \in \mathbb{R}^{N\times q}$ such that the i^{th} column of $\mathbf{S}$ is the q_i^{th} column of $\mathbf{I}$. One can also notice the advantage of using Nyström's method in terms of storage. The low rank approximation

Algorithm 2.8 Nyström's method

Input parameters: A positive semi-definite matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$

Step 1. Construct a set $\mathcal{Q}_1 \subseteq \{1, 2, \cdots, N\}$, and let q_i be the i^{th} element of $\mathcal{Q}_1$.

Step 2. Construct a matrix $\mathbf{A}_1 \in \mathbb{R}^{q\times N}$ such that the i^{th} row of $\mathbf{A}_1$ is the q_i^{th} row of $\mathbf{A}$.

Step 3. Construct a matrix $\mathbf{A}_2 \in \mathbb{R}^{N\times q}$ such that the i^{th} column of $\mathbf{A}_2$ is the q_i^{th} column of $\mathbf{A}$.

Step 4. Construct a matrix $\mathbf{A}_3 \in \mathbb{R}^{q\times q}$ such that $(\mathbf{A}_3)_{i,j}$ is $(\mathbf{A})_{q_i,q_j}$.

Step 5. Compute $\mathbf{A}_q = \mathbf{A}_2\mathbf{A}_3^+\mathbf{A}_1$, where $\mathbf{A}_3^+$ is the pseudo-inverse of $\mathbf{A}_3$.

$\mathbf{A}_q$ can be stored as three matrices: $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3^+$. When $q \ll N$, the storage requirement of $\mathbf{A}_q$ is much less than the original matrix $\mathbf{A}$. Also, the computational cost of performing pseudo-inverse $\mathbf{A}_3^+$ is not expensive when q is small.

In general, $\mathcal{Q}_1$ is constructed using one of the following methods:

i. Uniform sampling (with or without replacement).

ii. Adaptive sampling based on the scores assigned on the columns of $\mathbf{A}$.

iii. Deterministic methods based on the decrease in error, $\|\mathbf{A}_q - \mathbf{A}\|$, by selecting a particular column of $\mathbf{A}$.

We will focus on the case where uniform sampling without replacement is deployed. This version of Nyström method is called the naïve Nyström's method.

Theorem 2.14 *Let $\mathbf{A} \in \mathbb{R}^{N\times N}$ be a positive semi-definite matrix, $\mathbf{S} \in \mathbb{R}^{N\times q}$ be a matrix as defined in (2.14), and the eigenvalue decomposition $\mathbf{A}$ has the form in (2.12).*

Let τ denotes the coherence of $\mathbf{U}_1$,

$$\tau = \mu_0(\mathbf{U}_1) \triangleq \frac{N}{q}\max_i(\mathbf{U}_1\mathbf{U}_1^T)_{ii}.$$

Then $\forall \delta, \epsilon \in (0,1)$*, suppose*

$$q \geq \frac{2\tau k \log(k/\delta)}{(1-\epsilon)^2}.$$

The error of using the naïve Nyström method is

$$\|\mathbf{A} - \mathbf{A}_q\| \leq \lambda_{k+1}(\mathbf{A}) \left(1 + \frac{N}{\epsilon q}\right),$$

with probability at least $1-\delta$*, where* $\lambda_{k+1}(\mathbf{A})$ *is the* $(k+1)^{th}$ *largest eigenvalue of* $\mathbf{A}$*.*

Theorem 2.14 shows the theoretical performance of the naïve Nyström method. As expected, naïve Nyström method achieves good error bound when $\mathbf{A}$ has a large spectral gap, i.e. when $\lambda_N(\mathbf{A}) \leq \lambda_{N-1}(\mathbf{A}) \leq \cdots \leq \lambda_{k+1}(\mathbf{A}) \ll \lambda_k(\mathbf{A}) \leq \cdots \leq \lambda_1(\mathbf{A})$.

We refer readers to [DM05, Git13, WS01, SS00] for different versions of Nyström method. Apart from Nyström method, random projection is another technique for low rank approximation. See [HMT11] for a review in details.

2.3 Continuous-Time Markov Decision Processes

Markov decision processes (MDPs) are considered to be one of the standard models for sequential decision problems under uncertainties. Many practical applications in operations research can be categorized as MDPs. Examples include road maintenance [GS97], nuclear plant management [RR95], and revenue management [SSJL99]. To formally define MDPs, we first introduce Markov chains.

2.3.1 Continuous-Time Markov Chains

Continuous-time Markov Chains (CTMCs), by definition, follow the Markov property, i.e. the probability of the future only depends on the current situation but not the past [Ros06]. Every CTMC is constructed by (1) a group of states to illustrate all possible situations in the system

and (2) a Markov generator which defines the probabilistic transitions between any two states. We further assume the number of states is always finite. One could refer such Markov chains as discrete-state continuous-time Markov chains.

Suppose we let $x(t)$ to be a CTMC. We can represent the transitions by a matrix called transition probability matrix $\mathbf{P}(t,s) = (p_{ij}(t,s))$ where $p_{ij}(t,s)$ represents the probability of $x(t) = j$ given $x(s) = i$ for $0 \leq s \leq t$; that is,

$$p_{ij}(t,s) = \mathbb{P}(x(t) = j | x(s) = i).$$

We should emphasize that, in this thesis, we are only interested in Markov chains that are irreducible, which means it is possible (positive probability) for $x(t)$ to visit any state in the future regardless of the current state. For CTMCs, it is known that all transition probability matrices follow the below differential equation

$$\begin{aligned} \frac{\mathrm{d}\mathbf{P}(t,s)}{\mathrm{d}t} &= \mathbf{P}(t,s)\mathbf{Q}(t) \quad , \quad s \leq t, \\ \mathbf{P}(s,s) &= \mathbf{I}. \end{aligned}$$

The matrix $\mathbf{Q}(t) = (q_{ij}(t))$ is called the Markov generator which satisfies

$$q_{ii}(t) = -\sum_{j \neq i} q_{ij}(t) \quad , \quad q_{ij}(t) \geq 0, \quad \text{for} \quad j \neq i.$$

We point out that the diagonal elements of $\mathbf{Q}(t)$, q_{ii}'s, represent the rates of exiting current state. The off-diagonal elements, q_{ij}'s, represent the likelihood that a transition from i to j will occur. Suppose a CTMC is in state i at time 0, it will leave state i at time t where t is a random variable which follows an exponential distribution with parameter q_{ii}, and enter a state j with probability $q_{ij}/|q_{ii}|$, $\forall j \neq i$. In this thesis, we use "$x \sim \mathbf{Q}(t)$" to represent the following statement: $x(t)$ is a CTMC in which the uncertainty is govern by the Markov generator $\mathbf{Q}(t)$.

The magnitude of $\mathbf{Q}$ is related to the "speed" of the process. In general, the larger magnitude represents a faster process. For instance, consider the two Markov generators, $\mathbf{Q}_1$ and $\mathbf{Q}_2$,

where $\mathbf{Q}_2 = 10\mathbf{Q}_1$. In such case $\mathbf{Q}_1$ and $\mathbf{Q}_2$ will generate the same Markov processes except the one from $\mathbf{Q}_2$ is 10 times faster.

2.3.2 Continuous-Time MDPs

The general goal of Markov Decision Process is to find the best policy $\mathbf{u}$ over a Markov process, in which the underlying uncertainties of the process depend partly on the policy. A policy is analogous to a lookup table which suggests an action a_j to each state j in the Markov process.

We denote an N state MDP with $x(t)$ and takes values from the state space $\mathcal{X} := \{1, 2, \cdots, N\}$. For each state i, $1 \leq i \leq N$, the available actions of the state i are denoted in the set $\mathcal{A}_i$. Therefore, a policy can be described by $\mathbf{u} = (a_1, a_2, \cdots, a_N)$ where $a_i \in \mathcal{A}_i$ for $\forall i \in \mathcal{X}$. The policy space for all policies $\mathbf{u}$'s is denoted by $\mathcal{U} := \{(a_1, a_2, \cdots, a_N) : a_i \in \mathcal{A}_i, \forall i \in \mathcal{X}\}$.

The unconstrained MDP is stated as follows,

$$\begin{aligned} \min_{\mathbf{u}\in\mathcal{U}} \quad & J(i,\mathbf{u}) = \mathbb{E}\left[\int_0^\infty e^{-\rho t} G(x(t), \mathbf{u}(x(t)))\ \mathrm{d}t\right], \\ \text{subject to} \quad & x \sim \mathbf{Q}(\mathbf{u}(x(t)))\ ,\ t \geq 0, \\ & x(0) = i, \end{aligned} \tag{2.15}$$

where $\mathbf{Q}(\cdot) = (q_{ij}) \in \mathbb{R}^{N\times N}$ is a Markov generator, $G(\cdot,\cdot)$ is the cost function, and $\rho > 0$ is the discount factor. As we can see, the generator $\mathbf{Q}$ depends on feedback control (policy $\mathbf{u}$); so our actions affect the Markov chain and thus the uncertainty. Also, the Markov generator $\mathbf{Q}(\cdot)$ we consider is time-invariant and independent of time t.

2.3.3 Computational Methods

The optimization model (2.15) can be solved by introducing the value function

$$\mathbf{v}(i) = \min_{\mathbf{u}\in\mathcal{U}} J(i,\mathbf{u}), \tag{2.16}$$

which satisfies

$$\rho \mathbf{v}(i) = \min_{a \in \mathcal{A}_i} \left[G(i,a) + \sum_{j \in \mathcal{X}, j \neq i} q_{ij}(a)[\mathbf{v}(j) - \mathbf{v}(i)] \right]. \tag{2.17}$$

Equation (2.17) is called the Hamilton-Jacobi-Bellman (HJB) equation. An equivalent form is

$$\mathbf{v}(i) = \min_{a \in \mathcal{A}_i} \left[\frac{G(i,a)}{|q_{ii}(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}(a)}{|q_{ii}(a)| + \rho} \mathbf{v}(j) \right]. \tag{2.18}$$

The derivation of (2.17) and (2.18) can be founded in [YZ13]. The above problem can be recognized as a nonlinear equation

$$A\mathbf{v} = 0, \tag{2.19}$$

where

$$(A\mathbf{v})(i) := \min_{a \in \mathcal{A}_i} \left[\frac{G(i,a)}{|q_{ii}(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}(a)}{|q_{ii}(a)| + \rho} \mathbf{v}(j) \right] - \mathbf{v}(i). \tag{2.20}$$

By solving the HJB equation, one can find the value function $\mathbf{v}^\star(x)$ which represents the lowest possible expected cost of the problem. Once $\mathbf{v}^\star(x)$ is found, the optimal policy of (2.15) can be obtained by

$$\mathbf{u}^\star(i) \in \arg\min_{a \in \mathcal{A}_i} \left[\frac{G(i,a)}{|q_{ii}(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}(a)}{|q_{ii}(a)| + \rho} \mathbf{v}^\star(j) \right]. \tag{2.21}$$

Therefore, solving the HJB equation is equivalent to solving for the optimal policy $\mathbf{u}^\star$ [Ber07]. The state-of-the-art methods for solving HJB equations are characterized as different categories such as linear programming, policy iteration, and value iteration [Pow11]. For the purpose of this chapter, we will only introduce the latter two methods.

Value Iteration

The method of value iteration is based on the following nonlinear operator

$$(Tv)(i) := \min_{a \in \mathcal{A}_i} \left[\frac{G(i,a)}{|q_{ii}(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}(a)}{|q_{ii}(a)| + \rho} \mathbf{v}(j) \right]. \tag{2.22}$$

It is well-known that T is a contraction mapping [YZ13], e.g. it satisfies

$$\|T\mathbf{v}_1 - T\mathbf{v}_2\|_\infty \leq \alpha \|\mathbf{v}_1 - \mathbf{v}_2\|_\infty, \tag{2.23}$$

where $0 \leq \alpha \leq 1$. In our case,

$$\alpha = \max_{i \in \mathcal{X}, a \in \mathcal{A}_i} \frac{|q_{ii}(a)|}{|q_{ii}(a)| + \rho}. \tag{2.24}$$

By the properties of contraction mapping and the Banach fixed point theorem [TBI97], for any initial guess $\mathbf{v}^0$, one can compute the solution of the HJB equation by iteratively applying T on $\mathbf{v}^0$. That is,

$$\mathbf{v}^\tau := T^\tau \mathbf{v}^0 = \underbrace{(T \circ T \circ \cdots \circ T)}_{\tau \ T\text{'s}} \mathbf{v}^0 \to \mathbf{v}^\star \quad \text{as} \quad \tau \to \infty. \tag{2.25}$$

Using the operator T iteratively as the above equation, one can compute the approximation of $\mathbf{v}^\star$. This approach is called value iteration.

Linear Programming

One can also solve the model (2.15) using linear programming (LP). It is known that solving (2.18) is equivalent to solving

$$\begin{aligned} \max_{\mathbf{v} \in \mathbb{R}^N} \quad & \sum_{i \in \mathcal{X}} \mathbf{v}(i), \\ \text{s.t.} \quad & \mathbf{v}(i) \leq \frac{G(i,a)}{|q_{ii}(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}(a)}{|q_{ii}(a)| + \rho} \mathbf{v}(j), \quad \forall a \in \mathcal{A}_i, i \in \mathcal{X}. \end{aligned} \tag{2.26}$$

The constraints of (2.26) can be proved to be equivalent to $\mathbf{v} \leq T\mathbf{v}$. Based on the monotonicity property of T [Pow11], this constraint requires that all feasible solution $\mathbf{v}$ to satisfy the condition $\mathbf{v} \leq \mathbf{v}^\star$, in which $\mathbf{v}^\star$ is the optimal value function in (2.16). Since this a maximization problem and $\mathbf{v}^\star$ is a feasible solution, solving the above LP is equivalent to solving the HJB equation (2.18), and thus equivalent to solving the model (2.15).

We refer readers to [Ros06, Ber07, Pow11] for more details in CTMCs and MDPs.

Chapter 3

Singularly Perturbed Markov Decision Processes

With four parameters I can fit an elephant, and with five I can make him wiggle his trunk.

John von Neumann

Singular perturbation techniques allow the derivation of an aggregate model whose solution is asymptotically optimal for Markov Decision Processes with strong and weak interactions. In this chapter, we develop an algorithm that takes advantage of the asymptotic optimality of the aggregate model in order to compute the solution of the original model. We derive conditions for which the proposed algorithm has better worst case complexity than conventional contraction algorithms. Based on our complexity analysis we show that the major benefit of aggregation is that the reduced order model is no longer ill conditioned. The reduction in the number of states (due to aggregation) is a secondary benefit. This is a surprising result since intuition would suggest that the reduced order model can be solved more efficiently because it has fewer states. However we show that this is not necessarily the case. Our theoretical analysis and numerical experiments show that the proposed algorithm can compute the optimal solution with a reduction in computational complexity and without any penalty in accuracy.

3.1 Introduction

Recently there has been considerable interest in modeling and control of stochastic dynamics across different timescales. Typical applications appear in molecular dynamics [Chr09], networked systems [Mey08], manufacturing [SZ94], and optimal control of energy systems [PW14], just to name a few. Controlling dynamics across different scales is computationally difficult and a considerable amount of literature has been devoted to the challenge of finding approximate models that capture the effective dynamics of the system. The main techniques used for optimal control are based around aggregation, averaging and homogenization. Starting from the work of Simon and Ando [SA61] hierarchical decomposition and aggregation has been at the core of approximation techniques for modeling and controlling dynamics across different scales. The literature around this topic is substantial and we refer the interested reader to [KKO87] for early work on singular perturbation techniques in optimal control. The averaging principle and applications in manufacturing are described in [SZ94]. The homogenization for deterministic optimal control problems has been studied in [BM08]. The recent research monograph by Yin and Zhang [YZ13] describes the main mathematical results in the context of stochastic optimal control using the theory of singularly perturbed Markov processes. The mathematical framework described in [YZ13] is the one we adopt in this chapter. The main result of the aggregation techniques and averaging principles reviewed in [SZ94] and [YZ13] is the derivation of an approximate model that captures the slow dynamics of the system. The approximate model is based on an asymptotic analysis of a singularly perturbed control problem (see [YZ13] for details, and Section 3.2 of this chapter for precise definitions).

The mathematical properties and especially the use of asymptotic techniques coupled with the perturbation approach for controlling Markov processes have been extensively studied. However, numerical methods that take into consideration the specific structure of multiscale Markov processes have not received much attention. Given all the work that has gone into the development of aggregate models, it is surprising that the obvious question of whether the reduced order models can be solved more efficiently than the original model has not been addressed. We take the first steps towards answering this question for a particular class of

multiscale Markov processes. Based on our complexity analysis *we show that the major benefit of aggregation is that the reduced order model is no longer ill conditioned, and the reduction in the number of states (due to aggregation) is a secondary benefit.* This is a surprising result since intuition would suggest that the reduced order model can be solved more efficiently because it has fewer states. However, it will be shown later that this is not necessarily the case. There is no standard definition for an ill conditioned Markov Decision Process (MDP). In the context of this chapter a MDP is ill conditioned if the contraction modulus of value iteration is approximately equal to one. This means that progress at each iteration will be extremely slow. We propose a class of multiresolution contraction algorithms that are not sensitive to the ill conditioning of weakly connected MDPs. Because we are considering a particular class of MDPs we are able to improve the worst case complexity of algorithms based on value iteration. We illustrate our approach on value iteration, but any contraction algorithm can potentially be improved using the proposed scheme.

It is important to stress that the proposed algorithm aims to solve the original model and not just obtain an approximation using the aggregate model. The aggregate model is only asymptotically optimal and our algorithm exploits its approximate optimality to reduce the number of iterations with the high dimensional (and often ill-conditioned) model. Our algorithm is ideal when there is some scale separation but it is not known whether there is sufficient scale separation to just solve the approximate model. This setting is the most frequent scenario encountered in practice. For simplicity we study a Multiscale Markov Decision Process (MMDP) with two timescales, but generalizing the results to problems with more than two time scales is straightforward.

The rest of the chapter is structured as follows: In Section 3.2 we define the notation we use and provide a review of existing results. In Section 3.3 we review complexity results for the value iteration algorithm. We extend some known results from discrete time to continuous time and give particular emphasis to MMDPs. In Section 3.4 we review the Full Approximation Scheme (FAS). The FAS can be used to accomplish some of the objectives we set to achieve in this chapter, i.e. take advantage of the structure of MMDPs to improve the computational efficiency of algorithms for this class of MDPs. The FAS is a non-linear extension to the

traditional multigrid scheme, and in Section 3.4, we show that it may not be an appropriate choice for MDPs. Based on our observations of the complexity of MMDPs in Section 3.3 and the FAS scheme in Section 3.4, we propose an alternative scheme in Section 3.5. We named our proposed scheme the **A**lternating **M**ultiresolution **S**cheme (AMS) since it uses features from the FAS and known results regarding the quality of the approximate (aggregate) model. In Section 3.6, we propose a refinement of our scheme that allows the scheme to be applied to problems that have a large number of actions. Finally, in Section 3.7, we illustrate the proposed scheme on two applications, one from manufacturing and one from chemistry.

3.2 Multiscale Markov Decision Processes

The notation and framework for MMDPs we adopt in this chapter is standard and more information and results can be obtained in [YZ13].

3.2.1 Markov Decision Processes

Let $x_h(t)$ denote the state of a discrete state continuous time Markov Decision Process (MDP) at time t. We use the subscript h to denote processes that capture effects at the fast time scale h. We assume that the chain can take one of the finite number states $\mathcal{X}^h \triangleq \{l_1, l_2, \cdots, l_N\}$. For each of the states i, $1 \leq i \leq N$, the available actions of state i are denoted by the set $\mathcal{A}_i^h$. A policy $\mathbf{u}_h : \mathcal{X}^h \to \mathcal{A}^h$ maps states into actions and is described by $\mathbf{u}_h = (a_1, a_2, \cdots, a_N)$ where $a_i \in \mathcal{A}_i^h$ for $\forall i \in \mathcal{X}^h$. The space of all policies $\mathbf{u}_h$'s is denoted by $\mathcal{U}^h \triangleq \{(a_1, a_2, \cdots, a_N) : a_i \in \mathcal{A}_i^h, i = 1, 2, \ldots, N\}$ and we use $\mathcal{A}^h$ to denote the space of all possible actions i.e. $\mathcal{A}^h = \cup_{i=1}^N \mathcal{A}_i^h$. Note that $\mathcal{A}_i^h$'s are assumed to be time independent in this chapter. We assume that we are given a cost function $G^h : \mathcal{X}^h \times \mathcal{A}^h \to \mathbb{R}$ that measures the cost associated with a particular state-action pair. We will focus on the infinite horizon case and denote the discount factor by ρ. All the results reported in the chapter can be extended to the finite horizon case. We use $\mathcal{MDP}(N, L)$ to denote the class of problems for which $|\mathcal{X}^h| = N$, $|\mathcal{A}_i^h| = L$, for $i = 1, 2, \ldots, N$. It is easy to generalize our results to the case where each of the action spaces $\mathcal{A}_i^h$ have different

cardinality $|\mathcal{A}_i^h| = L_i$, but for ease of exposition we assume that $|\mathcal{A}_i^h| = L$, for $i = 1, 2, \ldots, N$. With the notation introduced above we are now in a position to state the class of problems we study in this chapter,

$$\min_{\mathbf{u}_h \in \mathcal{U}^h} \qquad J^h(i, \mathbf{u}_h) = \mathbb{E}\left[\int_0^\infty e^{-\rho t} G^h(x_h(t), \mathbf{u}_h(x_h(t)))\, \mathrm{d}t \middle| x_h(0) = i\right]. \tag{3.1}$$

The expectation above is taken with respect to a probability matrix $\mathbf{P}$ and we use $\mathbf{P}_{i,j}(t, s)$ to denote the probability of the process, $x_h(t)$, transitioning to state j at time t given that it starts from state i at time s. According to the theory of Markov processes the transition matrix satisfies the following equation,

$$\frac{\mathrm{d}\mathbf{P}(t, s)}{\mathrm{d}t} = \mathbf{P}(t, s)\mathbf{Q}_h^\epsilon(\mathbf{u}_h), \quad \mathbf{P}(s, s) = \mathbf{I}_N, \tag{3.2}$$

where $\mathbf{Q}_h^\epsilon$ denotes the infinitesimal generator of x_h, and $\mathbf{I}_N$ denotes the $N \times N$ identity matrix. We are focusing on a Markov process with a multiscale structure and so we assume the generator of the process is defined as follows,

$$\mathbf{Q}_h^\epsilon(\mathbf{u}_h) = \frac{1}{\epsilon}\hat{\mathbf{Q}}(\mathbf{u}_h) + \mathbf{W}(\mathbf{u}_h), \tag{3.3}$$

where $\hat{\mathbf{Q}}(\mathbf{u}_h) = \mathrm{diag}(\hat{\mathbf{Q}}_1(\mathbf{u}_h), \hat{\mathbf{Q}}_2(\mathbf{u}_h), \ldots, \hat{\mathbf{Q}}_m(\mathbf{u}_h))$ is a block diagonal matrix with m blocks, with $\hat{\mathbf{Q}}_k(\mathbf{u}_h) \in \mathbb{R}^{n_k \times n_k}$ and $\sum_{j=1}^m n_j = N$, for $k = 1, 2, \ldots, m$. We further assume that each $\hat{\mathbf{Q}}_k(\mathbf{u}_h)$ is a weakly irreducible Markov generator. A Markov generator $\mathbf{Q}$ is said to be weakly irreducible if there exists a row vector $\boldsymbol{\nu} \geq 0$ such that $\boldsymbol{\nu}\mathbf{Q} = \mathbf{0}$ and $\|\boldsymbol{\nu}\|_1 = 1$ [YZ13]. For ease of exposition we assume that all blocks have the same size (n). All our results can easily be generalized to the case where each block has size n_i. We use $\mathcal{X}_k = \{l_{k1}, \ldots, l_{kn_k}\}$, $k = 1, \ldots, m$ to denote the states corresponding to $\hat{\mathbf{Q}}_k$. This decomposition is done so that $\mathcal{X} = \cup_{k=1}^m \mathcal{X}_k$. The small parameter ϵ is used to capture the multiscale structure of the process. When $\epsilon \ll 1$ the Markov process jumps frequently between the states within a block $\mathcal{X}_k$ and less frequently between states that belong to different blocks. The matrix $\mathbf{W}(\mathbf{u}_h)$ is also assumed to be a Markov generator and it is used to model the transition between the blocks. The smaller the

ϵ, the faster the transitions inside the blocks. As ϵ approaches zero, the transitions inside the blocks happen at such a fast rate that the process can be approximated by the equilibrium distribution inside each of the blocks. This idea can be made rigorous and we refer the reader to Chapter 6 of [YZ13] for the details. Our aim is to study the complexity and propose an efficient algorithm for the solution of the stochastic control problem in (3.1).

The class of weakly connected processes will be denoted by $\mathcal{MMDP}(\epsilon, n, m, L)$, which is a subclass of $\mathcal{MDP}(nm, L)$. Using the dynamic programming principle it can be shown that the value function associated with the problem,

$$\mathbf{v}_h(i) = \min_{\mathbf{u}_h \in \mathcal{U}^h} J^h(i, \mathbf{u}_h), \tag{3.4}$$

satisfies the so called the Hamilton-Jacobi-Bellman (HJB) equation,

$$\rho \mathbf{v}_h(i) = \min_{a \in \mathcal{A}_i^h} \left[G^h(i,a) + \sum_{j \in \mathcal{X}^h, j \neq i} q_{ij}^h(a)[\mathbf{v}_h(j) - \mathbf{v}_h(i)] \right]. \tag{3.5}$$

Notice that we use min instead of inf in our problem definition because the action space is finite and we will assume G^h is bounded. It was shown in [YZ13] that (3.5) is equivalent to,

$$\mathbf{v}_h(i) = \min_{a \in \mathcal{A}_i^h} \left[\frac{G^h(i,a)}{|q_{ii}^h(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}^h(a)}{|q_{ii}^h(a)| + \rho} \mathbf{v}_h(j) \right]. \tag{3.6}$$

Our analysis will be based on the properties of the contraction operator derived from value iteration. With that in mind we can rewrite the problem of computing the value function as the solution of the following nonlinear equation,

$$A_h \mathbf{v}_h = 0,$$

where A_h is a nonlinear operator defined as follows,

$$(A_h \mathbf{v}_h)(i) \triangleq \min_{a \in \mathcal{A}_i^h} \left[\frac{G^h(i,a)}{|q_{ii}^h(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}^h(a)}{|q_{ii}^h(a)| + \rho} \mathbf{v}_h(j) \right] - \mathbf{v}_h(i). \tag{3.7}$$

The contraction operator will be denoted by T_h and is defined below,

$$(T_h \mathbf{v}_h)(i) \triangleq \min_{a \in \mathcal{A}_i^h} \left[\frac{G^h(i,a)}{|q_{ii}^h(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}^h(a)}{|q_{ii}^h(a)| + \rho} \mathbf{v}_h(j) \right]. \tag{3.8}$$

So far we have not used our assumption that $\epsilon \ll 1$. This setting has been extensively studied in the last thirty years and in the next section we summarize the results we will need in our analysis.

3.2.2 The Coarse Model

The computational cost of solving (3.1) exactly is extremely high when the Markov process has a large number of states. Many researchers noticed that if the problem has the multiscale structure described in the previous section then the computational costs can be reduced by considering an approximate model. In the approximate model each set of states associated with each of the "fast" blocks is aggregated into a single state. For this reason the resulting approximate model is called the aggregate model. In this chapter, however, we will adopt the terminologies from the multigrid community by using **fine model** (defined in (3.1)) and **coarse model** (defined in (3.9) below) instead of exact and aggregate model. It can be shown that if ϵ is small enough, the coarse model becomes arbitrarily accurate. There are many results of this type for the model described in the previous section, as well as generalizations to different models. These results are described in [YZ13], and we refer the reader there for a comprehensive literature review. In our work we will need to make use of the coarse model and we describe the notation we use below.

The state space of the coarse model is denoted by $\mathcal{X}^H \triangleq \{l'_1, l'_2, \cdots, l'_m\}$, where each state i in the coarse model represents block i in the fine model. The available actions of state i in the coarse model a_i^H, are combinations of the available actions in block i, and they form the action space $\mathcal{A}_i^H \triangleq \{(a_1^i, a_2^i, \ldots, a_n^i) : a_j^i \in \mathcal{A}_{\mathcal{X}_i(j)}^h\}$. Therefore, $\mathbf{u}_H$ is the policy of the coarse model, and it takes values from the policy space $\mathcal{U}^H \triangleq \{(a_1^H, a_2^H, \ldots, a_m^H) : a_i^H \in \mathcal{A}_i^H, i = 1, 2, \ldots, m\}$. The coarse model is an $\mathcal{MDP}(m, L^n)$ model.

In order to define the coarse model we also need to define both the coarse Markov generator and the coarse objective function. Let $\boldsymbol{\varphi}_1(\mathbf{u}_H), \ldots, \boldsymbol{\varphi}_m(\mathbf{u}_H)$ denote the stationary distributions of the blocks $1, 2, \ldots, m$ in the form of column vectors under policy $\mathbf{u}_H$. We obtain the corresponding Markov generator,

$$\mathbf{Q}_H(\mathbf{u}_H) = \boldsymbol{\varphi}(\mathbf{u}_H)\mathbf{W}(\mathbf{u}_H)\tilde{\mathbf{1}},$$

where

$$\begin{aligned} \boldsymbol{\varphi}(\mathbf{u}_H) &= \operatorname{diag}(\boldsymbol{\varphi}_1^T(\mathbf{u}_H), \boldsymbol{\varphi}_2^T(\mathbf{u}_H), \ldots, \boldsymbol{\varphi}_m^T(\mathbf{u}_H)), \\ \tilde{\mathbf{1}} &= \operatorname{diag}(\underbrace{\mathbf{1}_{n\times 1}, \mathbf{1}_{n\times 1}, \ldots, \mathbf{1}_{n\times 1}}_{m \text{ copies}}), \end{aligned}$$

where $\mathbf{1}_{n\times 1} \triangleq (1, 1, \ldots, 1)^T \in \mathbb{R}^{n\times 1}$, and $\operatorname{diag}(\cdot)$ is a function which maps its argument to a diagonal matrix. The coarse cost function is given by,

$$G^H(i, \mathbf{u}_H) = \sum_{k=1}^{n} (\boldsymbol{\varphi}_i(\mathbf{u}_H))_k G^h(\mathcal{X}_i(k), ((\mathbf{u}_H)_i)_k) \quad \forall i \in \mathcal{X}^H.$$

Given the notation above, the coarse model is,

$$\begin{aligned} \min_{\mathbf{u}_H} \quad & J^H(i, \mathbf{u}_H) = \mathbb{E}\left[\int_0^\infty e^{-\rho t} G^H(x_H(t), \mathbf{u}_H(x_H(t)))\, \mathrm{d}t\right], \\ \text{s.t} \quad & x_H \sim \mathbf{Q}_H(\mathbf{u}_H(x_H(t)))\ ,\ t \geq 0, \\ & x_H(0) = i \quad , \quad \mathbf{u}_H \in \mathcal{U}^H, \\ & \mathbf{v}_H(i) = \min_{\mathbf{u}_H \in \mathcal{U}^H} J^H(i, \mathbf{u}_H). \end{aligned} \tag{3.9}$$

The corresponding HJB equation becomes,

$$\mathbf{v}_H(i) = \min_{a_H \in \mathcal{A}_i^H} \left[\frac{G^H(i, a_H)}{|q_{ii}^H(a_H)| + \rho} + \sum_{j \neq i} \frac{q_{ij}^H(a_H)}{|q_{ii}^H(a_H)| + \rho} \mathbf{v}_H(j)\right]. \tag{3.10}$$

For the problem, we denote by A_H and T_H the nonlinear operator and its corresponding contraction operator, respectively. Using singular perturbation techniques (see [YZ13]) it can be

shown that under the assumptions made in this chapter the following result holds,

$$\mathbf{v}_H^\star(k) \to \mathbf{v}_h^\star(i), \quad \forall i \in \mathcal{X}_k \quad \text{as} \quad \epsilon \to 0,$$

where $\mathbf{v}_H^\star$ denotes the solution of (3.9) and $\mathbf{v}_h^\star$ denotes the solution of (3.1). Also,

$$|\mathbf{v}_H^\star(k) - \mathbf{v}_h^\star(i)| = \mathcal{O}(\epsilon) \quad \text{for} \quad i \in \mathcal{X}_k. \tag{3.11}$$

3.3 Computational Complexity of Multiscale Markov Decision Processes

In this section we review the complexity of value iteration for the MMDP model introduced in the previous section. The purpose of this section is twofold. Firstly, the complexity of MDPs in continuous time has not received as much attention as that of their discrete-time counterparts. Even though the complexity results here are new, they are straightforward generalizations of results from discrete time. The second and main purpose of this section is to point out that the convergence rate of value iteration becomes arbitrarily bad when ϵ becomes small. We believe that this insight is an important consideration when designing algorithms for multiscale processes. Previously, it was claimed that the coarse model might have lower complexity because it has fewer states than the fine model. Here we show an additional and (as discussed later on) more important advantage is that the coarse model is better conditioned since it does not depend on ϵ. We also show that the complexity results below are tight.

3.3.1 Value Iteration

Value iteration is one of the first methods to be proposed to solve dynamic programming problems. Value iteration is used to compute the value function. After the value function,

$\mathbf{v}_h^\star(x)$, is obtained, the optimal policy $\mathbf{u}_h^\star$ can be obtained by,

$$\mathbf{u}_h^\star(i) \in \arg\min_{a \in \mathcal{A}_i^h} \left[\frac{G^h(i,a)}{|q_{ii}^h(a)| + \rho} + \sum_{j \neq i} \frac{q_{ij}^h(a)}{|q_{ii}^h(a)| + \rho} \mathbf{v}_h^\star(j) \right].$$

Therefore, solving the HJB equation is equivalent to solving for the optimal policy $\mathbf{u}_h^\star$ [Ber07]. State-of-the-art deterministic methods for solving HJB equations fall into three broad categories: linear programming, policy iteration, and value iteration [Pow11]. In this chapter, value iteration is applied to solve the HJB equation even though the central idea of this chapter can be applied to all three of the methods. The extension of the proposed framework to the stochastic case (e.g. to Approximate Dynamic Programming techniques) is beyond the scope of the current chapter. Value iteration is simply defined as,

$$\mathbf{v}_h^{k+1} = T_h \mathbf{v}_h^k. \tag{3.12}$$

The nonlinear operator T_h was defined in (3.8) and it is well known that it is a contraction mapping,

$$\|T_h \mathbf{v}_1 - T_h \mathbf{v}_2\|_\infty \leq \alpha_h \|\mathbf{v}_1 - \mathbf{v}_2\|_\infty, \tag{3.13}$$

where $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^N$, α_h is the Lipschitz constant, $0 < \alpha_h < 1$. For the MDP model we study in this chapter the Lipschitz constant is given by,

$$\alpha_h = \max_{i \in \mathcal{X}^h, a \in \mathcal{A}_i^h} \frac{|q_{ii}^h(a)|}{|q_{ii}^h(a)| + \rho}. \tag{3.14}$$

The Lipschitz constant above is derived in [YZ13], where the HJB equation (3.6) is reformulated as an HJB equation for discrete time problems with a discount factor α_h in (3.14). For discrete time problems, the discount rate itself is the Lipschitz constant of the problem [Pow11]. Using the Banach fixed point theorem [TBI97], it can be shown that for an initial guess $\mathbf{v}_h^0$, one can compute the solution of the HJB equation by iteratively applying T_h on $\mathbf{v}_h^0$,

$$\mathbf{v}_h^{\tau_h} \triangleq T_h^{\tau_h} \mathbf{v}_h^0 = \underbrace{(T_h \circ T_h \circ \cdots \circ T_h)}_{\tau_h \text{ copies}} \mathbf{v}_h^0 \to \mathbf{v}_h^\star \quad \text{as} \quad \tau_h \to \infty.$$

The Lipschitz constant α_h is an upper bound for the convergence rate of the value iteration algorithm. In particular,

$$\|\mathbf{v}_h^\star - \mathbf{v}_h^{\tau_h}\|_\infty \leq \alpha_h \|\mathbf{v}_h^\star - \mathbf{v}_h^{\tau_h - 1}\|_\infty \leq \alpha_h^{\tau_h} \|\mathbf{v}_h^\star - \mathbf{v}_h^0\|_\infty. \tag{3.15}$$

A smaller α_h guarantees a faster convergence rate for the algorithm. Other than equation (3.15), we will make use of the following well-known properties of contraction mappings,

$$\begin{aligned} \|\mathbf{v}_h^\star - \mathbf{v}_h^{\tau_h}\|_\infty &\leq \frac{\alpha_h}{1-\alpha_h} \|\mathbf{v}_h^{\tau_h} - \mathbf{v}_h^{\tau_h - 1}\|_\infty & (3.16) \\ &\leq \frac{\alpha_h^{\tau_h}}{1-\alpha_h} \|\mathbf{v}_h^1 - \mathbf{v}_h^0\|_\infty. & (3.17) \end{aligned}$$

3.3.2 Model Assumptions

In this section, we state our assumptions, and these will hold throughout the chapter. Some of our results will be asymptotic and will rely on the assumption that the problem has multiscale structure stronger than certain threshold ϵ_0. To be precise we assume that $\epsilon > 0$ is small enough such that the value function of the fine model (3.1) and its corresponding coarse model satisfy the following inequality,

$$|\mathbf{v}_H^\star(k) - \mathbf{v}_h^\star(i)| \leq \tilde{K}\epsilon, \ \forall \epsilon \in (0, \epsilon_0), \ \forall i \in \mathcal{X}_k, \tag{3.18}$$

for some constants $\tilde{K}$ and $\epsilon_0 < 1$. That this inequality holds for an ϵ small enough follows from Theorem 7.10 (page 273) in [YZ13].

Our second main assumption is that the objective function is bounded. We will assume that there exists a constant ζ such that,

$$0 \leq G^h(x_h, a_h) \leq \zeta, \ \forall x_h \in \mathcal{X}^h, a_h \in \mathcal{A}^h.$$

The bounded assumption is needed to avoid trivialities. Since $G^h(\cdot, \cdot)$ is assumed to be bounded,

the value function should also be bounded; in other words, there exists a constant $\hat{K}$ such that,

$$0 \leq \|\mathbf{v}_h^\star\|_\infty \leq \hat{K},\ 0 \leq \|\mathbf{v}_H^\star\|_\infty \leq \hat{K}.$$

Without loss of generality we will assume that the initial guess $\mathbf{v}_h^0$ and $\mathbf{v}_H^0$ are both zero vectors of the appropriate dimensions. Finally, to simplify our notation, instead of using $\hat{K}$ and $\tilde{K}$, we will directly use $K \triangleq \max\{\hat{K}, \tilde{K}\}$, where $\tilde{K}$ is defined in equation (3.18). With these two assumptions, we obtain

$$0 = \|\mathbf{v}_h^0\|_\infty \leq \|\mathbf{v}_h^1\|_\infty \leq \|\mathbf{v}_h^2\|_\infty \leq \cdots \leq \|\mathbf{v}_h^\star\|_\infty \leq K, \tag{3.19}$$

$$0 = \|\mathbf{v}_H^0\|_\infty \leq \|\mathbf{v}_H^1\|_\infty \leq \|\mathbf{v}_H^2\|_\infty \leq \cdots \leq \|\mathbf{v}_H^\star\|_\infty \leq K. \tag{3.20}$$

The above inequalities follow from the fact that both operators T_h and T_H are monotone contraction operators [Ber07]. Also, the above assumption yield

$$\begin{aligned} \|\mathbf{v}_h^\star - \mathbf{v}_h^i\|_\infty &\leq K,\ \|T_h\mathbf{v}_h^i - \mathbf{v}_h^i\|_\infty \leq K,\ \ \forall \mathbf{v}_h^i \quad \text{for } i = 1, 2, \cdots, \\ \|\mathbf{v}_H^\star - \mathbf{v}_H^i\|_\infty &\leq K,\ \|T_H\mathbf{v}_H^i - \mathbf{v}_H^i\|_\infty \leq K,\ \forall \mathbf{v}_H^i \quad \text{for } i = 1, 2, \cdots. \end{aligned}$$

Notice that (3.18) and $\tilde{K} \leq K$ gives

$$|\mathbf{v}_H^\star(k) - \mathbf{v}_h^\star(i)| \leq K\epsilon,\ \forall i \in \mathcal{X}_k. \tag{3.21}$$

3.3.3 Complexity

In this section we discuss the complexity of continuous time MDPs. The complexity result in this section is a variant of the existing discrete time result [CT91]. We use $\delta > 0$ to denote the convergence tolerance for the value iteration algorithm, i.e. the algorithm terminates when,

$$\|\mathbf{v}_h^\star - \mathbf{v}_h^{\tau_h}\| < \delta. \tag{3.22}$$

The parameter $\delta > 0$ is user specified and since T_h is a contraction mapping, for large enough τ_h, the above inequality can be satisfied. A more interesting question is how large τ_h should be to guarantee that (3.22) holds. We answer this question in the lemma below by providing an upper bound and then we give an example to show that this bound is tight.

Lemma 3.1 *The number of iterations in the value iteration algorithm is bounded by*

$$\tau_h \leq \max\left\{\frac{\log\left(\frac{K}{(1-\alpha_h)\delta}\right)}{|\log \alpha_h|}, 0\right\}, \tag{3.23}$$

where $K = \max\{\hat{K}, \tilde{K}\}$.

Proof We use equation (3.17),

$$\|\mathbf{v}_h^\star - \mathbf{v}_h^{\tau_h}\|_\infty \leq \frac{\alpha_h^{\tau_h}}{1-\alpha_h}\|\mathbf{v}_h^1 - \mathbf{v}_h^0\|_\infty \leq \frac{\alpha_h^{\tau_h}}{1-\alpha_h}K, \tag{3.24}$$

where we used the fact that $\|T_h\mathbf{v}_h^0 - \mathbf{v}_h^0\| < K$. We then select τ_h' such that

$$\frac{\alpha_h^{\tau_h'}}{1-\alpha_h}K = \delta.$$

Rearranging the preceding equation, we obtain the following expression

$$\tau_h' = \frac{\log\left(\frac{K}{(1-\alpha_h)\delta}\right)}{|\log \alpha_h|}.$$

Since $\mathbf{v}_h^{\tau_h'}$ guarantees the desired accuracy, we have $\tau_h \leq \max\{\tau_h', 0\}$. ∎

Equation (3.23) gives an upper bound for the number of iterations we need when using value iteration. The complexity of the value iteration algorithm can be easily derived from Lemma 3.1. The complexity model we consider in this chapter is consistent with [CT91], where each arithmetic operation or comparison is considered to cost one unit of computation.

Theorem 3.2 *For $\mathcal{MDP}(N, L)$, the worst–case complexity for the value iteration algorithm in* (3.12) *is*

$$\mathcal{O}\left(\max\left\{\frac{\log\left(\frac{1}{(1-\alpha_h)\delta}\right)}{|\log\alpha_h|}, 0\right\}N^2L\right). \tag{3.25}$$

Proof For the contraction operator in equation (3.8) and for an MDP problem $\mathcal{MDP}(N, L)$, the worst-case complexity of computing $T_h\mathbf{v}_h$ is $\mathcal{O}(N^2L)$. Since the total complexity of the algorithm is the number of iterations multiplied by the cost per iteration, we obtain the complexity result in (3.25) by applying Lemma 3.1. ∎

A natural question to ask is whether the complexity result in Theorem (3.2) is tight. We end this section by showing that indeed the bound is tight.

Remark

Consider an instance of $\mathcal{MDP}(N, L)$ that satisfies the following

- $|\mathcal{A}_i^h| = 1, \forall i \in \mathcal{X}^h$. This assumption means that the corresponding HJB equation is a linear equation.
- The cost function $G^h(i, a_i) = g > 0, \forall i \in \mathcal{X}^h$ is a constant. Therefore, the value function $\mathbf{v}_h^\star(i) = \mathbf{v}^\star$ is also a positive constant.
- $q_{ii}^h = q, \forall i \in \mathcal{X}^h$ i.e. each state has the same jump rate.
- The initial guess $\mathbf{v}_h^0 = 0$ is a zero vector, so $\mathbf{v}_h^0 < T_h\mathbf{v}_h^0 < T_h^2\mathbf{v}_h^0 < \cdots < \mathbf{v}_h^\star$.

Given the assumptions above, it follows that $\mathbf{v}_h^\tau \triangleq T_h^\tau \mathbf{v}_h^0$ are all constants, i.e. $\mathbf{v}_h^\tau(i) = \mathbf{v}^\tau, \forall i \in \mathcal{X}^h, \tau \in \mathbb{Z}^+$. Consider the error reduction rate between iteration τ and $\tau + 1$,

$$\begin{aligned}
\|\mathbf{v}_h^\star - \mathbf{v}_h^{\tau+1}\|_\infty &= \|\mathbf{v}_h^\star - T_h^{\tau+1}\mathbf{v}_h^0\|_\infty \\
&= \|T_h\mathbf{v}_h^\star - T_h\mathbf{v}_h^\tau\|_\infty \\
&= \max_{i\in\mathcal{X}^h}\left|\frac{g}{|q|+\rho} + \sum_{j\neq i}\frac{q_{ij}^h}{|q|+\rho}\mathbf{v}^\star - \frac{g}{|q|+\rho} - \sum_{j\neq i}\frac{q_{ij}^h}{|q|+\rho}\mathbf{v}^\tau\right| \\
&= \max_{i\in\mathcal{X}^h}\left|\sum_{j\neq i}\frac{q_{ij}^h}{|q|+\rho}(\mathbf{v}^\star - \mathbf{v}^\tau)\right| \\
&= \max_{i\in\mathcal{X}^h}\left|\frac{|q|}{|q|+\rho}\right| \|\mathbf{v}_h^\star - \mathbf{v}_h^\tau\|_\infty \\
&= \alpha_h\|\mathbf{v}_h^\star - \mathbf{v}_h^\tau\|_\infty.
\end{aligned}$$

Therefore, in this particular instance of an $\mathcal{MDP}(N, L)$, the number of iterations is exactly the one given by Lemma 3.1.

3.3.4 Convergence Rate and Complexity for Multiscale Markov Decision Processes

The main motivation for stating Theorem 3.2 is that it will enable us to make precise statements concerning the computational advantages of the coarse model derived in Section 3.2.2. Using the results derived above we show that *the principal benefit of the coarse model is not that the number of states is less, but that the rate of convergence is much higher* (provided that scale separation is present). In fact, the complexity of the coarse model when no scale separation is present, i.e. $\epsilon \approx 1$, is greater than that of the original model. The lemma below shows that $\mathcal{MMDP}(\epsilon, n, m, L)$ becomes ill conditioned as ϵ approaches zero. Note that there is no standard definition for an ill conditioned MDP. However, in the context of this chapter an MDP is ill conditioned if the contraction modulus of value iteration is approximately equal to one. The lemma below shows that this indeed is the case if the MDP is singularly perturbed.

Lemma 3.3 *For $\mathcal{MMDP}(\epsilon, n, m, L)$ with Lipschitz constant α_h,*

$$\alpha_h \to 1 \quad as \quad \epsilon \to 0.$$

Proof In $\mathcal{MMDP}(\epsilon, n, m, L)$, the Lipschitz constant has the form

$$\alpha_h = \max_{i \in \mathcal{X}^h, a \in \mathcal{A}_i^h} \frac{\left|\frac{1}{\epsilon}\hat{q}_{ii}(a) + w_{ii}(a)\right|}{\left|\frac{1}{\epsilon}\hat{q}_{ii}(a) + w_{ii}(a)\right| + \rho} \to 1 \quad \text{as} \quad \epsilon \to 0.$$

∎

When ϵ is small the guaranteed improvement in each iteration is almost zero for the fine model. On the other hand, for the coarse model the corresponding Lipschitz constant is given by,

$$\alpha_H = \max_{i \in \mathcal{X}^H, a_H \in \mathcal{A}_i^H} \frac{\left|q_{ii}^H(a_H)\right|}{|q_{ii}^H(a_H)| + \rho}.$$

Crucially, α_H is independent of the multiscale structure of the original model. Therefore, there exists a $\epsilon^\star$ such that

$$\alpha_H \leq \alpha_h \quad \text{for} \quad \epsilon \leq \epsilon^\star.$$

In other words, the guaranteed convergence behavior of coarse model is superior to that of the fine model when ϵ is small enough. We end this section by comparing the computational complexity associated with the two models,

$$\begin{aligned} &\text{Fine Model, } \mathcal{MMDP}(\epsilon, n, m, L) : \mathcal{O}\left(\max\left\{\frac{\log\left(\frac{1}{(1-\alpha_h)\delta}\right)}{|\log \alpha_h|}, 0\right\}(mn)^2 L\right), \\ &\text{Coarse Model, } \mathcal{MDP}(m, L^n) : \mathcal{O}\left(\max\left\{\frac{\log\left(\frac{1}{(1-\alpha_H)\delta}\right)}{|\log \alpha_H|}, 0\right\} m^2 L^n\right). \end{aligned} \tag{3.26}$$

If $\epsilon \approx 1$, then there are no benefits to aggregating the model using the singular perturbation approach. Indeed the preceding equation shows the coarse model has an exponential dependence

on n that is not present in the original model. We will discuss ways to alleviate this issue in Section 3.6. Finally in the setting of this chapter as $\epsilon \to 0$, the complexity of the fine model goes to infinity.

3.4 Analysis of the Full Approximation Scheme

The conventional way to exploit multiresolution structure of a model is the Full Approximation Scheme (FAS) (see e.g. [Hac03]). The FAS is an extension of the multigrid scheme to non-linear problems. Algorithms based on multigrid are in spirit close to the scheme we propose in this chapter. In other words, multigrid algorithms try to solve the fine model by considering a hierarchy of approximate models. We also develop a scheme that fits within this general principle but we propose a different way to couple the models together than the one used in FAS. We stress that the theory around the FAS is still valid, and that the convergence proof developed in [Hac03] can be used to show that the FAS will converge to the solution of $\mathcal{MMDP}(\epsilon, n, m, L)$. However, we will use a simple numerical example to illustrate the point that even though convergence is guaranteed, the rate of convergence is likely to be worse than just solving the fine model with the single level value iteration algorithm. In Section 3.5, we show how to overcome this problem of the FAS by proposing a different way of incorporating information from the coarse model to the iterations of the fine model. We refer the interested reader to the tutorial in [BHM00] for an introduction to multigrid and the FAS. The FAS is rigorously developed in [Hac03]. In this section, we just mention some of the key ideas behind multigrid and FAS in order to understand how the existing framework is likely to fail for multiscale MDPs.

3.4.1 Prolongation and Restriction

The first step in the development of the FAS is the definition of the prolongation and restriction operators. The prolongation operator is used to transfer solutions from the coarse model to the fine model. We use $\mathbf{I}_H^h$ and $\mathbf{I}_h^H$ to denote the prolongation and restriction operators, respectively.

Typically, they are linear operators and in this chapter we take $\mathbf{I}_H^h$ and $\mathbf{I}_h^H$ to be constant matrices. The exact definition of these operators is problem dependent. For the class of models we consider in this chapter it is natural to define $\mathbf{I}_H^h$ and $\mathbf{I}_h^H$ based on the asymptotic properties of the fine and coarse models. The prolongation operator is given by

$$\mathbf{I}_H^h = \text{diag}(\underbrace{\mathbf{1}_{n\times 1}, \mathbf{1}_{n\times 1}, \ldots, \mathbf{1}_{n\times 1}}_{\text{m copies}}) \in \mathbb{R}^{nm\times m}.$$

The choice of $\mathbf{I}_H^h$ is based on equation (3.21), which shows that the value functions are asymptotically the same for the states that are in the same block. The definition of the restriction operator is not as straightforward as that of $\mathbf{I}_H^h$. There is no obvious property to approximate $\mathbf{v}_H$ by $\mathbf{v}_h$. However, a natural choice that can be rigorously justified (see Section 3.5) is to restrict $\mathbf{v}_h$ into the same size as $\mathbf{v}_H$ using the stationary distribution of each block. Let $\boldsymbol{\varphi}_i$ denote the column vector for the stationary distributions associated with block i. We define the restriction operator as follows

$$\mathbf{I}_h^H = \text{diag}(\boldsymbol{\varphi}_1^T, \boldsymbol{\varphi}_2^T, \ldots, \boldsymbol{\varphi}_m^T) \in \mathbb{R}^{m\times nm}. \tag{3.27}$$

In other words, we "compress" the values of the value function in block i by forming a convex combination with the elements in each of the blocks. Notice that $\mathbf{Q}_h^\epsilon(\mathbf{u}_h)$ depends on the policy $\mathbf{u}_h$ and so there exist many different stationary distributions for each block. To address this problem, we select $\mathbf{Q}_h^\epsilon(\tilde{\mathbf{u}}_h)$ with $\tilde{\mathbf{u}}_h$ as the best policy for the current incumbent solution at iteration τ, $\mathbf{v}_h^\tau$. That is, we select $\tilde{\mathbf{u}}_h$ such that

$$\tilde{\mathbf{u}}_h(i) \in \arg\min_{a\in\mathcal{A}_i^h} \left[\frac{G^h(i,a)}{|q_{ii}^h(a)| + \rho} + \sum_{j\neq i} \frac{q_{ij}^h(a)}{|q_{ii}^h(a)| + \rho} \mathbf{v}_h^\tau(j) \right],$$

for the current solution $\mathbf{v}_h^\tau$, and apply equation (3.27) with $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \ldots, \boldsymbol{\varphi}_m$ as the stationary distributions of blocks $1, 2, \ldots, m$ in $\mathbf{Q}_h^\epsilon(\tilde{\mathbf{u}}_h)$. It follows from (3.21) that there exists a constant K such that,

$$\|\mathbf{v}_h^\star - \mathbf{I}_H^h \mathbf{v}_H^\star\|_\infty \leq K\epsilon. \tag{3.28}$$

3.4.2 The FAS Algorithm

With the definitions of $\mathbf{I}_h^H$ and $\mathbf{I}_H^h$ provided above we are now in a position to fully specify the FAS. The main idea of the FAS is simple and we describe it as a solution algorithm for the following general nonlinear equation,

$$A_h(\mathbf{v}_h) = \mathbf{f}_h. \tag{3.29}$$

In our case A_h is given in (3.7) and $\mathbf{f}_h$ can be taken to be zero (at the finest level). Given an incumbent solution $\hat{\mathbf{v}}_h$, we can proceed to compute the exact correction for $\hat{\mathbf{v}}_h$ so that it solves (3.29). That is we compute an $\mathbf{e}_h^\star$ such that,

$$A_h(\hat{\mathbf{v}}_h + \mathbf{e}_h^\star) = \mathbf{f}_h.$$

Of course the preceding nonlinear equation is just as hard as the original problem. The idea behind the FAS is instead of computing $\mathbf{e}_h^\star$ using the fine model, an approximation of $\mathbf{e}_h^\star$ is computed using the coarse model by solving the following correction problem

$$A_H(\mathbf{I}_h^H \hat{\mathbf{v}}_h + \mathbf{e}_H) = \mathbf{d}_H,$$

where

$$\mathbf{d}_H \triangleq A_H(\mathbf{I}_h^H \hat{\mathbf{v}}_h) - s\mathbf{I}_h^H(A_h(\hat{\mathbf{v}}_h) - \mathbf{f}_h),$$

for some stepsize s. The existences of s and d_H establish a useful relation between the fine and coarse models [Hac03]. Finally, we complete the correction $\hat{\mathbf{v}}_h + 1/s\mathbf{I}_H^h\mathbf{e}_H$. Figure 3.1 illustrates the main steps of FAS. We first compute $\hat{\mathbf{v}}_h$ by τ applications of T_h. We then restrict the solution to the coarse scale and perform some iterations in the coarse scale to obtain an approximate correction $\mathbf{e}_H$. We then prolongate the error correction term to the fine model and continue performing iterations at the fine scale. The addition of the prolongated error $1/s\mathbf{I}_H^h\mathbf{e}_H$ to the current solution $\hat{\mathbf{v}}_h$ can lead to faster convergence rates than just using the fine model. The step size s is needed because this is a nonlinear problem. Obviously it is possible to have

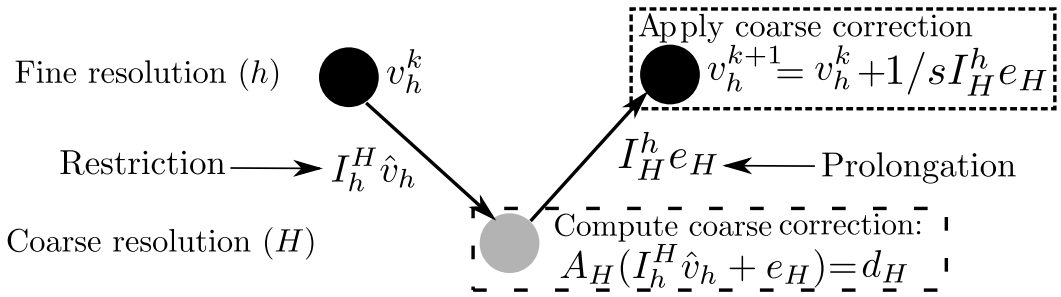


Figure 3.1: The Full Approximation Scheme (FAS)

more than one level. The full details of the algorithm are given in [Hac03].

3.4.3 Numerical Example from a Multiscale Manufacturing System

The full approximation scheme appears to be a good method to solve multiscale MDPs. It solves both the problems we set out to address in this chapter, i.e. it uses the coarse model that is better conditioned, but still computes an exact solution for the original model. However, we will show using a simple example that for MDPs with multiscale structure the FAS can have an extremely slow convergence rate. The algorithm still converges but it is much slower than simple value iteration. We propose a solution to this issue in the next section.

The example we use is not a contrived model but a simple and widely used model motivated by a manufacturing application. The model is described in [YZ13] and concerns the control of a manufacturing process with two machines. Each machine has two states, up and down. We use 1 to denote that machine is working, and 0 for the state when the machine is broken down. The total number of states in the system are $\{(1,1),(0,1),(1,0),(0,0)\}$, where (i,j) represents the state where machine 1 in state i and machine 2 in state j. In this manufacturing process, the state of each machine depends on the action a, which is the rate of preventive maintenance. The overall goal of the problem is to pick the policy $\mathbf{u}$ such that the machines do not break down often while the cost of maintenance is not too high. The model further assumes that the two machines have failure rates that occur in different timescales. To reflect this assumption

the following generator is used,

$$\mathbf{Q}_h^{\epsilon}(a) = \frac{1}{\epsilon}\begin{bmatrix} -\lambda_1(a) & \lambda_1(a) & 0 & 0 \\ \mu_1(a) & -\mu_1(a) & 0 & 0 \\ 0 & 0 & -\lambda_1(a) & \lambda_1(a) \\ 0 & 0 & \mu_1(a) & -\mu_1(a) \end{bmatrix} + \begin{bmatrix} -\lambda_2(a) & 0 & \lambda_2(a) & 0 \\ 0 & -\lambda_2(a) & 0 & \lambda_2(a) \\ \mu_2(a) & 0 & -\mu_2(a) & 0 \\ 0 & \mu_2(a) & 0 & -\mu_2(a) \end{bmatrix}, \tag{3.30}$$

where $\lambda_1(a)/\epsilon$ and $\mu_1(a)/\epsilon$ are the breakdown and repair rates for machine 1, and $\lambda_2(a)$ and $\mu_2(a)$ for machine 2, respectively. As we can see, equation (3.30) is in the same form of equation (3.3). Intuitively, the more preventive maintenance is performed on a machine, then the machine is more likely to stay in state 1. For this simple example, we assume $\mathcal{X}^h = \{1, 2, 3, 4\}$, $a \in \{1, 2, \ldots, 5\}$, and

$$\lambda_1(a) = 1/a, \qquad \mu_1(a) = a^2,$$
$$\lambda_2(a) = 3/a, \qquad \mu_2(a) = 3a.$$

A higher value of a would ensure the system is online more often. Of course the more maintenance is performed the higher the costs. To reflect this trade-off we use the following objective function,

$$G^h(x, a) = x^2 + a^2 \quad , \quad \forall x \in \mathcal{X}^h \quad , \quad a \in \{1, 2, 3, 4, 5\}.$$

We used the FAS scheme to solve the infinite horizon version of the model described above. We plot the iteration history of the FAS against the exact solution of the fine model in Figure 3.2. The exact solution was obtained using linear programming. It may initially appear that the FAS has a similar performance as the value iteration algorithm when applied to the fine model. In closer inspection this is not the case. To illustrate this point we zoom in to the part of the computation where the FAS jumps to the coarse model (iteration 5000 in this example). From Figure 3.2 we see that actually no useful computation is performed during the coarse iterations. We point out that we tried different strategies for updating the step size as well as experimenting with the different parameters (such as when to jump to the coarse model and

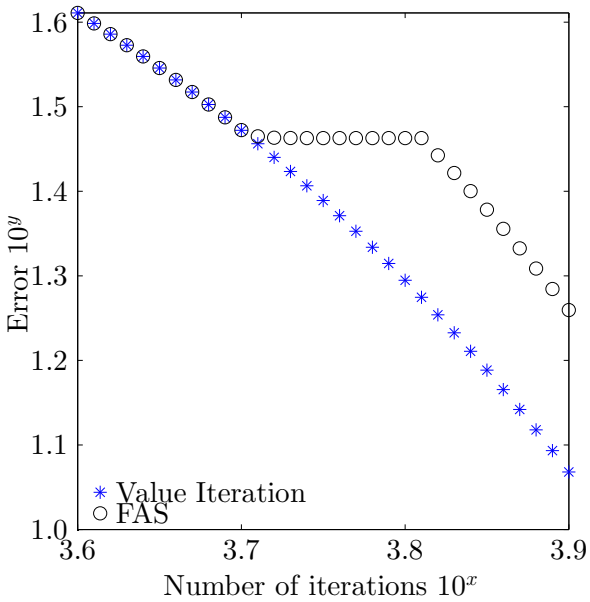


Figure 3.2: Performance of the FAS, $\epsilon = 10^{-2}$, initial number of iterations in the fine model: 5000, stepsize $s = 1$. The figure shows that no useful computation is performed by the FAS during the coarse iterations.

how many iterations to perform there). The numerical performance of the FAS is disappointing. It appears that the correction does not help the incumbent solution to get closer to the exact solution. In the next section, we discuss some possible reasons why the FAS may not be suitable for solving MDPs.

3.4.4 Lack of progress in the coarse iterations of the FAS

In this section, we provide some possible explanations as to why the coarse iterations of the FAS do not provide useful corrections to the current fine solution. To simplify the analysis, suppose that there is only one policy $\mathbf{u}_h$. We drop the dependence on $\mathbf{u}_h$ from $\mathbf{Q}_h^\epsilon$ and $G^h(\cdot)$. It is easy to generalize our conclusions to the case when the policy space is richer. With these simplifications our model reduces to the following linear equation

$$(A_h \mathbf{v}_h)(i) \triangleq \frac{G^h(i)}{|q_{ii}^h| + \rho} + \sum_{j \neq i} \frac{q_{ij}^h}{|q_{ii}^h| + \rho} \mathbf{v}_h(j) - \mathbf{v}_h(i) = 0,$$

which can be written more compactly as $\mathcal{L}_h \mathbf{v}_h = \mathbf{b}_h$, where

$$\mathcal{L}_h = \begin{bmatrix} -1 & \frac{q^h_{12}}{|q^h_{11}|+\rho} & \cdots & \frac{q^h_{1nm}}{|q^h_{11}|+\rho} \\ \frac{q^h_{21}}{|q^h_{22}|+\rho} & -1 & \cdots & \frac{q^h_{2nm}}{|q^h_{22}|+\rho} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{q^h_{nm1}}{|q^h_{nmnm}|+\rho} & \cdots & \cdots & -1 \end{bmatrix}, \ \mathbf{b}_h = \begin{bmatrix} -\frac{G^h(1)}{|q^h_{11}|+\rho} \\ -\frac{G^h(2)}{|q^h_{22}|+\rho} \\ \vdots \\ -\frac{G^h(nm)}{|q^h_{nmnm}|+\rho} \end{bmatrix}.$$

The corresponding coarse model also reduces to the linear system $\mathcal{L}_H \mathbf{v}_H = \mathbf{b}_H$, where

$$\mathcal{L}_H = \begin{bmatrix} -1 & \frac{q^H_{12}}{|q^H_{11}|+\rho} & \cdots & \frac{q^H_{1m}}{|q^H_{11}|+\rho} \\ \frac{q^H_{21}}{|q^H_{22}|+\rho} & -1 & \cdots & \frac{q^H_{2m}}{|q^H_{22}|+\rho} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{q^H_{m1}}{|q^H_{mm}|+\rho} & \cdots & \cdots & -1 \end{bmatrix}, \ \mathbf{b}_H = \begin{bmatrix} -\frac{G^h(1)}{|q^h_{11}|+\rho} \\ -\frac{G^h(2)}{|q^h_{22}|+\rho} \\ \vdots \\ -\frac{G^h(m)}{|q^h_{mm}|+\rho} \end{bmatrix}.$$

Given an incumbent solution $\mathbf{v}_h$ and the exact correction $\mathbf{e}^\star_h$, we have

$$\mathcal{L}_h(\mathbf{v}_h + \mathbf{e}^\star_h) = \mathbf{b}_h.$$

From which we obtain the following,

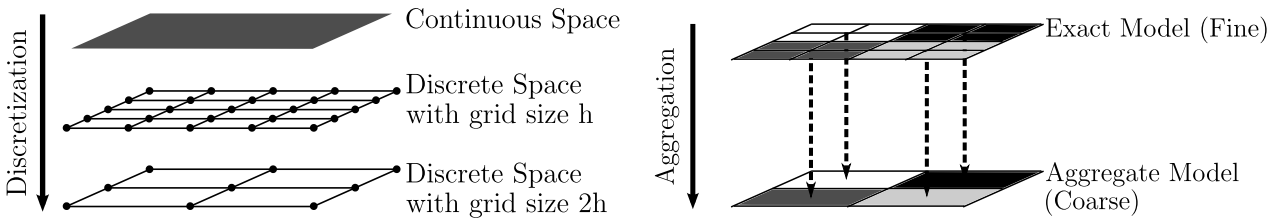
$$\mathbf{e}^\star_h = \mathcal{L}_h^{-1}\mathbf{b}_h - \mathbf{v}_h = \mathcal{L}_h^{-1}\mathbf{b}_h - \mathcal{L}_h^{-1}\mathcal{L}_h\mathbf{v}_h = \mathcal{L}_h^{-1}(\mathbf{b}_h - \mathcal{L}_h\mathbf{v}_h).$$

Letting $\mathbf{d}_h = \mathcal{L}_h\mathbf{v}_h - \mathbf{b}_h$ we obtain,

$$\mathbf{e}^\star_h = \mathcal{L}_h^{-1}(\mathbf{b}_h - \mathcal{L}_h\mathbf{v}_h) = -\mathcal{L}_h^{-1}\mathbf{d}_h.$$

The FAS approximates $\mathbf{e}^\star_h$ by computing a correction in the coarse model. For the correction problem, we let $\mathbf{v}_H \triangleq \mathbf{I}^H_h\mathbf{v}_h$, $\mathbf{d}_H \triangleq \mathcal{L}_H\mathbf{v}_H - s\mathbf{I}^H_h\mathbf{d}_h$, and compute,

$$\tilde{\mathbf{v}}_H \triangleq \mathcal{L}_H^{-1}\mathbf{d}_H = \mathbf{v}_H - s\mathcal{L}_H^{-1}\mathbf{I}^H_h\mathbf{d}_h.$$



(a) Traditional Discretization in Differential Equations (b) Aggregation for MMDP

Figure 3.3: Different ideas between discretization and aggregation

Then, the correction is

$$\tilde{\mathbf{e}}_h \triangleq \frac{1}{s}\mathbf{I}_H^h(\tilde{\mathbf{v}}_H - \mathbf{v}_H) = \frac{1}{s}\mathbf{I}_H^h(-s\mathcal{L}_H^{-1}\mathbf{I}_h^H\mathbf{d}_h) = -\mathbf{I}_H^h\mathcal{L}_H^{-1}\mathbf{I}_h^H\mathbf{d}_h.$$

In the case when $\mathbf{I}_H^h\mathcal{L}_H^{-1}\mathbf{I}_h^H \approx \mathcal{L}_h^{-1}$, the correction problem provides a good approximation of $\mathbf{e}_h^\star$. Traditionally, multigrid methods are aimed towards the solution of differential equations and discretize a continuous space into different grid sizes. The assumption that $\mathbf{I}_H^h\mathcal{L}_H^{-1}\mathbf{I}_h^H \approx \mathcal{L}_h^{-1}$ usually holds because $\mathcal{L}_H$ and $\mathcal{L}_h$ are discrete operators derived from the same continuous operator. However, in the case of MMDP, our coarse model is obtained by averaging each block with its stationary distribution, which makes $\mathcal{L}_H$ different from $\mathcal{L}_h$. Also, as $\epsilon \to 0$, $\mathcal{L}_H$ remains unchanged but this is not the case for $\mathcal{L}_h$. Figure 3.3 illustrates the differences between the two kinds of problems. In order to give some deeper insights into the numerical challenges caused by $\mathcal{MMDP}$ models we consider the example from the previous subsection when we have a single action, $a = 1$. In this simple setting we can compute $\mathcal{L}_h$ and $\mathcal{L}_H$ exactly and see the differences between the two operators. We performed this analysis with the parameters described in the previous section and found that the difference between $\mathbf{I}_H^h\mathcal{L}_H^{-1}\mathbf{I}_h^H$ and $\mathcal{L}_h^{-1}$ is very large especially for smaller ϵ. The difference between the two operators was measured using the spectral norm. We also computed the eigenvalues of $\mathcal{L}_h^{-1}$ in closed form. The resulting expression are long but can be easily computed using a symbolic mathematics package. From our calculations we observed that as ϵ approaches zero the matrix $\mathcal{L}_h$ becomes nearly singular and therefore its inverse does not exist. In contrast, $\mathcal{L}_H$ is independent of ϵ and its inverse always exists. This explains why the difference between $\mathbf{I}_H^h\mathcal{L}_H^{-1}\mathbf{I}_h^H$ and $\mathcal{L}_h^{-1}$ is very large for small ϵ. In addition, we found that this difference, when measured using the spectral norm, is a log-linear function of ϵ. This indicates FAS is not suitable for our problem because the basic

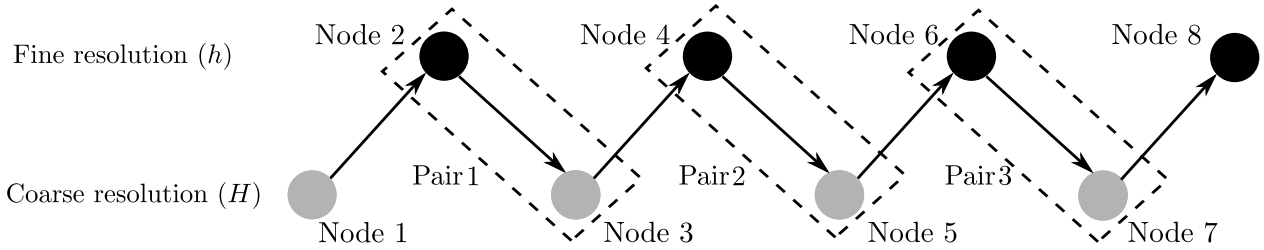


Figure 3.4: The Alternating Multiresolution Scheme (AMS)

motivation of FAS does not fit with the setup of $\mathcal{MMDP}$ models. In the next section, we will introduce a new scheme, the alternating multiresolution scheme that attempts to address some of these issues.

3.5 The Alternating Multiresolution Scheme

We have already seen in Section 3.4 that the traditional full approximation scheme is not suitable for $\mathcal{MMDP}$. We introduce a new algorithm, the **A**lternating **M**ultiresolution **S**cheme (AMS), to address the low convergence rate of the FAS. One can think of the AMS as a modified version of FAS. In particular, we eliminate the correction problem in the coarse model and replace it with the original coarse problem. The main idea of the AMS is to split all the iterations in the coarse model into many pieces.

In the AMS, neither the coarse model nor the fine model is solved completely once. Instead, we apply the coarse contraction map $\tau_{H,P}$ times to the initial guess of the coarse model, then project the solution to the fine model as an initial guess. We then apply the fine contraction map T_h for $\tau_{h,P}$ times, project it back to the coarse model for $\tau_{H,P}$ iterations to find the approximate error, and so on. The scheme is shown in Figure 3.4. For convenience we number the nodes and pair up one fine iteration node with one coarse iteration node together. Starting with node 1, which is a coarse iteration node, we add 1 to the iteration counter whenever we switch between coarse and fine iterations. With this indexing convention all the odd nodes are iterations with the coarse model, and all the even nodes are iterations with the fine model. For an alternating multiresolution scheme with M nodes, we pair up node $2j$ and node $2j+1$ together, $j = 1, 2, \ldots, (M-2)/2$. A P-AMS denotes the alternating multiresolution scheme

Algorithm 3.1 The Alternating Multiresolution Scheme (P-AMS)

· Start with initial guess $\mathbf{v}_H^0$.
· $\mathbf{v}_H^1 \leftarrow T_H^{\tau_{H,P}} \mathbf{v}_H^0$.
· $\mathbf{v}_h^0 \leftarrow \mathbf{I}_H^h \mathbf{v}_H^1$.
for $p = 1, 2, \ldots, P$ **do**
  · $\mathbf{v}_h^1 \leftarrow T_h^{\tau_{h,P}} \mathbf{v}_h^0$.
  · $\mathbf{v}_H^1 \leftarrow \mathbf{I}_h^H \mathbf{v}_h^1$.
  · $\mathbf{v}_H^2 \leftarrow T_H^{\tau_{H,P}} \mathbf{v}_H^1$.
  · $\mathbf{v}_h^0 \leftarrow \mathbf{v}_h^1 + s\mathbf{I}_H^h \left[(\mathbf{v}_H^2 - \mathbf{v}_H^1)\right]$ (where s is the stepsize).
end for
while $A_h(\mathbf{v}_h^0) \geq \delta$ **do**
  · $\mathbf{v}_h^0 \leftarrow T_h \mathbf{v}_h^0$.
end while

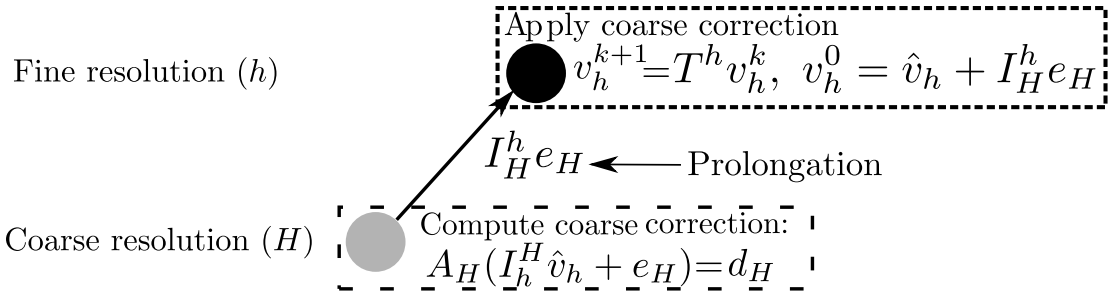


Figure 3.5: The One-way Multiresolution Scheme (OWMS)

with P pairs. Figure 3.4 illustrates the P-AMS with P=3. We state the full algorithm below. In order to have a fully specified algorithm we need to decide $\tau_{h,P}$, $\tau_{H,P}$, and a strategy to pick the step-size parameter s. We discuss how the number of iterations is determined below. We first discuss this issue on a simplified version of AMS before addressing the general case.

3.5.1 One-way Multiresolution Scheme

We begin our analysis of the AMS for the specific case where we only have two nodes. We call this specific scheme the **O**ne-**W**ay **M**ultiresolution **S**cheme (OWMS). In this scheme we solve the coarse model first and prolongate the solution as an initial guess for the fine model. Figure 3.5 illustrates the simplified scheme. The Lemma below gives an upper bound on the number of iterations that need to be performed in the coarse model. The significance of the lemma below is that it provides an estimate of the number of iterations required and relies on known input data.

Lemma 3.4 *The number of iterations required to achieve the following accuracy in the coarse model,*

$$\|\mathbf{v}_H^{\tau_{H,0}} - \mathbf{v}_H^{\tau_{H,0}-1}\|_\infty \leq K\frac{\epsilon(1-\alpha_H)}{\alpha_H}. \tag{3.31}$$

is bounded by,

$$\tau_{H,0} \leq \frac{\log\left(\dfrac{1}{\epsilon(1-\alpha_H)}\right)}{|\log\alpha_H|}. \tag{3.32}$$

In addition when (3.31) is satisfied then $\mathbf{v}_H^{\tau_{H,0}}$ satisfies

$$\|\mathbf{v}_H^\star - \mathbf{v}_H^{\tau_{H,0}}\|_\infty \leq K\epsilon \tag{3.33}$$

where the constant K is defined in (3.21).

Proof Using the contraction property (3.16)

$$\|\mathbf{v}_H^\star - \mathbf{v}_H^{\tau_{H,0}}\|_\infty \leq \frac{\alpha_H}{1-\alpha_H}\|\mathbf{v}_H^{\tau_{H,0}} - \mathbf{v}_H^{\tau_{H,0}-1}\|_\infty \leq \frac{\alpha_H}{1-\alpha_H}K\frac{\epsilon(1-\alpha_H)}{\alpha_H} \leq K\epsilon,$$

In order to find the bound of $\tau_{H,0}$, we select $\tau'_{H,0}$ such that

$$\|\mathbf{v}_H^{\tau'_{H,0}} - \mathbf{v}_H^{\tau'_{H,0}-1}\|_\infty \leq \alpha_H^{\tau'_{H,0}-1}\|\mathbf{v}_H^1 - \mathbf{v}_H^0\|_\infty \leq \alpha_H^{\tau'_{H,0}-1}K = K\frac{\epsilon(1-\alpha_H)}{\alpha_H},$$

and so

$$\alpha_H^{\tau'_{H,0}-1}K = K\frac{\epsilon(1-\alpha_H)}{\alpha_H}, \tag{3.34}$$

$$\tau'_{H,0} = \frac{\log\left(\dfrac{1}{\epsilon(1-\alpha_H)}\right)}{|\log\alpha_H|}. \tag{3.35}$$

Notice that $\tau'_{H,0} > 0$ because both $\epsilon < 1$ and $\alpha_H < 1$. Since $\mathbf{v}_H^{\tau'_{H,0}}$ guarantees the desired accuracy, we have $\tau_{H,0} \leq \tau'_{H,0}$, for $\tau'_{H,0}$ in equation (3.35). ∎

Figure 3.6 illustrates the concept behind the stopping criterion developed in the preceding lemma. The reason we do not compute the exact $\mathbf{v}_H^\star$ is that using $\mathbf{v}_H^{\tau_{H,0}}$ as the initial point for

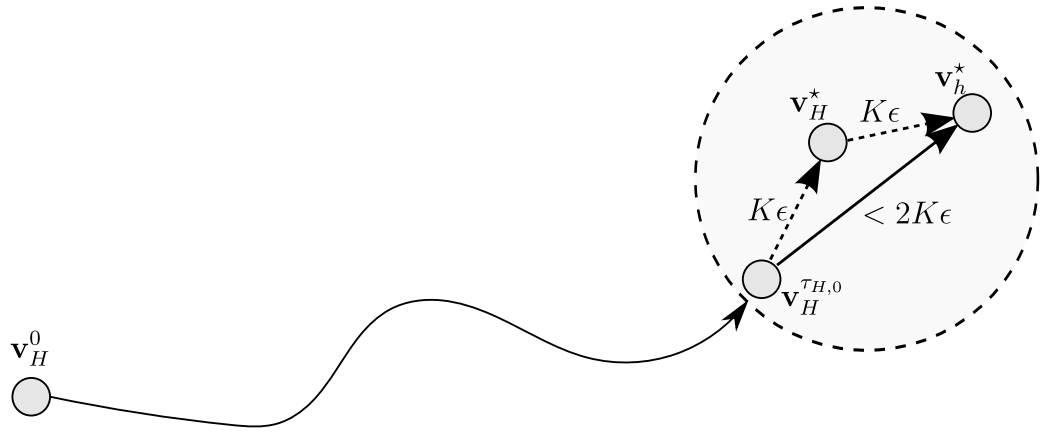


Figure 3.6: Stopping criteria in the coarse model

the fine iterations (bold line) is always faster than the alternative (dotted line) of computing the fine solution $\mathbf{v}_H^\star$ and then performing fine iterations to compute $\mathbf{v}_h^\star$. The preceding lemma just gives a rigorous backing to the intuitive idea that the exact solution of the approximate model does not add enough information to justify the cost of computing it.

The next step in the definition of the OWMS is the definition of $\tau_{h,0}$, i.e. the number of iterations that need to be performed in the fine scale. Of course this number must depend on a user specified error tolerance δ defined as follows,

$$\|\mathbf{v}_h^\star - \mathbf{v}_h^{\tau_{h,0}}\|_\infty < \delta.$$

Any solution that satisfies the solution above is called δ-optimal (note that under our assumptions $\mathbf{v}_h^\star$ is unique). The upper bound derived in the lemma below depends on δ and the amount of scale separation present in the problem ϵ.

Lemma 3.5 *Suppose that the initial point for the fine iterations is*

$$\mathbf{v}_h^0 \triangleq \mathbf{I}_H^h \mathbf{v}_H^{\tau'_{H,0}},$$

where $\tau'_{H,0}$ satisfies equation (3.31). Then the number of iterations in the fine model required

to compute a δ-optimal solution is bounded by

$$\tau_{h,0} \leq \max\left\{\frac{\log\left(\frac{K\epsilon(2-\alpha_H)}{\delta}\right)}{|\log\alpha_h|}, 0\right\}. \tag{3.36}$$

Proof Since the initial guess of the fine model is the solution in the coarse model, using Lemma 3.4, we have

$$\begin{aligned}
\|\mathbf{v}_h^0 - \mathbf{v}_h^\star\|_\infty &\leq \|\mathbf{v}_h^0 - \mathbf{I}_H^h\mathbf{v}_H^\star\|_\infty + \|\mathbf{I}_H^h\mathbf{v}_H^\star - \mathbf{v}_h^\star\|_\infty, \\
&\leq \alpha_H^{\tau'_{H,0}}\|\mathbf{v}_H^0 - \mathbf{v}_H^\star\|_\infty + K\epsilon, \\
&\leq \alpha_H^{\tau'_{H,0}}K + K\epsilon, \\
&= \epsilon(1-\alpha_H)K + K\epsilon, \\
&= K\epsilon(2-\alpha_H).
\end{aligned}$$

Hence, we select a parameter $\tau'_{h,0}$ such that

$$\|\mathbf{v}_h^{\tau'_{h,0}} - \mathbf{v}_h^\star\|_\infty \leq \alpha_h^{\tau'_{h,0}}\|\mathbf{v}_h^0 - \mathbf{v}_h^\star\|_\infty \leq \alpha_h^{\tau'_{h,0}}K\epsilon(2-\alpha_H) = \delta,$$

and so,

$$\tau'_{h,0} = \frac{\log\left(\frac{K\epsilon(2-\alpha_H)}{\delta}\right)}{|\log\alpha_h|},$$

as required. ∎

It follows from the lemma above that the number of iterations in the fine model decreases as ϵ decreases. This is because when ϵ decreases, the coarse model is a better approximation of the fine model and so we require less iterations in the fine model. This is in stark contrast to the classical single scale value iteration algorithm that requires more iterations as ϵ decreases. As $\epsilon \to 0$, the contraction modulus of value iteration α_h will tend to 1. In practice this means that if there is sufficient scale separation in the model, value iteration will be extremely slow. According to Lemma 3.1, the number of iterations required will tend to infinity as $\alpha_h \to 1$.

However, when $\epsilon \to 0$ then the prolongated value function $\mathbf{I}_H^h \mathbf{v}_H$ will equal tend to $\mathbf{v}_h$, and so only using the coarse model will be enough to get an accurate solution.

For OWMS and when $\epsilon \to 0$ it follows from Lemma 3.5 that the upper bound of the number of iterations needed in the fine model tends to zero, i.e. no iterations are needed in the fine model. This is very good news from the point of computation because as alluded to above when ϵ is small the algorithm may require arbitrarily many iterations using the full model to converge. At the regime of $\epsilon \to 0$, the coarse model can replace the fine model completely, and OWMS can detect that ϵ is small enough and not perform any expensive iterations using the fine model.

Note that ϵ is not an input parameter for our algorithm but represents the scale difference in the problem. On the other hand, δ is a user specified parameter and it represents the accuracy of the final solution. Therefore, the two parameters are independent of each other. For larger δ, the final solution could be less accurate, and so it requires less iterations in the fine model. This again can be seen from Lemma 3.5. While ϵ and δ are not directly related, they could have similar effects. For example, for a fixed ϵ a small δ (high accuracy) will mean that more iterations using the fine model will be performed by both value iteration and OWMS. Similarly, for a fixed δ a small ϵ would imply more iterations for the classical value iteration algorithm but the situation for OWMS is more complicated. For example, if $\epsilon \ll \delta$ then OWMS will make no iterations with the fine model. All these relationships can be derived from the expressions derived in Lemma 3.5, and depend on parameters that are known (up to a multiplicative constant) by the user.

Of course comparing just the number of iterations in the fine model is not sufficient. In order to perform a more rigorous and fair comparison between the newly proposed scheme OWMS and the classical single scale value iteration we derive the complexity of OWMS. We then find the conditions the MDP has to satisfy in order for the OWMS to have a more favorable complexity than value iteration.

Theorem 3.6 *For $\mathcal{MMDP}(\epsilon, n, m, L)$, the complexity of the OWMS is*

$$\mathcal{O}\left(\frac{\log\left(\frac{1}{\epsilon(1-\alpha_H)}\right)}{|\log\alpha_H|}m^2L^n + \max\left\{\frac{\log\left(\frac{\epsilon(2-\alpha_H)}{\delta}\right)}{|\log\alpha_h|}, 0\right\}(nm)^2L\right). \tag{3.37}$$

Proof The complexity of the algorithm is divided into two parts. The first part is the computational complexity associated with the coarse model. The second part is the complexity associated with fine iterations. Combining the information obtained by the Lemmas 3.4 and 3.5, we obtain the required result. ∎

We are now in a position to derive conditions that the MDP needs to satisfy so that we can guarantee that the proposed scheme will outperform value iteration.

Theorem 3.7 *Suppose that the tolerance $\delta < \min\{\epsilon(2-\alpha_H), 1\}$. For $\mathcal{MMDP}(\epsilon, n, m, L)$, the complexity of the OWMS is less than the complexity of the value iteration if*

$$n^2 \geq \frac{\log(\epsilon(1-\alpha_H))}{\log(\epsilon(2-\alpha_H)(1-\alpha_h))}\frac{|\log\alpha_h|}{|\log\alpha_H|}L^{n-1}. \tag{3.38}$$

Proof From equations (3.37) and (3.26), we know the complexity of both algorithms. Also, since $\delta < \min\{\epsilon(2-\alpha_H), 1\}$,

$$\log\left(\frac{\epsilon(2-\alpha_H)}{\delta}\right) > 0 \quad \text{and} \quad \log\left(\frac{1}{(1-\alpha_h)\delta}\right) > 0.$$

We proceed by computing the difference between the complexities.

$$\begin{aligned} & \frac{\log\left(\frac{1}{(1-\alpha_h)\delta}\right)}{|\log\alpha_h|}(nm)^2L - \frac{\log\left(\frac{1}{\epsilon(1-\alpha_H)}\right)}{|\log\alpha_H|}m^2L^n - \frac{\log\left(\frac{\epsilon(2-\alpha_H)}{\delta}\right)}{|\log\alpha_h|}(nm)^2L \\ = \; & \frac{m^2}{L}\left[\log\left(\frac{1}{\epsilon(1-\alpha_h)(2-\alpha_H)}\right)\frac{n^2}{|\log\alpha_h|} - \log\left(\frac{1}{\epsilon(1-\alpha_H)}\right)\frac{L^{n-1}}{|\log\alpha_H|}\right]. \end{aligned}$$

Using inequality (3.38) implies that the difference is greater than

$$\begin{aligned} &\geq \frac{m^2}{L}\left[\log\left(\frac{1}{\epsilon(1-\alpha_H)}\right)\frac{L^{n-1}}{|\log\alpha_H|} - \log\left(\frac{1}{\epsilon(1-\alpha_H)}\right)\frac{L^{n-1}}{|\log\alpha_H|}\right] \\ &\geq 0. \end{aligned}$$

∎

As we can see, the complexity of the OWMS is not always less than the single resolution algorithm. This is due to the fact that the number of actions for each coarse state is an exponential compared to the number of actions for each single state in the fine model. We will return to this issue in Section 3.6. If the problem has sufficient scale separation (which is the setting of this chapter) then we see that inequality (3.38) is asymptotically satisfied in ϵ because

$$\frac{\log(\epsilon(1-\alpha_H))}{\log(\epsilon(1-\alpha_h)(2-\alpha_H))}\frac{|\log\alpha_h|}{|\log\alpha_H|} \to 0 \quad \text{as} \quad \epsilon \to 0.$$

Therefore, for a small enough ϵ, the complexity of the OWMS is less than that of value iteration.

3.5.2 Convergence Analysis of AMS

In this section we turn our attention to the full AMS which includes the OWMS as a special case. We first prove some technical lemmas that will be used later on.

Lemma 3.8 $\mathbf{I}_h^H\mathbf{I}_H^h = \mathbf{I}_m$, *where* $\mathbf{I}_m$ *is the identity matrix in* $\mathbb{R}^{m\times m}$.

Proof By definition, $\mathbf{I}_H^h = \text{diag}(\underbrace{\mathbf{1}_{n\times1}, \mathbf{1}_{n\times1}, \ldots, \mathbf{1}_{n\times1}}_{\text{m copies}})$, so for any $\mathbf{v} = (\mathbf{v}_i) \in \mathbb{R}^m$,

$$\mathbf{I}_H^h\mathbf{v} = \text{diag}(\underbrace{\mathbf{1}_{n\times1}, \mathbf{1}_{n\times1}, \ldots, \mathbf{1}_{n\times1}}_{\text{m copies}})\mathbf{v} = \text{diag}(\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_m}),$$

where $\mathbf{v_i} = \mathbf{v}_i\mathbf{1}_{n\times1}$, for $i = 1, 2, \ldots, m$. The equality above follows from the fact that the vector

$\mathbf{v}$ is premultiplied with a block diagonal matrix. Using the definition of $\mathbf{I}_h^H$, we obtain

$$\mathbf{I}_h^H \mathbf{I}_H^h \mathbf{v} = \text{diag}(\boldsymbol{\varphi}_1^T, \boldsymbol{\varphi}_2^T, \ldots, \boldsymbol{\varphi}_m^T)\text{diag}(\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_m}).$$

Using the fact that the $\boldsymbol{\varphi}_i$ are the stationary distribution vectors,

$$\sum_{j=1}^{n} (\boldsymbol{\varphi}_i)_j = 1 \quad \forall i \in \{1, 2, \ldots, m\},$$

we obtain $\boldsymbol{\varphi}_i^T \mathbf{v_i} = \mathbf{v}_i$ and $\mathbf{I}_h^H \mathbf{I}_H^h \mathbf{v} = \mathbf{v}$, as claimed. ∎

Lemma 3.9 $\|\mathbf{I}_H^h\|_\infty = \|\mathbf{I}_h^H\|_\infty = 1$.

Proof $\mathbf{I}_H^h$ is a matrix with only one 1 in each of its rows, and so $\|\mathbf{I}_H^h\|_\infty = 1$. On the other hand, $\mathbf{I}_h^H$ is a stochastic matrix, and so $\|\mathbf{I}_h^H\|_\infty = 1$. ∎

Lemma 3.10 $\|\mathbf{I} - \mathbf{I}_H^h \mathbf{I}_h^H\|_\infty \leq 2$, *where* $\mathbf{I}$ *is the identity matrix in* $\mathbb{R}^{mn \times mn}$.

Proof Notice that $\mathbf{I}_h^H$ is a stochastic matrix, and

$$\mathbf{I}_H^h \mathbf{I}_h^H = \text{diag}(\underbrace{\mathbf{1}_{n\times 1}, \mathbf{1}_{n\times 1}, \ldots, \mathbf{1}_{n\times 1}}_{\text{m copies}})\text{diag}(\boldsymbol{\varphi}_1^T, \boldsymbol{\varphi}_2^T, \ldots, \boldsymbol{\varphi}_m^T) = \text{diag}(\hat{\boldsymbol{\varphi}}_1^T, \hat{\boldsymbol{\varphi}}_2^T, \ldots, \hat{\boldsymbol{\varphi}}_m^T),$$

where $\hat{\boldsymbol{\varphi}}_i^T \triangleq \mathbf{1}_{n\times 1} \boldsymbol{\varphi}_i^T$ for $i = 1, 2, \ldots, m$. In addition, $\mathbf{I}_H^h \mathbf{I}_h^H$ is also a stochastic matrix because each of $\hat{\boldsymbol{\varphi}}_i^T$ is a stochastic matrix. Therefore, the sum of the absolute values for every row of $\mathbf{I} - \mathbf{I}_H^h \mathbf{I}_h^H$ must be less than or equal to 2. ∎

Suppose we have a current solution $\mathbf{v}_h$, we then restrict it to the coarse model for correction, we then prolong the correction to the fine model, and add it to $\mathbf{v}_h$. We call this process a correction and the associated *correction operator* is defined as

$$\mathcal{T}_\tau \mathbf{v}_h \triangleq \mathbf{v}_h + s\mathbf{I}_H^h (T_H^\tau \mathbf{I}_h^H \mathbf{v}_h - \mathbf{I}_h^H \mathbf{v}_h). \tag{3.39}$$

If the correction $\mathbf{v}'_h = \mathcal{T}_\tau \mathbf{v}_h$ is useful for the problem, then the new solution $\mathbf{v}'_h$ should be closer to the optimal solution than the original $\mathbf{v}_h$. In the next lemma we provide a link between the corrected value function $\mathbf{v}'_h$ and the current incumbent $\mathbf{v}_h$.

Lemma 3.11 *For the current value $\mathbf{v}_h$ in the fine level, let $\mathbf{v}'_h = \mathcal{T}_\tau \mathbf{v}_h$, then*

$$\|\mathbf{v}^\star_h - \mathbf{v}'_h\|_\infty \leq [|1-s| + s\alpha^\tau_H] \, \|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty + (s\alpha^\tau_H + s)K\epsilon + 2Ks, \tag{3.40}$$

where $s \geq 0$ is the fixed stepsize.

Proof

$$\begin{aligned}
\|\mathbf{v}^\star_h - \mathbf{v}'_h\|_\infty &= \|\mathbf{v}^\star_h - \mathbf{v}_h - s\mathbf{I}^h_H(\mathbf{v}^\tau_H - \mathbf{I}^H_h \mathbf{v}_h)\|_\infty \\
&= \|\mathbf{v}^\star_h - \mathbf{v}_h - s\mathbf{I}^h_H \mathbf{v}^\tau_H + s\mathbf{I}^h_H \mathbf{I}^H_h \mathbf{v}_h\|_\infty \\
&\leq |1-s|\|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty + s\|\mathbf{v}^\star_h - \mathbf{I}^h_H \mathbf{v}^\tau_H\|_\infty + s\|\mathbf{v}_h - \mathbf{I}^h_H \mathbf{I}^H_h \mathbf{v}_h\|_\infty \quad \text{(triangle ineq.)} \\
&\leq |1-s|\|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty + s\|\mathbf{v}^\star_h - \mathbf{I}^h_H \mathbf{v}^\star_H\|_\infty \\
&\quad + s\|\mathbf{I}^h_H \mathbf{v}^\star_H - \mathbf{I}^h_H \mathbf{v}^\tau_H\|_\infty + s\|\mathbf{I} - \mathbf{I}^h_H \mathbf{I}^H_h\|_\infty \|\mathbf{v}_h\|_\infty \quad \text{(triangle ineq.)} \\
&\leq |1-s|\|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty + sK\epsilon + s\|\mathbf{I}^h_H\|_\infty \|\mathbf{v}^\star_H - \mathbf{v}^\tau_H\|_\infty + 2Ks \quad \text{(model assumptions)} \\
&\leq |1-s|\|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty + sK\epsilon + s\alpha^\tau_H \|\mathbf{v}^\star_H - \mathbf{I}^H_h \mathbf{v}_h\|_\infty + 2Ks \\
&\leq |1-s|\|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty + sK\epsilon + s\alpha^\tau_H \|\mathbf{v}^\star_H - \mathbf{I}^H_h \mathbf{v}^\star_h\|_\infty \\
&\quad + s\alpha^\tau_H \|\mathbf{I}^H_h \mathbf{v}^\star_h - \mathbf{I}^H_h \mathbf{v}_h\|_\infty + 2Ks \quad \text{(triangle ineq.)} \\
&\leq |1-s|\|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty + sK\epsilon + s\alpha^\tau_H \|\mathbf{I}^H_h\|_\infty \|\mathbf{I}^h_H \mathbf{v}^\star_H - \mathbf{v}^\star_h\|_\infty + s\alpha^\tau_H \|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty \\
&\quad + 2Ks \\
&\leq (|1-s| + s\alpha^\tau_H)\|\mathbf{v}^\star_h - \mathbf{v}_h\|_\infty + (s + s\alpha^\tau_H)K\epsilon + 2Ks. \quad \text{(model assumptions)}
\end{aligned}$$

∎

Lemma 3.11 provides a bound for the difference between the new solution $\mathbf{v}'_h$ and the optimal solution $\mathbf{v}^\star_h$. Using the preceding result we then derive the conditions required for the new error

to be smaller than the previous error. Therefore when these conditions are satisfied, Algorithm 3.1 is a contraction, and the convergence is guaranteed by the fixed point theorem.

Theorem 3.12 *Algorithm 3.1 is guaranteed to be a contraction if*

$$\begin{aligned}(1-\alpha_H^\tau)\|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty &\geq (\alpha_H^\tau+1)K\epsilon+2K \quad \textit{for} \quad 0\leq s\leq 1,\\ \left[\frac{2}{s}-(1+\alpha_H^\tau)\right]\|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty &\geq (\alpha_H^\tau+1)K\epsilon+2K \quad \textit{for} \quad s>1.\end{aligned}$$

Proof In the case $0\leq s\leq 1$, using Lemma 3.11, we obtain

$$\begin{aligned}\|\mathbf{v}_h^\star-\mathbf{v}_h'\|_\infty &\leq [|1-s|+s\alpha_H^\tau]\,\|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty+(s\alpha_H^\tau+s)K\epsilon+2Ks\\ &= \|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty+s\left([\alpha_H^\tau-1]\,\|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty\right)+s\left((\alpha_H^\tau+1)K\epsilon+2K\right)\\ &\leq \|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty-s\left((\alpha_H^\tau+1)K\epsilon+2K\right)+s\left((\alpha_H^\tau+1)K\epsilon+2K\right)\\ &\leq \|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty.\end{aligned}$$

For the case $1<s$, we obtain

$$\begin{aligned}\|\mathbf{v}_h^\star-\mathbf{v}_h'\|_\infty &\leq [|1-s|+s\alpha_H^\tau]\,\|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty+(s\alpha_H^\tau+s)K\epsilon+2Ks\\ &= \|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty+s\left(-\frac{2}{s}+(1+\alpha_H^\tau)\right)\|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty+s\left((\alpha_H^\tau+1)K\epsilon+2K\right)\\ &\leq \|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty-s\left((\alpha_H^\tau+1)K\epsilon+2K\right)+s\left((\alpha_H^\tau+1)K\epsilon+2K\right)\\ &\leq \|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty.\end{aligned}$$

∎

We note from the conditions of Theorem 3.12 that $(\alpha_H^\tau+1)K\epsilon+2K$ is a constant throughout all the iterations. However, $\|\mathbf{v}_h^\star-\mathbf{v}_h\|_\infty$ depends on the current solution $\mathbf{v}_h$. This indicates the correction is not guaranteed to be useful when the current solution is too close to the exact solution $\mathbf{v}_h^\star$. This suggests we should restrict the number of coarse iterations of the algorithm. We should not perform coarse iterations when the current solution is close to $\mathbf{v}_h^\star$. The result does not provide insight into how to select s since it depends on the optimal solution. In the

result below we provide the range that s can take so that the conditions of Theorem 3.12 are satisfied.

Corollary 3.13 *Algorithm 3.1 is a contraction only if the stepsize is chosen so that,*

$$0 \leq s \leq \frac{2}{1+\alpha_H^\tau}.$$

Proof Using the result from Theorem 3.12 we obtain,

$$(\alpha_H^\tau + 1)K\epsilon + 2K \geq 0.$$

Algorithm 3.1 is guaranteed to be a contraction only if

$$1 - \alpha_H^\tau \geq 0 \quad \text{for} \quad 0 \leq s \leq 1, \tag{3.41}$$

$$\frac{2}{s} - (1+\alpha_H^\tau) \geq 0 \quad \text{for} \quad s > 1. \tag{3.42}$$

In the case when $s \leq 1$, equation (3.41) is always satisfied, and in addition the following holds,

$$|1-s| + s\alpha_H^\tau = 1 - s + s\alpha_H^\tau = 1 - s(1-\alpha_H^\tau) \leq 1.$$

However, when $s > 1$, equation (3.42) is only satisfied when

$$s \leq \frac{2}{1+\alpha_H^\tau}.$$

For $1 < s \leq \dfrac{2}{1+\alpha_H^\tau}$,

$$|1-s| + s\alpha_H^\tau = -1 + s + s\alpha_H^\tau = -1 + s(1+\alpha_H^\tau) \leq 1.$$

∎

The above corollary shows that when $s > 2/(1+\alpha_H^\tau)$, it is not guaranteed that the correction would be useful. With the results obtained above concerning the correction iterations we are

now in a position to derive the complexity of the proposed scheme.

Let $\mathbf{v}_{a,h}^{2i}$ denote the solution after calculations in node $2i$, and let $\mathbf{v}_{a,H}^{2i-1}$ be the solution after calculations in node $2i-1$ for $i = 1, 2, \ldots, P+1$. Since the incumbent solution $\mathbf{v}_{a,H}^{2i-1}$ is in the coarse level, we prolongate the solution of the coarse model to the fine model as follows,

$$\mathbf{v}_a^j = \begin{cases} \mathbf{v}_{a,h}^j & \text{if } j \text{ is even,} \\ \mathbf{I}_H^h \mathbf{v}_{a,H}^j & \text{if } j \text{ is odd,} \end{cases} \quad \text{for } j = 1, 2, ..., 2(P+1).$$

With this notation the flow of computations for the AMS is,

$$\mathbf{v}_a^0 \Rightarrow \mathbf{v}_a^1 \Rightarrow \mathbf{v}_a^2 \Rightarrow \cdots \Rightarrow \mathbf{v}_a^{2P+1} \Rightarrow \mathbf{v}_a^{2(P+1)},$$

where $\mathbf{v}_a^{2(P+1)}$ is the final solution computed by the algorithm and therefore satisfies,

$$\|\mathbf{v}_h^\star - \mathbf{v}_a^{2(P+1)}\|_\infty \leq \delta,$$

where δ is a user specified error tolerance. We assume that for P-AMS, we always perform $\tau_{H,P}$ iterations in each coarse iteration node and $\tau_{h,P}$ iterations in first P fine iteration nodes for constants P, $\tau_{h,P}$, and $\tau_{H,P}$. In the complexity analysis below we fix these constants, and calculate the number of iterations the algorithm needs to refine the final solution from the AMS.

Lemma 3.14 *Suppose that in the P-AMS we perform $\tau_{H,P}$ iterations in node $1, 3, 5, \ldots, 2P+1$ and $\tau_{h,P}$ iterations in node $2, 4, 6, \ldots, 2P$, and with a constant stepsize s satisfying the conditions in Corollary 3.13. Then the initial point ($\mathbf{v}_a^{2P+1}$) from which the computations in the final node $2P+2$ start, satisfies*

$$\|\mathbf{v}_h^\star - \mathbf{v}_a^{2P+1}\|_\infty \leq \eta^P K\epsilon + \eta^P \alpha_H^{\tau_{H,P}} K + \frac{1-\eta^{P+1}}{1-\eta} sC, \tag{3.43}$$

where $\eta \triangleq (|1-s| + s\alpha_H^{\tau_{H,P}})\alpha_h^{\tau_{h,P}}$ and $C \triangleq (1+\alpha_H^{\tau_{H,P}})K\epsilon + 2K$.

Proof We will prove this using induction. Using the contraction mapping property,

$$\begin{aligned}\|\mathbf{v}_h^\star - \mathbf{v}_a^1\|_\infty \leq \|\mathbf{v}_h^\star - \mathbf{I}_H^h \mathbf{v}_H^\star\|_\infty + \|\mathbf{I}_H^h \mathbf{v}_H^\star - \mathbf{v}_a^1\|_\infty &\leq K\epsilon + \alpha_H^{\tau_{H,P}} \|\mathbf{v}_H^\star - \mathbf{v}_H^0\|_\infty, \\ &= K\epsilon + \alpha_H^{\tau_{H,P}} K\end{aligned}$$

and

$$\|\mathbf{v}_h^\star - \mathbf{v}_a^2\|_\infty \leq \alpha_h^{\tau_{h,P}} \|\mathbf{v}_h^\star - \mathbf{v}_a^1\|_\infty.$$

Using Lemma 3.11, we have

$$\|\mathbf{v}_h^\star - \mathbf{v}_a^3\|_\infty \leq \left[|1-s| + s\alpha_H^{\tau_{H,P}}\right] \|\mathbf{v}_h^\star - \mathbf{v}_a^2\|_\infty + (s\alpha_H^{\tau_{H,P}} + s)K\epsilon + 2Ks,$$

which gives the following estimate,

$$\begin{aligned}\|\mathbf{v}_h^\star - \mathbf{v}_a^3\|_\infty &\leq \left[|1-s| + s\alpha_H^{\tau_{H,P}}\right] \alpha_h^{\tau_{h,P}} \|\mathbf{v}_h^\star - \mathbf{v}_a^1\|_\infty + (s\alpha_H^{\tau_{H,P}} + s)K\epsilon + 2Ks \\ &= \eta \|\mathbf{v}_h^\star - \mathbf{v}_a^1\|_\infty + sC.\end{aligned}$$

By induction, after P pairs, we have

$$\begin{aligned}\|\mathbf{v}_h^\star - \mathbf{v}_a^{2P+1}\|_\infty \leq \eta^P \|\mathbf{v}_h^\star - \mathbf{v}_a^1\|_\infty + sC \sum_{k=0}^{P} \eta^k &= \eta^P \|\mathbf{v}_h^\star - \mathbf{v}_a^1\|_\infty + \left(\frac{1-\eta^{P+1}}{1-\eta}\right) sC \\ &\leq \eta^P K\epsilon + \eta^P \alpha_H^{\tau_{H,P}} K + \left(\frac{1-\eta^{P+1}}{1-\eta}\right) sC,\end{aligned}$$

as required. ∎

Lemma 3.14 establishes the error of the current solution before entering the last node. We now complete the analysis by deriving an upper bound for the number of iterations required in the fine model to get the final solution with tolerance δ.

Lemma 3.15 *Suppose that in the P-AMS we perform $\tau_{H,P}$ iterations in node $1, 3, 5, \ldots, 2P+1$ and $\tau_{h,P}$ iterations in node $2, 4, 6, \ldots, 2P$, and with a constant stepsize s satisfying the conditions*

in Corollary 3.13. Then the number of iterations in the last node τ_a is bounded by

$$\tau_a \leq \max\left\{\frac{\log\left(\frac{KZ}{\delta}\right)}{|\log \alpha_h|}, 0\right\}, \tag{3.44}$$

where

$$Z \triangleq \eta^P \epsilon + \eta^P \alpha_H^{\tau_{H,P}} + \frac{1-\eta^{P+1}}{1-\eta} s\left((1+\alpha_H^{\tau_{H,P}})\epsilon + 2\right), \tag{3.45}$$

$$\eta \triangleq (|1-s| + s\alpha_H^{\tau_{H,P}})\alpha_h^{\tau_{h,P}}. \tag{3.46}$$

Proof Using Lemma 3.14 implies

$$\begin{aligned}\|\mathbf{v}_h^\star - \mathbf{v}_a^{2P+1}\|_\infty &\leq \eta^P K\epsilon + \eta^P \alpha_H^{\tau_{H,P}} K + \left(\frac{1-\eta^{P+1}}{1-\eta}\right) sC \\ &\leq K \underbrace{\left[\eta^P \epsilon + \eta^P \alpha_H^{\tau_{H,P}} + \frac{1-\eta^{P+1}}{1-\eta} s\left((1+\alpha_H^{\tau_{H,P}})\epsilon + 2\right)\right]}_{=:Z}.\end{aligned}$$

Now let $\mathbf{v}_h^0 \triangleq \mathbf{v}_a^{2P+1}$ and select τ_a' such that

$$\|\mathbf{v}_h^{\tau_a'} - \mathbf{v}_h^\star\|_\infty \leq \alpha_h^{\tau_a'} \|\mathbf{v}_h^0 - \mathbf{v}_h^\star\|_\infty \leq \alpha_h^{\tau_a'} KZ = \delta,$$

for a fixed tolerance δ. Using the same analysis as in Lemma 3.5, we obtain

$$\tau_a' = \frac{\log\left(\frac{KZ}{\delta}\right)}{|\log \alpha_h|}. \tag{3.47}$$

and therefore $\tau_a \leq \max\{\tau_a', 0\}$ as required. ∎

Theorem 3.16 *The $\mathcal{MMDP}(\epsilon, n, m, L)$, a P-AMS with $\tau_{H,P}$ iterations in node $1, 3, 5, \ldots, 2P+1$ and $\tau_{h,P}$ iterations in node $2, 4, 6, \ldots, 2P$ and a constant stepsize s satisfying the conditions*

in Corollary 3.13 has the following worst-case computational complexity,

$$\mathcal{O}\left(\left[\max\left\{\frac{\log\left(\frac{Z}{\delta}\right)}{|\log\alpha_h|},0\right\}+P\tau_{h,P}\right](mn)^2L+(P+1)\tau_{H,P}m^2L^n\right), \tag{3.48}$$

where Z is defined in (3.45)*.*

Proof Using Lemma 3.15, we obtain the total number of iterations in both the fine and coarse models. Therefore, the complexity is the sum of the number of iterations multiplied by the cost of each iteration for both fine and coarse model. ∎

The theorem below shows that OWMS is one specific case of AMS if the parameters of AMS are judiciously chosen.

Theorem 3.17 *Suppose the tolerance $\delta < \min\{Z, \epsilon(2-\alpha_H)\}$, where Z is defined in* (3.45)*. For $\mathcal{MMDP}(\epsilon, n, m, L)$, AMS has the same complexity as OWMS in the case when $P = 0$, $s = 0$, and $\tau_{H,P} = \tau'_{H,0}$, where $\tau'_{H,0}$ is defined in equation (3.35).*

Proof Using Theorem 3.16 with $P = 0$, $s = 0$, and $\tau_{H,P} = \tau'_{H,0}$, where $\tau'_{H,0}$ is defined in equation (3.35), we obtain

$$\begin{aligned}
&\left[\frac{\log\left(\frac{Z}{\delta}\right)}{|\log\alpha_h|}+P\tau_{h,P}\right](mn)^2L+(P+1)\tau_{H,P}m^2L^n\\
&=\frac{\log\left(\frac{Z}{\delta}\right)}{|\log\alpha_h|}(mn)^2L+\tau_{H,P}m^2L^n\\
&=\frac{\log\left(\frac{Z}{\delta}\right)}{|\log\alpha_h|}(mn)^2L+\frac{\log\left(\frac{1}{\epsilon(1-\alpha_H)}\right)}{|\log\alpha_H|}m^2L^n,
\end{aligned}$$

where

$$\begin{aligned} Z &= \eta^P \epsilon + \eta^P \alpha_H^{\tau_{H,P}} + \frac{1-\eta^{P+1}}{1-\eta} s\left((1+\alpha_H^{\tau_{H,P}})\epsilon + 2\right) \\ &= \epsilon + \alpha^{\tau'_{H,0}} \\ &= \epsilon + \epsilon(1-\alpha_H) \qquad \text{(Equation (3.34) implies } \alpha_H^{\tau'_{H,0}} = \epsilon(1-\alpha_H)) \\ &= \epsilon(2-\alpha_H). \end{aligned}$$

Therefore, the complexity of AMS becomes

$$\mathcal{O}\left(\frac{\log\left(\frac{\epsilon(2-\alpha_H)}{\delta}\right)}{|\log \alpha_h|}(mn)^2 L + \frac{\log\left(\frac{1}{\epsilon(1-\alpha_H)}\right)}{|\log \alpha_H|} m^2 L^n \right),$$

which is same as the OWMS. ∎

Using the preceding theorem we can conclude that if we define the parameters

$$(s^\star, \tau_{h,P}^\star, \tau_{H,P}^\star, P^\star) \in \arg \min_{s,\tau_{h,P},\tau_{H,P},P} \left[\max\left\{ \frac{\log\left(\frac{Z}{\delta}\right)}{|\log \alpha_h|}, 0 \right\} + P\tau_{h,P} \right] (mn)^2 L + (P+1)\tau_{H,P} m^2 L^n,$$

then the complexity of AMS with parameters $(s^\star, \tau_{h,P}^\star, \tau_{H,P}^\star, P^\star)$ must be less than or equal to the complexity of OWMS. Notice that the parameters $(s^\star, \tau_{h,P}^\star, \tau_{H,P}^\star, P^\star)$ are optimizing the worst case complexity. While the result above is useful it is difficult to obtain a closed form solution for these parameters. Instead in our numerical results we will use a suboptimal solution that is motivated by the optimal parameter selection problem above.

3.6 Action Space Sampling for the Coarse Model

It follows from the complexity analysis of the preceding sections that the advantages of the coarse model are (a) the dimensionality reduction in the state space and (b) the improved convergence rate. However, the action space in the coarse model is exponentially larger than the fine model. If the fine model $\mathcal{MMDP}(\epsilon, n, m, L)$ is aggregated using perturbation theory then the coarse model is an $\mathcal{MDP}(m, L^n)$ problem. We will take advantage of the well-established links between linear programming and MDPs together with constraints sampling techniques from [CC05, Cal10] to address the computational cost associated with the coarse model.

3.6.1 Linear Programming and MDPs

It is well known that the HJB equation (3.10) can be solved by the following LP,

$$\begin{aligned} \max \quad & \sum_{i \in \mathcal{X}^H} \mathbf{v}_H(i) \\ \text{s.t} \quad & \mathbf{v}_H(i) \leq \frac{G^H(i, a_H)}{|q_{ii}^H(a_H)| + \rho} + \sum_{j \neq i} \frac{q_{ij}^H(a_H)}{|q_{ii}^H(a_H)| + \rho} \mathbf{v}_H(j) \quad \forall a_H \in \mathcal{A}_i^H, i \in \mathcal{X}^H \end{aligned}$$

The LP formulation of $\mathcal{MDP}(m, L^n)$ has m variables and mL^n constraints. For large-scale problems, it is very likely that $m \ll mL^n$, and so we will have a lot more constraints than variables. In this commonly encountered scenario, most of the constraints are not active at the optimum, and eliminating them could reduce the computational cost of the problem [dFVR04]. We will use the constraint sampling technique to reduce the action space and so the complexity. We will make the following assumption regarding the relationship between samples and states.

Assumption 3.18 *Let $\mathcal{SSET} \triangleq \{(a_1, x_1), (a_2, x_2), (a_3, x_3), \ldots, (a_R, x_R)\}$ be the set of R samples from the probability mass function $\psi(a_H, x_H)$, and let $\mathcal{S}_i^A \triangleq \{(a_H, i) : a_H \in \mathcal{A}_i^H\}$. Then, $\mathcal{SSET} \cap \mathcal{S}_i^A \neq \emptyset$, $\forall i \in \mathcal{X}^H$.*

Assumption 3.18 implies that the set of samples will contain at least one state-action pair for each state. We use this assumption to ensure that our samples are enough to formulate another

MDP which has action sets that are the (non-empty) subset of the action sets in the original problem.

Lemma 3.19 *Let* $\tilde{\mathcal{A}}_i^H \triangleq \{a_H : (a_H, i) \in \mathcal{SSET} \cap \mathcal{S}_i^A\}$*, then the optimal solution of the following LP,*

$$\begin{aligned} \max \quad & \sum_{i \in \mathcal{X}^H} \tilde{\mathbf{v}}_H(i), \\ s.t. \quad & \tilde{\mathbf{v}}_H(i) \leq \frac{G^H(i, a_H)}{|q_{ii}^H(a_H)| + \rho} + \sum_{j \neq i} \frac{q_{ij}^H(a_H)}{|q_{ii}^H(a_H)| + \rho} \tilde{\mathbf{v}}_H(j) \quad \forall (a_H, i) \in \mathcal{SSET}. \end{aligned} \tag{3.49}$$

is the value function of the MDP below,

$$\begin{aligned} \min \qquad & \tilde{J}^H(i, \mathbf{u}_H) = \mathbb{E}\left[\int_0^\infty e^{-\rho t} G^H(x_H(t), \mathbf{u}_H(x(t)))\ dt\right], \\ s.t \qquad & x_H \sim \mathbf{Q}_H(\mathbf{u}_H(x_H(t)))\ ,\ t \geq 0, \\ & x_H(0) = i \quad , \quad \mathbf{u}_H \in \tilde{\mathcal{U}}^H, \\ & \tilde{\mathbf{v}}_H(i) = \min_{\mathbf{u}_H \in \tilde{\mathcal{U}}^H} \tilde{J}^H(i, \mathbf{u}_H), \end{aligned} \tag{3.50}$$

where $\tilde{\mathcal{U}}^H$ *is the policy space for* $\tilde{\mathcal{A}}_i^H$*,* $\forall i \in \mathcal{X}^H$*.*

Proof The LP in (3.49) is just the reformulation of (3.50) as a linear program. ∎

We refer to the MDP in (3.50) as the **reduced coarse model**. Note that in the reduced coarse model we still maintain the fast convergence rate due to the elimination of the multiscale structure and at the same time we are able to control the complexity per iteration by decreasing the number of actions. Of course when the reduced order policy space contains the optimal policy then indeed the solution of the reduced coarse model coincides with the coarse solution. This simple observation is established below.

Lemma 3.20 *The solution of the reduced coarse model is same as the solution of the coarse model when,*

$$\mathbf{u}_H^\star(i) \in \tilde{\mathcal{A}}_i^H \quad , \quad \forall i \in \mathcal{X}^H. \tag{3.51}$$

In other words, the policy space of the reduced coarse model contains the optimal policy.

Proof Since the policy space in the reduced coarse model is a subset of the policy space in the coarse model, if $\mathbf{u}_H^\star$ minimizes the expected discounted cost in the coarse model, it also minimizes the expected discounted cost in the reduced coarse model with the same value function. ∎

Of course it is unreasonable to make such a strong assumption as the one above. Instead we will analyze the performance of OWMS and AMS by sampling the actions in the coarse model uniformly. In practice, it is often the case that some action-state pairs are more important than others. As a result the uniform distribution assumption may not be the best from a computational perspective. However, if no additional assumption is made about the MDP then this is a valid assumption to examine. We use basic combinatorics to obtain a quantitative estimate of the probability of obtaining the optimal policy from the reduced coarse model.

Theorem 3.21 *For $\mathcal{MMDP}(\epsilon, n, m, L)$ suppose that a P-AMS with $\tau_{H,P}$ iterations in node $1, 3, 5, \ldots, 2P+1$ and $\tau_{h,P}$ iterations in node $2, 4, 6, \ldots, 2P$ and stepsize s is used. If $R = (1-\sigma)^{1/m} L^n$ samples are drawn from $\mathcal{A}_i^H$, $\forall i \in \mathcal{X}^H$ then the optimal solution will be obtained with probability $1-\sigma$ and the complexity of AMS is,*

$$\mathcal{O}\left(\left[\max\left\{\frac{\log\left(\frac{Z}{\delta}\right)}{|\log \alpha_h|}, 0\right\} + P\tau_{h,P}\right](mn)^2 L + (P+1)\tau_{H,P} m^2 R\right), \tag{3.52}$$

where Z is defined in Lemma 3.15.

Proof For state i in the coarse model, only one optimal action is needed to construct the optimal policy. For R actions that are drawn from $\mathcal{A}_i^H$, the total number of possible combinations is $\binom{L^n}{R}$. If the optimal action is obtained in the samples, the total number of possible

combinations is $\binom{L^n-1}{R-1}$. So, the probability of obtaining the optimal action is

$$\frac{\binom{L^n-1}{R-1}}{\binom{L^n}{R}} = \frac{R}{L^n}.$$

The optimal policy is obtained only if each action space in the reduced coarse model contains its optimal action. The probability of obtaining the optimal policy is then $1-\sigma \triangleq \left(\frac{R}{L^n}\right)^m$, and so $R = (1-\sigma)^{1/m}L^n$. If $\tilde{\mathbf{v}}_H^\star = \mathbf{v}_H^\star$, the convergence of the reduced coarse model is same as coarse model with less actions in each state. Therefore, with $R = (1-\sigma)^{1/m}L^n$, the AMS has probability $1-\sigma$ to obtain $\mathbf{v}_h^\star$ with complexity

$$\mathcal{O}\left(\left[\max\left\{\frac{\log\left(\frac{Z}{\delta}\right)}{|\log\alpha_h|},0\right\}+P\tau_{h,P}\right](mn)^2L+(P+1)\tau_{H,P}m^2R\right),$$

as stated in Theorem 3.16. ∎

OWMS is a specific case of AMS and so Theorem 3.21 covers the case of OWMS too. While the result above guarantees $\tilde{\mathbf{v}}_H^\star = \mathbf{v}_H^\star$ with certain probability, it may still require a large number of samples. To address this last point we make use of the result from Calafiore [Cal10] to estimate the number of action-state pairs that would guarantee one extra action-state pair would not change the value function of the reduced coarse model with certain probability. This is useful because it provides a guide on the number of sufficient samples required to obtain the most "useful" actions so that the value function $\tilde{\mathbf{v}}_H^\star$ is likely to be close to $\mathbf{v}_H^\star$.

Theorem 3.22 *Consider the reduced coarse model with state-action pairs set $\mathcal{SSET}$ with size mR (R actions in each state on average) and let Assumption 3.18 hold. Let $(\tilde{a}_H, \tilde{x}_H)$ be a state-action pair drawn from $\psi(a_H, x_H)$. Let $\tilde{\mathbf{v}}_H^\star$ be the value function of the reduced coarse model*

with $\mathcal{SSET}$, and $\hat{\mathbf{v}}_H^\star$ be the value function of the reduced coarse model with $\mathcal{SSET} \cup (\tilde{a}_H, \tilde{x}_H)$. Let

$$\mathcal{V}(\mathcal{SSET}) \triangleq \mathbb{P}((\tilde{a}_H, \tilde{x}_H) \in \psi(a_H, x_H) : \tilde{\mathbf{v}}_H^\star = \hat{\mathbf{v}}_H^\star), \tag{3.53}$$

which is the probability that adding an extra action-state pair drawn from $\psi(a_H, x_H)$ does not change the optimal policy in the reduced coarse model. Then,

$$\mathbb{P}(\mathcal{SSET} \in \psi(a_H, x_H)^{mR} : \mathcal{V}(\mathcal{SSET}) \geq 1 - \ell) \geq 1 - \beta, \tag{3.54}$$

if

$$mR \geq \frac{2}{\ell} \log \beta^{-1} + \frac{4}{\ell}(m-1) \tag{3.55}$$

for $\ell, \beta \in (0, 1)$.

Proof The proof is provided in [Cal10]. ∎

Note that the lower bound of the number of state-action pairs does not depend on the number of state-action pairs in the original coarse model. In practice, it is possible that one would have an idea what actions are more likely to be optimal in each state. Such knowledge can be used to construct a better constraint sampling density $\psi(a_H, x_H)$.

3.7 Numerical Experiments

In this section, we illustrate the performance of the AMS and the OWMS using two numerical examples. The first example is the widely used example from the field of manufacturing we introduced in Section 3.4.3. The second example is motivated from stochastic molecular dynamics. Applications in molecular dynamics (MD) have a strong multiscale structure and applications related to MD and stochastic optimal control are beginning to emerge [SWH12]. In the results below we compare the performance of the proposed algorithms and the conventional value iteration algorithm. The numerical performance of FAS has already been discussed. For reasons explained in Section 3.4.4, FAS is not competitive with any of the other methods.

The AMS requires the specification of 4 parameters. In the numerical examples below we fix $\tau_{h,P} = \tau_{H,P} = 100$. The stepsize s is selected according to the estimates in Corollary 3.13. The parameter P is determined adaptively. To monitor the progress of the algorithm we define the following measure,

$$\Phi(j) \triangleq \|A_h \mathbf{v}_a^{j-2} - A_h \mathbf{v}_a^j\|_\infty, \quad \forall j \in \{2 + 2i : i \in \mathbb{Z}^+\},$$

which can be interpreted as the improvement of the solution in node $j-1$ and j. It follows from Theorem 3.12 that the coarse correction is only guaranteed to be a contraction if the current solution is far away from the true solution. With that in mind, we define

$$\Psi(j) \triangleq \frac{\Phi(j-2) - \Phi(j)}{\Phi(j-2)}, \quad \forall j \in \{2 + 2i : i \in \mathbb{Z}^+\},$$

to be the percentage change compared to the last two nodes. This measure of change was used to detect whether the coarse correction is still useful during calculations, and we stop using the coarse correction after node j when $\Psi(j)$ is less than some constant ϱ. In the experiments below we set $\varrho = 0.1$, and we used $\epsilon = 10^{-2}$. With this parameter choice for ϵ, the two problems contain some multiscale structure but it is not strong enough to just use the coarse model.

3.7.1 Manufacturing Example

Recall that in Section 3.4.3 we introduced an example motivated by a multiscale manufacturing process. We showed that the Full Approximation Scheme (FSA) fails in this example and this was one of our motivations for developing the proposed scheme. Figure 3.7 shows the performance of the different schemes for this example. The parameter settings are exactly the same as in Section 3.4.3. In Figure 3.7 (left) we show the iteration history, in terms of the distance to the exact solution, for each of the algorithms. The exact solution was obtained using linear programming. We do not plot the performance of the FAS for this model because it makes it difficult to see the differences between value iteration and our algorithms. The iteration history of FAS was plotted against value iteration in Figure 3.2. In Figure 3.7 (right)

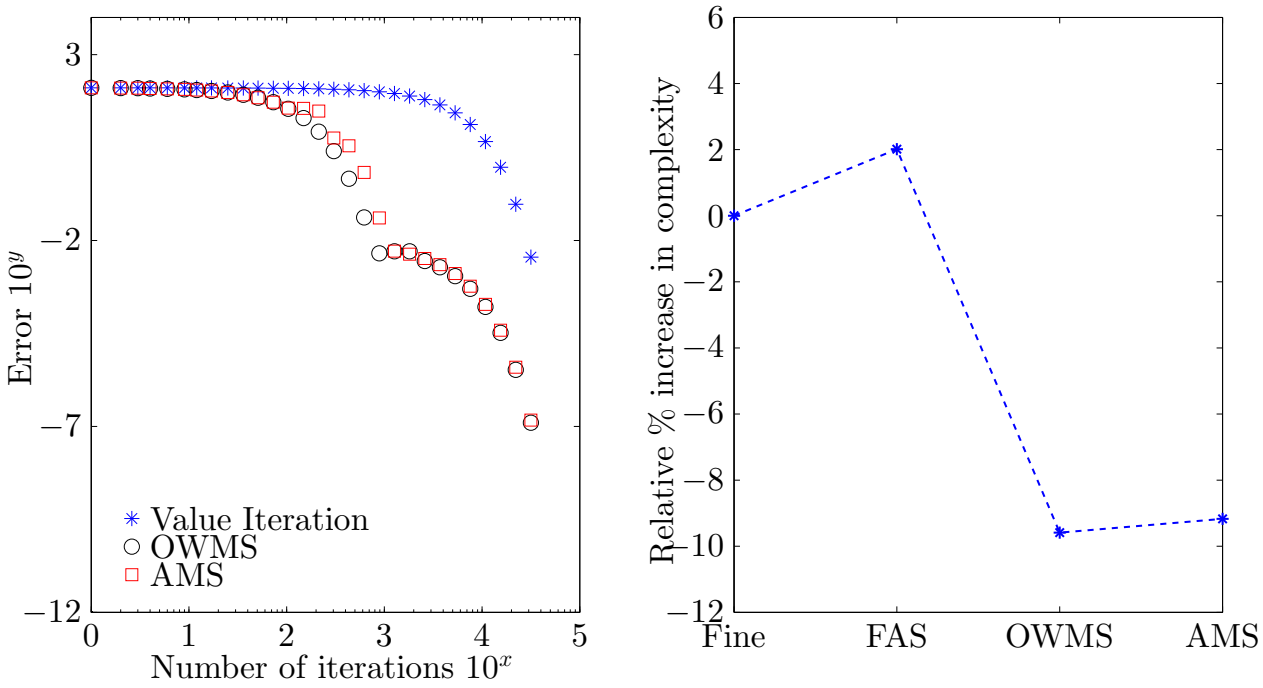


Figure 3.7: Numerical performance of the different algorithms. Parameters: $\mathbf{v}_h^0 = 0$, $\mathbf{v}_H^0 = 0$, $s = 1.15$, and $\rho = 0.05$. (Left) Iteration History. (Right) Relative increase in realized complexity of the different algorithms. Value iteration was taken to be the base line. Compared to value iteration conventional FAS has an increased complexity, whereas the proposed schemes achieve a 10% reduction.

we plot the (relative) comparisons of realized complexity of the different algorithms as the number of iterations multiplied by the cost per iteration. We show the realized complexity of each algorithm in relation to the realized complexity of value iteration. Since the size of the problem is not large we do not apply sampling in this example. Both of our proposed multiresolution algorithms are better than solely using the fine model. In this toy example, the total complexity can be reduced by 10% without any penalty on the accuracy. Notice that in this example, we have $\alpha_H = 0.9967$. Therefore, the choice of our stepsize $s = 1.15$ is reasonable because $s < 2/(1 + \alpha_H^{\tau_{H,P}})$, with $\tau_{H,P} = 100$, and $s = 1.15$.

3.7.2 Example from Molecular Dynamics

In this section we use the proposed scheme to solve a larger problem motivated by molecular dynamics. The problem of controlling molecular dynamics is an active research area with many applications in material science and chemical engineering [SWH12]. The potential energy of

molecules is usually modeled as a stochastic differential equation, or as a Markov chain in the discrete case. Even though the transitions from one energy level to another energy level are considered to be stochastic, the underlying randomness is structured. Molecules are stable when they are at a local minimum of the potential energy and are very likely to make fast changes around the neighborhood of the local minimum. It is rare that molecules would move from one stable configuration to another. The event that a molecule jumps from one well to another is characterized as a rare event.

In this example, we consider a Markov chain with 50 states, where $\hat{\mathbf{Q}}$ is a block diagonal matrix with 10 blocks, and each block is a 5×5 matrix. The state space is $\mathcal{X}^h = \{1, 2, \ldots, 50\}$. In this particular example, each block represents the transitions between different configurations within a stable configuration. In practice this Markov chain is obtained by discretizing a stochastic differential equation. For this reason, we assume that $\hat{q}_{ij} = 0$ if $|i - j| > 1$. The matrix $\mathbf{W}$ represents the connection from one stable configuration to another. We assume $w_{ij} = 0$ if (i, j) is not from the set $\{(a, b) : (a, b) \in \{5, 6, 10, 11, 15, 16, 20, 21, 25, 26\}^2, |b - a| \leq 1\}$. Since the system is large, we simply sample the entries for $\hat{\mathbf{Q}}$ and $\mathbf{W}$ uniformly from the set $\{1, 2, 3, \ldots, 9\}$. In order to introduce a control element into the model, we assume that there exists a catalysis that can be used to speed up or slow down the rate by which the system moves between states. To be precise, we assume the following form of the generator,

$$\mathbf{Q}_h^\epsilon(a) = 3^a \left[\frac{1}{\epsilon} \hat{\mathbf{Q}} + \mathbf{W} \right],$$

where $a \in \{-1, -2/3, -1/3, 0, 1/3, 2/3, 1\}$. The objective function is $G(x, a) = x + 50|a|$, and as before we solve the infinite horizon model. Optimizing the system with this particular choice of cost function aims to control the dynamics so that the system remains in or close to state 1 without using too much catalysis. Figure 3.8 shows the numerical results of this example. For this example FAS performed particularly bad. In order to have a clearer comparison between the proposed algorithms and value iteration we do not plot the iteration history of the FAS. Notice that the OWMS scheme takes more time to converge than value iteration. After a few hundred iterations in the coarse model, the coarse model becomes ineffective because the

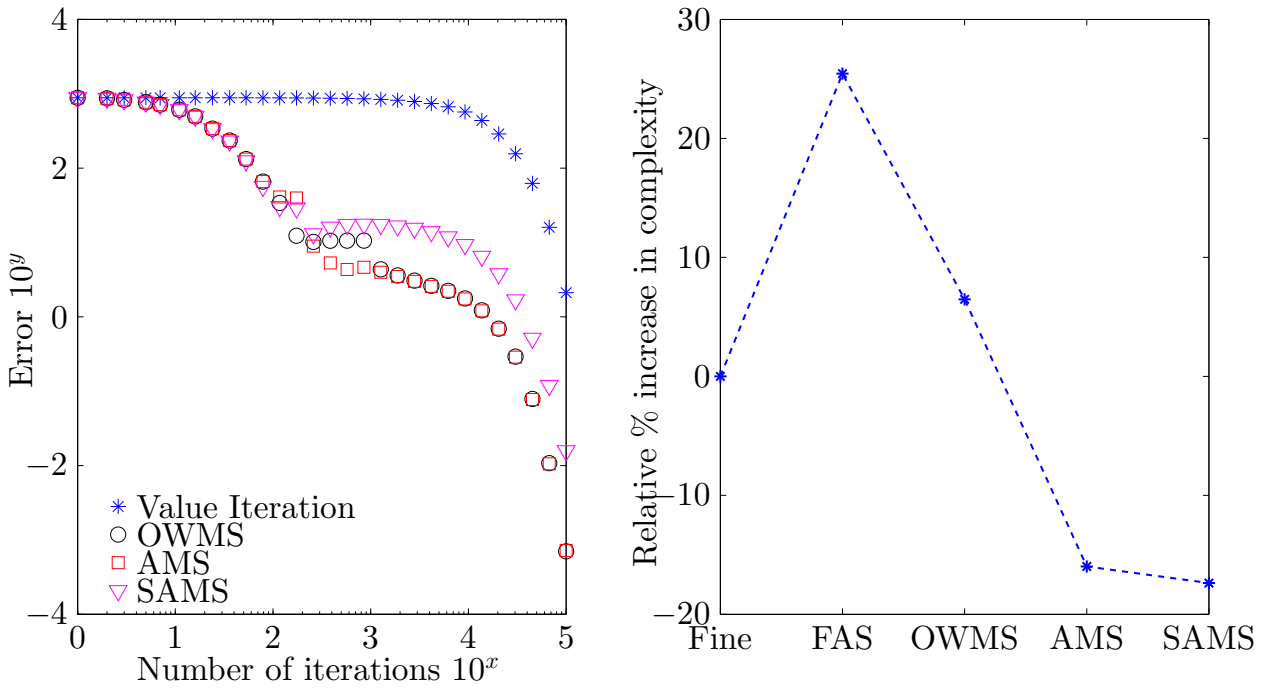


Figure 3.8: Numerical results of different algorithms. Parameters: $\mathbf{v}_h^0 = 0$, $\mathbf{v}_H^0 = 0$, $s = 1.1$, $R = 10000$, and $\rho = 0.05$. (Left) Convergence in each iteration. (Right) Relative increase in realized complexity of the different algorithms. Value iteration was taken to be the base line.

current solution is "too close" to the exact solution. Therefore, in this example, OWMS spends a lot of expensive iterations that do not achieve a significant error reduction. However AMS still outperforms value iteration. In the fine model, each state has 7 available actions, and so in the coarse model, each state has $7^5 = 16807$ actions. As the action space is so large, it is reasonable to apply the action sampling technique described in Section 3.6. In Figure 3.8 we plot the results when using AMS with action sampling (SAMS), and $R = 10000$ samples. From Figure 3.8 (left) we see that applying the sampling technique leads to a slower convergence rate of our scheme compared to the original AMS and OWMS. However, since the size of the coarse model is reduced the time spent per iteration is less. This point is made in Figure 3.8 (right) where the proposed scheme with action sampling computes the solution with less realized complexity compared to all the others. The performance is not much better than the original AMS due to the fact that uniform sampling is not very effective. In practice, we would expect that it is possible to find a good distribution $\psi(a_H, x_H)$ via empirical analysis. Still even without optimizing the proposed schemes we are able to achieve close to a 20% improvement over value iteration.

3.8 Discussion

We proposed the **A**lternating **M**ultiresolution **S**cheme (AMS), for Markov Decision Processes with a multiscale structure. Our scheme is an alternative framework and, under certain conditions, is theoretically superior to the conventional numerical methods used for this class of problems. The main idea of AMS is to use the coarse (aggregate) model as much as possible and avoid using the expensive full (fine) model for all the iterations. It was already known that the coarse model has less states than the fine model. But more importantly we showed that the coarse model is also better conditioned. Using complexity analysis and numerical examples we showed that the proposed scheme outperforms value iteration and the conventional multigrid based method (FAS). We also proposed a sampling method to address the problem of large action space in the coarse model.

Our proposed scheme exploits the multiscale structure of the problem, but does not depend on it, i.e. the convergence does not depend on having scale separation. When there is no scale separation the algorithm still computes the correct solution but there are no real benefits in terms of reduction of computation times. When the multiscale structure is very sharp, most of the calculations will be done in the coarse model, and the final solution is computed by sightly correcting the approximate solution using the fine model.

We believe that the general scheme proposed in this chapter can be extended to more general settings or even to different classes of problems. For example, one could replace value iteration by policy iteration. As long as the underlying algorithm is a contraction, the theoretical results of this chapter can be used to evaluate its performance.

Chapter 4

Empirical Risk Minimization: Probabilistic Complexity and Stepsize Strategy

There are two possible outcomes: if the result confirms the hypothesis, then you've made a measurement. If the result is contrary to the hypothesis, then you've made a discovery.

Enrico Fermi

Empirical risk minimization (ERM) is recognized as a special form in standard convex optimization. When using a first order method, the Lipschitz constant of the empirical risk plays a crucial role in the convergence analysis and stepsize strategies for these problems. We derive the probabilistic bounds for such Lipschitz constants using random matrix theory. We show that, on average, the Lipschitz constant is bounded by the ratio of the dimension of the problem to the amount of training data. We use our results to develop a new stepsize strategy for first order methods. The proposed algorithm, Probabilistic Upper-bound Guided stepsize strategy (PUG), outperforms the regular stepsize strategies with strong theoretical guarantee

on its performance.

4.1 Introduction

Empirical risk minimization (ERM) is one of the most powerful tools in applied statistics, and is regarded as the canonical approach to regression analysis. In the context of machine learning and big data analytics, various important problems such as support vector machines, (regularized) linear regression, and logistics regression can be cast as ERM problems, see for e.g. [SSBD14]. In an ERM problem, a training set with m instances, $\{(\mathbf{a}_i, b_i)\}_{i=1}^m$, is given, where $\mathbf{a}_i \in \mathbb{R}^n$ is an input and $b_i \in \mathbb{R}$ is the corresponding output, for $i = 1, 2, \ldots, m$. The ERM problem is then defined as the following convex optimization problem,

$$\min_{\mathbf{x}\in\mathbb{R}^n} \left\{ F(\mathbf{x}) \triangleq \frac{1}{m} \sum_{i=1}^{m} \phi_i(\mathbf{a}_i^T \mathbf{x}) + g(\mathbf{x}) \right\}, \tag{4.1}$$

where each loss function ϕ_i is convex with a Lipschitz continuous gradient, and the regularizer $g : \mathbb{R}^n \to \mathbb{R}$ is a continuous convex function which is possibly nonsmooth. Two common loss functions are

- Quadratic loss function: $\phi_i(x) = \frac{1}{2}(x - b_i)^2$.
- Logistic loss function: $\phi_i(x) = \log(1 + \exp(-xb_i))$.

One important example of g is the scaled 1-norm $\omega\|\mathbf{x}\|_1$ with a scaling factor $\omega \in \mathbb{R}^+$. This particular case is known as ℓ_1 regularization, and it has various applications in statistics [BCW14], machine learning [SZ13], signal processing [Don06], etc. The regularizer g acts as an extra penalty function to regularize the solution of (4.1). ℓ_1 regularization encourages sparse solutions, i.e. it favors solutions $\mathbf{x}$ with few non-zero elements. This phenomenon can be explained by the fact that the ℓ_1 norm is the tightest convex relaxation of the ℓ_0 norm, i.e. the cardinality of the non-zero elements of $\mathbf{x}$ [CRT06].

In general, if the regularizer g is nonsmooth, subgradient methods are used to solve (4.1). However, subgradient methods are not advisable if g is simple enough, and one can achieve higher efficiency by generalizing existing algorithms for unconstrained differentiable convex programs. Much research has been undertaken to efficiently solve ERM problems with simple g's. Instead of assuming the objective function is smooth and continuously differentiable, they aim to solve problems of the following form

$$\min_{\mathbf{x}\in\mathbb{R}^n}\{F(\mathbf{x}) \triangleq f(\mathbf{x}) + g(\mathbf{x})\}, \tag{4.2}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function with L-Lipschitz continuous gradient, and $g : \mathbb{R}^n \to \mathbb{R}$ is a continuous convex function which is nonsmooth but simple. By simple we mean that a proximal projection step can be performed either in closed form or is at least computationally inexpensive. Norms, and the ℓ_1 norm in particular, satisfy this property. A function f is said to have a L-Lipschitz continuous gradient if

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n. \tag{4.3}$$

For the purpose of this chapter, we denote the matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ to be a dataset such that the i^{th} row of $\mathbf{A}$ is $\mathbf{a}_i^T$, and so in the case of ERM problems,

$$f(\mathbf{x}) = \frac{1}{m}\sum_{i=1}^{m} \phi_i(\mathbf{a}_i^T\mathbf{x}) = \frac{1}{m}\sum_{i=1}^{m} \phi_i(\mathbf{e}_i^T\mathbf{A}\mathbf{x}), \tag{4.4}$$

where $\mathbf{e}_i \in \mathbb{R}^m$ has 1 as its i^{th} component and 0's elsewhere. f is called the empirical risk in ERM. We assume that each ϕ_i has a γ_i-Lipschitz continuous gradient and

$$\gamma \triangleq \max\{\gamma_1, \gamma_2, \ldots, \gamma_m\}.$$

Many algorithms [BT09, BF95, LSS14, QSG13, YFI89] have been developed to solve (4.1) and (4.2). One famous example is the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [BT09], which is a generalization of the optimal method proposed by Nesterov [Nes04] for unconstrained differentiable convex programs. FISTA, with backtracking stepsize strategy, is

known to converge according to the following rate,

$$F(\mathbf{x}_k) - F(\mathbf{x}_\star) \leq \frac{2\eta L \|\mathbf{x}_0 - \mathbf{x}_\star\|^2}{(k+1)^2}, \tag{4.5}$$

where $\mathbf{x}_\star$ is a solution of (4.2), and η is the parameter which is used in the backtracking stepsize strategy. The convergence result in (4.5) contains three key components: the distance between the initial guess and the solution $\|\mathbf{x}_0 - \mathbf{x}_\star\|$, the number of iterations k, and the Lipschitz constant L. While it is clear that the first two components are important to explain the convergence behavior, the Lipschitz constant, L, is relatively mysterious.

The appearance of L in (4.5) is due to algorithm design. In each iteration, one would have to choose a constant $\tilde{L}$ to compute the stepsize that is proportional to $1/\tilde{L}$, and $\tilde{L}$ has to be large enough to satisfy the properties of the Lipschitz constant locally [BT09, QSG13]. Since the global Lipschitz constant condition (4.3) is a more restrictive condition, the Lipschitz constant L always satisfies the requirement of $\tilde{L}$, and so L is used in convergence analysis. We emphasize that the above requirement of $\tilde{L}$ is not unique for FISTA. For most first order methods that solve (4.2), L also appears in their convergence rates for the same reason.

Despite L being an important quantity in both convergence analysis and stepsize strategy, it is usually unknown and the magnitude could be arbitrary for a general nonlinear function; one could artificially construct a low dimensional function with large Lipschitz constant, and a high dimensional function with small Lipschitz constant.

Therefore, L is often treated as a constant [Nes04, Nes13] that is independent of the dimensions of the problem, and so the convergence result shown in (4.5) is considered to be "dimension-free" because both $\|\mathbf{x}_0 - \mathbf{x}_\star\|$ and k are independent of the dimension of the problem. Dimension-free convergence shows that for certain types of optimization algorithms, the number of iterations required to achieve a certain accuracy is independent of the dimension of the model. For large scale optimization models that appear in machine learning and big data applications, algorithms with dimension-free convergence are extremely attractive [BT09, BT13, ST13].

On the other hand, since L is considered to be an arbitrary constant, stepsize strategies for first order methods were developed independent of the knowledge of L. As we will show later, for adaptive strategies that try to use small $\tilde{L}$ (large stepsize), extra function evaluations will be needed. If one tries to eliminate the extra function evaluations, then $\tilde{L}$ has to be sufficiently large, and thus the stepsize would be small. This trade-off is due to the fact that L is unknown.

In this chapter, we take the first steps to show that knowledge of L can be obtained in the case of ERM because of its statistical properties. For the ERM problem, it is known that the Lipschitz constant is highly related to $\|\mathbf{A}\|$ [BT09, QR14], and so understanding the properties of $\|\mathbf{A}\|$ is the goal of this chapter. If $\mathbf{A}$ is arbitrary, then $\|\mathbf{A}\|$ would also be arbitrary and analyzing $\|\mathbf{A}\|$ would be impossible. However, for ERM problems that appear in practice, $\mathbf{A}$ is structured. Since $\mathbf{A}$ is typically constructed from a dataset then it is natural to assume that the rows of $\mathbf{A}$ are independent samples of some random variables. This particular structure of $\mathbf{A}$ allows us to consider $\mathbf{A}$ as a non-arbitrary but random matrix. We are therefore justified to apply techniques from random matrix theory to derive the statistical bounds for the Lipschitz constant.

The contributions of this chapter is twofold:

(a) We obtain average/probabilistic complexity bounds which provide a better understanding of how the dimension, size of training set, and correlation affect the computational complexity. In particular, we show that in the case of ERM, the complexity is not "dimension-free".

(b) The derived statistical bounds can be computed/estimated with almost no cost, which is an attractive benefit for algorithms. We develop a novel stepsize strategy called Probabilistic Upper-bound Guided stepsize strategy (PUG). We show that PUG may save unnecessary cost of function evaluations by adaptively choosing $\tilde{L}$ intelligently. Promising numerical results are provided at the end of this chapter.

Much research on bounding extreme singular values using random matrix theory has been taken in recent years, e.g. see [RV10, KM15, Tro12]. However, we would like to emphasize that

developments in random matrix theory is not our objective. Instead, we would like to consider this topic as a new and important application of random matrix theory. To the best of our knowledge, no similar work has been done in understanding how the statistics of the training set would affect the Lipschitz constant, computational complexity, and stepsize.

4.2 Preliminaries

This chapter studies the Lipschitz constant L of the empirical risk f given in (4.4). In order to satisfy condition (4.3), one could select an arbitrarily large L, however, this would create a looser bound on the complexity (see for e.g. (4.5)). Moreover, L also plays a big role in stepsize strategy for first order algorithms. In many cases such as FISTA, algorithms use stepsize that is proportional to $1/L$. Therefore, a smaller L is always preferable because it does not only imply lower computational complexity, but also allows a larger stepsize for algorithms. While the lowest possible L that satisfies (4.3) is generally very difficult to compute, in this section, we will estimate the upper and lower bounds of L using the dataset $\mathbf{A}$. For the mathematical clarity, we emphasize that $\|\cdot\|$ represents the standard 2-norm, and some equations presented in this chapter do not hold if other norms is taken.

Notice that the Lipschitz constant condition (4.3) is equivalent to the following condition [Nes04].

$$f(\mathbf{y}) \leq f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) + \frac{L}{2}\|\mathbf{y}-\mathbf{x}\|^2, \quad \forall \mathbf{x},\mathbf{y} \in \mathbb{R}^n. \tag{4.6}$$

Therefore, a L that satisfies (4.6) also satisfies (4.3), and vice versa.

Proposition 4.1 *Suppose f is of the form (4.4), then L satisfies the Lipschitz constant condition (4.6) with*

$$L \leq \left\| Diag\left(\sqrt{\frac{\gamma_1}{m}}, \cdots, \sqrt{\frac{\gamma_m}{m}}\right)\mathbf{A}\right\|^2 \leq \left\| Diag\left(\sqrt{\frac{\gamma_1}{m}}, \cdots, \sqrt{\frac{\gamma_m}{m}}\right)\right\|^2 \|\mathbf{A}\|^2 \leq \frac{\gamma}{m}\|\mathbf{A}\|^2,$$

where γ_i is the Lipschitz constant of ϕ_i for $i = 1, 2, \ldots, m$ and $\gamma \triangleq \max\{\gamma_1, \gamma_2, \ldots, \gamma_m\}$.

Proof See Proposition 2.1 in [QR14]. ∎

Proposition 4.1 provides an upper bound for L, where γ is the maximum Lipschitz constant of the loss functions, and it is usually known or easy to compute. For example, it is known that $\gamma = 1$ for quadratic loss functions, and $\gamma = \max_i b_i^2/4$ for logistics loss functions.

The upper bound of L is tight for the class of ERM problems. We can prove that by considering the example of least squares, where we have $\gamma = 1$ and

$$L = \frac{\gamma}{m}\|\mathbf{A}\|^2 = \frac{1}{m}\|\mathbf{A}\|^2.$$

In order to derive the lower bound of L, we need the following assumption.

Assumption 4.2 *The loss function ϕ_i is a strongly convex function, for $i = 1, 2, \ldots, m$. That is, there exists a positive constant $\tau > 0$ such that*

$$\phi_i(x) + \phi_i'(x)(y - x) + \frac{\tau}{2}|y - x|^2 \leq \phi_i(y), \quad \forall x, y \in \mathbb{R},$$

for $i = 1, 2, \ldots, m$.

The above assumption requires the loss function ϕ_i to be strongly convex, which is not restrictive in practical setting. In particular, the quadratic loss function satisfies Assumption 4.2, and the logistics loss function satisfies Assumption 4.2 within a bounded box $[-b, b]$ for any positive $b \in \mathbb{R}^+$. With the above assumption, we derive the lower bound of L using $\mathbf{A}$.

Proposition 4.3 *Suppose f is of the form (4.4) with ϕ_i satisfying Assumption 4.2 for $i = 1, 2, \ldots, m$, then L satisfies the Lipschitz constant condition (4.6) with*

$$\frac{\tau\lambda_{\min}(\mathbf{A}^T\mathbf{A})}{m} \leq L.$$

Proof By Assumption 4.2, for $i = 1, 2, \ldots, m$,

$$\phi_i(\mathbf{e}_i^T\mathbf{A}\mathbf{y}) \geq \phi_i(\mathbf{e}_i^T\mathbf{A}\mathbf{x}) + \phi_i'(\mathbf{e}_i^T\mathbf{A}\mathbf{x})(\mathbf{e}_i^T\mathbf{A}\mathbf{y} - \mathbf{e}_i^T\mathbf{A}\mathbf{x}) + \frac{\tau}{2}|\mathbf{e}_i^T\mathbf{A}\mathbf{y} - \mathbf{e}_i^T\mathbf{A}\mathbf{x}|^2.$$

Therefore,

$$\begin{aligned} f(\mathbf{y}) &\geq \frac{1}{m}\sum_{i=1}^{m}\left(\phi_i(\mathbf{e}_i^T\mathbf{A}\mathbf{x}) + \phi_i'(\mathbf{e}_i^T\mathbf{A}\mathbf{x})(\mathbf{e}_i^T\mathbf{A}\mathbf{y} - \mathbf{e}_i^T\mathbf{A}\mathbf{x}) + \frac{\tau}{2}|\mathbf{e}_i^T\mathbf{A}\mathbf{y} - \mathbf{e}_i^T\mathbf{A}\mathbf{x}|^2\right), \\ &= f(\mathbf{x}) + \frac{1}{m}\sum_{i=1}^{m}\left(\mathbf{e}_i^T\mathbf{A}\phi_i'(\mathbf{e}_i^T\mathbf{A}\mathbf{x})(\mathbf{y}-\mathbf{x}) + \frac{\tau}{2}|\mathbf{e}_i^T\mathbf{A}\mathbf{y} - \mathbf{e}_i^T\mathbf{A}\mathbf{x}|^2\right), \\ &= f(\mathbf{x}) + \langle\nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle + \frac{\tau}{2m}\|\mathbf{A}\mathbf{y} - \mathbf{A}\mathbf{x}\|^2, \\ &\geq f(\mathbf{x}) + \langle\nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle + \frac{\tau\lambda_{\min}(\mathbf{A}^T\mathbf{A})}{2m}\|\mathbf{y}-\mathbf{x}\|^2. \end{aligned}$$

∎

From Proposition 4.1 and 4.3, we bound L using the largest and lowest eigenvalues of $\mathbf{A}^T\mathbf{A}$. Even though $\mathbf{A}$ can be completely different for different dataset, the statistical properties of $\mathbf{A}$ can be obtained via random matrix theory.

4.3 Complexity Analysis using Random Matrix Theory

In this section, we will study the statistical properties of $\|\mathbf{A}\|^2 = \|\mathbf{A}^T\mathbf{A}\| = \lambda_{\max}(\mathbf{A}^T\mathbf{A})$ as well as $\lambda_{\min}(\mathbf{A}^T\mathbf{A})$. Recall that $\mathbf{A}$ is an $m \times n$ matrix containing m observations, and each observation contains n measurements which are independent samples from n random variables, i.e. we assume the rows of the matrix $\mathbf{A}$ are samples from a vector of n random variables $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$ with covariance matrix $\boldsymbol{\Sigma}$. To simplify the analysis, we assume, without loss of generality, that the observations are normalized, and so all the random variables have mean zero and unit variance. Therefore, $\mathbb{E}[\xi_i] = 0$ for $i = 1, 2, \cdots, n$, the diagonal elements of $\boldsymbol{\Sigma}$ are all 1's, and $\boldsymbol{\Sigma} = \mathbb{E}\left[\boldsymbol{\xi}\boldsymbol{\xi}^T\right]$. This assumption is useful and simplifies the arguments and the analysis of this section but it is not necessary. The results from this section could be generalized without the above assumption, but it does not give further insights for the purposes of this section. In particular, this assumption will be dropped for the proposed stepsize strategy PUG, and so PUG is vaild for all the datasets used in practice.

4.3.1 Statistical Bounds

We will derive both the upper and lower bounds for the expected $\|\mathbf{A}\|^2$, and show that the average $\|\mathbf{A}\|^2$ increases nearly linearly in both m and n. The main tools for the proofs below can be found in Chapter 2 and [Tro12].

Lower Bounds

We will start by proving the lower bound in the general setting, where the random variables are correlated with general covariance matrix $\mathbf{\Sigma}$; then, we will add assumptions on $\mathbf{\Sigma}$ to derive lower bounds in different cases.

Theorem 4.4 *Let $\mathbf{A}$ be an $m \times n$ random matrix in which its rows are independent samples of some random variables $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$ with $\mathbb{E}[\xi_i] = 0$ for $i = 1, 2, \cdots, n$, and covariance matrix $\mathbf{\Sigma}$. Denote $\mu_{\max} = \lambda_{\max}(\mathbf{\Sigma})$. Then*

$$m\mu_{\max} = m\lambda_{\max}(\mathbf{\Sigma}) \leq \mathbb{E}\left[\|\mathbf{A}\|^2\right]. \tag{4.7}$$

In particular, if $\xi_1, \xi_2, \cdots, \xi_n$ are some random variables with zero mean and unit variance, then

$$\max\{m\mu_{\max}, n\} \leq \mathbb{E}\left[\|\mathbf{A}\|^2\right]. \tag{4.8}$$

Proof We first prove (4.7). Denote $\mathbf{a}_i^T$ as the i^{th} row of $\mathbf{A}$. We can rewrite $\mathbf{A}^T\mathbf{A}$ as

$$\mathbf{A}^T\mathbf{A} = \sum_{k=1}^{m} \mathbf{a}_k\mathbf{a}_k^T,$$

where $\mathbf{a}_k\mathbf{a}_k^T$'s are independent random matrices with $\mathbb{E}\left[\mathbf{a}_k\mathbf{a}_k^T\right] = \mathbf{\Sigma}$. Therefore, by Jensen's inequality

$$\mathbb{E}\left[\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right)\right] = \mathbb{E}\left[\lambda_{\max}\left(\sum_{k=1}^{m}\mathbf{a}_k\mathbf{a}_k^T\right)\right] \geq \lambda_{\max}\left(\sum_{k=1}^{m}\mathbb{E}\left[\mathbf{a}_k\mathbf{a}_k^T\right]\right) = m\lambda_{\max}(\mathbf{\Sigma}).$$

In order to prove (4.8), we use the fact that

$$\mathbb{E}\left[\|\mathbf{A}\|^2\right] = \mathbb{E}\left[\|\mathbf{A}^T\|^2\right] = \mathbb{E}\left[\|\mathbf{A}\mathbf{A}^T\|\right] \geq \|\mathbb{E}\left[\mathbf{A}\mathbf{A}^T\right]\|,$$

where the last inequality is obtained by applying Jensen's inequality. We can write $\mathbf{A}\mathbf{A}^T$ as

$$\mathbf{A}\mathbf{A}^T = \sum_{i=1}^{m}\sum_{j=1}^{m} \mathbf{a}_i^T \mathbf{a}_j \mathbf{Y}_{i,j},$$

where $\mathbf{Y}_{i,j} \in \mathbb{R}^{m\times m}$ is a matrix such that $(\mathbf{Y}_{i,j})_{p,q} = 1$ if $i = p$ and $j = q$, and otherwise $(\mathbf{Y}_{i,j})_{p,q} = 0$. By the assumption that each entry of $\mathbf{A}$ is a random variable with zero mean and unit variance, we obtain

$$\mathbb{E}\left[\mathbf{a}_i^T \mathbf{a}_i\right] = \mathbb{E}\left[a_{i,1}^2 + a_{i,2}^2 + \cdots + a_{i,n}^2\right] = \mathbb{E}\left[a_{i,1}^2\right] + \mathbb{E}\left[a_{i,2}^2\right] + \cdots + \mathbb{E}\left[a_{i,n}^2\right] = n,$$

for $i = 1, 2, \cdots, m$, and for $i \neq j$,

$$\mathbb{E}\left[\mathbf{a}_i^T \mathbf{a}_j\right] = \mathbb{E}\left[a_{i,1}\right]\mathbb{E}\left[a_{j,1}\right] + \mathbb{E}\left[a_{i,2}\right]\mathbb{E}\left[a_{j,2}\right] + \cdots + \mathbb{E}\left[a_{i,n}\right]\mathbb{E}\left[a_{j,n}\right] = 0.$$

Therefore,

$$\mathbb{E}[\|\mathbf{A}\|^2] \geq \|\mathbb{E}[\mathbf{A}\mathbf{A}^T]\| = \left\|\mathbb{E}\left[\sum_{i=1}^{m}\sum_{j=1}^{m} \mathbf{a}_i^T \mathbf{a}_j \mathbf{Y}_{i,j}\right]\right\| = \left\|\sum_{i=1}^{m} n\mathbf{Y}_{i,i}\right\| = \|n\mathbf{I}_n\| = n.$$

∎

Theorem 4.4 provides a lower bound for the expected $\|\mathbf{A}^T\mathbf{A}\|$. The inequality in (4.7) is a general result and makes minimal assumptions on the covariance $\mathbf{\Sigma}$. Note that the lower bound is independent of n. The reason is that this general setting covers cases where $\mathbf{\Sigma}$ is not full rank: some ξ_i's could be fixed 0's instead of having unit variance. In fact, when all ξ_i's are 0's for $i = 1, 2, \cdots, n$, which implies $\mathbf{\Sigma} = \mathbf{0}_{n\times n}$, the bound (4.7) is tight because $\mathbf{A} = \mathbf{0}_{m\times n}$. For the setting that we consider in this chapter, equation (4.8) is a tighter bound than (4.7) and depends on both m and n. In the case where all variables are independent, we could simplify

the results above into the following.

Corollary 4.5 *Let* $\mathbf{A}$ *be an* $m \times n$ *random matrix in which its rows are independent samples of some random variables* $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$ *with* $\mathbb{E}[\xi_i] = 0$, $\mathbb{E}[\xi_i^2] = 1$, *and* ξ_i*'s are independent for* $i = 1, 2, \cdots, n$, *then*

$$\max\{m, n\} \leq \mathbb{E}\left[\|\mathbf{A}\|^2\right]. \tag{4.9}$$

Proof Since all random variables are independent, $\boldsymbol{\Sigma} = \mathbf{I}_n$ and so $\mu_{\max} = \lambda_{\max}(\boldsymbol{\Sigma}) = 1$. ∎

Upper Bounds

In order to compute an upper bound for the expected $\|\mathbf{A}^T\mathbf{A}\|$, we first compute its tail bounds. The idea of the proof is to rewrite the $\mathbf{A}^T\mathbf{A}$ as a sum of independent random matrices, and then use the existing results in random matrix theory to derive the tail bounds of $\|\mathbf{A}^T\mathbf{A}\|$. We then compute the upper bound for the expected value. Notice that our approach for computing the tail bounds, in principle, is the same as in [Tro12]. However, we present a tail bound that is easier to be integrated into the upper bound of the expected $\|\mathbf{A}^T\mathbf{A}\|$. That is, the derived bound can be directly used to bound $\|\mathbf{A}^T\mathbf{A}\|$ without any numerical constant.

Lemma 4.6 *Let* $\mathbf{A}$ *be an* $m \times n$ *random matrix in which its rows are independent samples of some random variables* $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$ *with covariance matrix* $\boldsymbol{\Sigma}$. *Denote* $\mu_{\max} = \lambda_{\max}(\boldsymbol{\Sigma})$ *and suppose*

$$\lambda_{\max}\left[\boldsymbol{\xi}\boldsymbol{\xi}^T\right] \leq R \quad \textit{almost surely.} \tag{4.10}$$

Then, for any $\theta, t \in \mathbb{R}^+$,

$$\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \geq t\right\} \leq n \exp\left[-\theta t + m \ \log\left(1 + (e^{\theta R} - 1)\mu_{\max}/R\right)\right]. \tag{4.11}$$

In particular,

$$\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \geq t\right\} \leq n \left[\frac{\mu_{\max}(mR - t)}{t(R - \mu_{\max})}\right]^{\frac{t}{R}} \left[1 + \frac{t - \mu_{\max} m}{mR - t}\right]^m. \tag{4.12}$$

Proof Denote $\mathbf{a}_i^T$ as the i^{th} row of $\mathbf{A}$. We can rewrite $\mathbf{A}^T\mathbf{A}$ as

$$\mathbf{A}^T\mathbf{A} = \sum_{k=1}^{m} \mathbf{a}_k\mathbf{a}_k^T.$$

Notice that $\mathbf{a}_k\mathbf{a}_k^T$'s are independent, random, positive-semidefinite matrices, and $\mathbb{E}\left[\mathbf{a}_k\mathbf{a}_k^T\right] = \mathbf{\Sigma}$, for $k = 1, 2, \cdots, m$. Also, Using the Lemma 2.13, for any $\theta > 0$, we have

$$\begin{aligned}
\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \geq t\right\} &= \mathbb{P}\left\{\lambda_{\max}\left(\sum_{k=1}^{m} \mathbf{a}_k\mathbf{a}_k^T\right) \geq t\right\}, \\
&\leq n\exp\left[-\theta t + m\ \log\lambda_{\max}\left(\frac{1}{m}\sum_{k=1}^{m}\mathbb{E}e^{\theta\mathbf{a}_k\mathbf{a}_k^T}\right)\right].
\end{aligned}$$

Notice that $\lambda_{\max}(\mathbf{a}_k\mathbf{a}_k^T) \leq R$, by rescaling on Lemma 2.12, we have,

$$\mathbb{E}\left[e^{\tilde{\theta}(1/R)(\mathbf{a}_k\mathbf{a}_k^T)}\right] \preccurlyeq \mathbf{I}_n + (e^{\tilde{\theta}} - 1)\left(\mathbb{E}\left[(1/R)\left(\mathbf{a}_k\mathbf{a}_k^T\right)\right]\right), \quad \text{for any } \tilde{\theta} \in \mathbb{R},$$

and thus

$$\mathbb{E}\left[e^{\theta(\mathbf{a}_k\mathbf{a}_k^T)}\right] \preccurlyeq \mathbf{I}_n + \frac{(e^{\theta R} - 1)}{R}\mathbb{E}\left[\mathbf{a}_k\mathbf{a}_k^T\right] = \mathbf{I}_n + \frac{(e^{\theta R} - 1)}{R}\mathbf{\Sigma}, \quad \text{for any } \theta \in \mathbb{R}.$$

$$\begin{aligned}
\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \geq t\right\} &\leq n\exp\left[-\theta t + m\ \log\lambda_{\max}\left(\frac{1}{m}\sum_{k=1}^{m}\mathbf{I}_n + \frac{(e^{\theta R} - 1)}{R}\mathbf{\Sigma}\right)\right], \\
&= n\exp\left[-\theta t + m\ \log\left(1 + (e^{\theta R} - 1)\lambda_{\max}(\mathbf{\Sigma})/R\right)\right], \\
&= n\exp\left[-\theta t + m\ \log\left(1 + (e^{\theta R} - 1)\mu_{\max}/R\right)\right],
\end{aligned}$$

where the penultimate equality is valid since

$$\lambda_{\max}(\mathbf{I}_n + \mathbf{\Sigma}) = 1 + \lambda_{\max}(\mathbf{\Sigma}).$$

Using standard calculus, the upper bound is minimized when

$$\theta^\star = \frac{1}{R}\log\left[\frac{t(R - \mu_{\max})}{\mu_{\max}(mR - t)}\right].$$

Therefore,

$$\begin{aligned}\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \geq t\right\} &\leq n\exp\left[-\theta^\star t + m\ \log\left(1+(e^{\theta^\star R}-1)\mu_{\max}/R\right)\right], \\ &= n\left[\frac{\mu_{\max}(mR-t)}{t(R-\mu_{\max})}\right]^{\frac{t}{R}}\left[1+\frac{t-\mu_{\max}m}{mR-t}\right]^m.\end{aligned}$$

∎

We emphasize that Assumption (4.10) in Lemma 4.6 does not hold for unbounded random variables; however, in practice, assumption (4.10) is mild due to the fact that datasets that are used in the problem (4.1) are usually normalized and bounded. Therefore, it is reasonable to assume that an observation will be discarded if its magnitude is larger than some constant.

We clarify that the tail bound (4.11) is only meaningful when t is sufficiently large; otherwise the right hand side would be greater than or equal to one. The tail bound (4.12) is the tightest bound over all possible θ's in (4.11), but it is difficult to interpret the relationships between the variables. The following lemma takes a less optimal θ in (4.11), but yields a bound that is easier to understand.

Lemma 4.7 *In the same setting as Lemma 4.6, we have*

$$\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \geq t\right\} \leq n\exp\left[\frac{2m\mu_{\max}-t}{R}\right]. \tag{4.13}$$

In particular, for $\epsilon \in \mathbb{R}$, we have

$$\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \leq 2m\mu_{\max} - R\log\left(\frac{\epsilon}{n}\right)\right\} \geq 1-\epsilon. \tag{4.14}$$

Proof Using equation (4.11), and the fact that $log(y) \leq y-1$, $\forall y \in \mathbb{R}^+$, we have

$$\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \geq t\right\} \leq n\exp\left[-\theta t + \frac{m\mu_{\max}}{R}(e^{\theta R}-1)\right]. \tag{4.15}$$

The above upper bound is minimized when $\theta = (1/R)\log\left[t/(m\mu_{\max})\right]$, and so

$$\begin{aligned}
\mathbb{P}\left\{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right) \geq t\right\} &\leq n\exp\left[-\frac{t}{R}\log\left[\frac{t}{m\mu_{\max}}\right] + \frac{m\mu_{\max}}{R}\left(\frac{t}{m\mu_{\max}} - 1\right)\right], \\
&= n\exp\left[\frac{t}{R}\left(-\log\left[\frac{t}{m\mu_{\max}}\right] + m\mu_{\max}\left(\frac{1}{m\mu_{\max}} - \frac{1}{t}\right)\right)\right], \\
&= n\exp\left[\frac{t}{R}\left(\log\left[\frac{m\mu_{\max}}{t}\right] + 1 - \frac{m\mu_{\max}}{t}\right)\right], \\
&= n\exp\left[\frac{t}{R}\left(\log\left[\frac{m\mu_{\max}e}{t}\right] - \frac{m\mu_{\max}}{t}\right)\right], \\
&\leq n\exp\left[\frac{t}{R}\left(\frac{m\mu_{\max}e}{t} - 1 - \frac{m\mu_{\max}}{t}\right)\right], \\
&= n\exp\left[\frac{1}{R}\left(m\mu_{\max}(e-1) - t\right)\right], \\
&\leq n\exp\left[\frac{1}{R}\left(2m\mu_{\max} - t\right)\right].
\end{aligned}$$

Set $\epsilon = n\exp\left[\frac{1}{R}\left(2m\mu_{\max} - t\right)\right]$, we obtain $t = 2m\mu_{\max} - R\log\left(\frac{\epsilon}{n}\right)$. ∎

The bound in (4.14) follows directly from (4.13) and shows that with high probability $1-\epsilon$ (for small ϵ), $\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right)$ is less than $2m\mu_{\max} + R\log(n) - R\log(\epsilon)$. Applying the results in Lemma 4.7 provides the upper bound for the expected $\|\mathbf{A}^T\mathbf{A}\|$.

Theorem 4.8 *In the same setting as Lemma 4.6, we have*

$$\mathbb{E}\left[\lambda_{\max}(\mathbf{A}^T\mathbf{A})\right] \leq 2m\mu_{\max} + R\log(n) + R. \tag{4.16}$$

Proof Using the equation (4.13), and the fact that

$$1 \leq n\exp\left[\frac{2m\mu_{\max} - t}{R}\right] \quad \text{when} \quad t \leq 2m\mu_{\max} - R\log\left[\frac{1}{n}\right],$$

we have

$$\begin{aligned}\mathbb{E}\left[\lambda_{\max}(\mathbf{A}^T\mathbf{A})\right] &= \int_0^\infty \mathbb{P}\{\lambda_{\max}(\mathbf{A}^T\mathbf{A}) > t\}\ \mathrm{d}t, \qquad \text{(see [Bil12] for details)} \\ &\leq \int_0^{2m\mu_{\max}-R\log\left[\frac{1}{n}\right]} 1\ \mathrm{d}t + \int_{2m\mu_{\max}-R\log\left[\frac{1}{n}\right]}^\infty n\exp\left[\frac{2m\mu_{\max}-t}{R}\right]\ \mathrm{d}t, \\ &= 2m\mu_{\max} - R\log\left[\frac{1}{n}\right] + R.\end{aligned}$$

∎

Therefore, for a matrix $\mathbf{A}$ which is constructed by a set of normalized data, we obtain the bound

$$\max\{m\mu_{\max}, n\} \leq \mathbb{E}\left[\|\mathbf{A}\|^2\right] \leq 2m\mu_{\max} + R\log(n) + R. \tag{4.17}$$

The result in (4.17) might look confusing because for small m and large n, the lower bound is of the order of n while the upper bound is of the order of $\log(n)$. The reason is that we have to take into account the factor of dimensionality in the constant R. To illustrate this, we prove the following corollary.

Corollary 4.9 *Let* $\mathbf{A}$ *be an* $m \times n$ *random matrix in which its rows are independent samples of some random variables* $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$ *with* $\mathbb{E}[\xi_i] = 0$ *for* $i = 1, 2, \cdots, n$, *and covariance matrix* $\boldsymbol{\Sigma} = \mathbb{E}\left[\boldsymbol{\xi}\boldsymbol{\xi}^T\right]$. *Denote* $\mu_{\max} = \lambda_{\max}(\boldsymbol{\Sigma})$ *and suppose* $|\xi_i| \leq c$ *almost surely for* $i = 1, 2, \cdots, n$. *Then*

$$\lambda_{\max}\left[\boldsymbol{\xi}\boldsymbol{\xi}^T\right] \leq c^2 n \quad \textit{almost surely.} \tag{4.18}$$

and so

$$\max\{m\mu_{\max}, n\} \leq \mathbb{E}\left[\|\mathbf{A}\|^2\right] \leq 2m\mu_{\max} + c^2 n\log(n) + c^2 n \tag{4.19}$$

Proof Since $\boldsymbol{\xi}\boldsymbol{\xi}^T$ is a symmetric rank 1 matrix, we have

$$\lambda_{\max}(\boldsymbol{\xi}\boldsymbol{\xi}^T) = \|\boldsymbol{\xi}\boldsymbol{\xi}^T\| \leq n\|\boldsymbol{\xi}\boldsymbol{\xi}^T\|_{\max} = n\max_{1\leq i,j\leq n}\{|\xi_i\xi_j|\} \leq c^2 n \quad \text{almost surely.}$$

∎

Therefore, R increases linearly in n for bounded $\boldsymbol{\xi}$. Recall that the lower bound of the expected $\|\mathbf{A}\|^2$ is linear in both m and n, and the upper bound in (4.19), is almost linear in both m and n. Therefore, our results on the bounds for the expected Lipschitz constant are nearly-optimal up to some constant.

On the other hand, in order to obtain the lower bound of L, we also need tail bound of $\lambda_{\min}(\mathbf{A}^T\mathbf{A})$, which is provided in the following theorem.

Theorem 4.10 *Let* $\mathbf{A}$ *be an* $m \times n$ *random matrix in which its rows are independent samples of some random variables* $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$ *with covariance matrix* $\boldsymbol{\Sigma}$. *Denote* $\mu_{\min} = \lambda_{\min}(\boldsymbol{\Sigma})$ *and suppose* $|\xi_i| \leq c$ *almost surely for* $i = 1, 2, \ldots, n$.

Then, if $\mu_{\min} \neq 0$, *for any* $\epsilon \in \left(n \exp\left[-\frac{m\mu_{\min}}{2nc^2}\right] \ , n\right)$

$$\mathbb{P}\left\{\lambda_{\min}\left(\mathbf{A}^T\mathbf{A}\right) \leq \sqrt{2c^2 nm\mu_{\min} \log\left(\frac{n}{\epsilon}\right)} + m\mu_{\min}\right\} \leq \epsilon.$$

Proof Suppose $|\xi_i| \leq c$ almost surely for $i = 1, 2, \ldots, n$. Then using Corollary 4.9 we have

$$\lambda_{\max}\left[\boldsymbol{\xi}\boldsymbol{\xi}^T\right] \leq c^2 n = R \quad \text{almost surely.}$$

Using the Theorem 1.1 from [Tro12], for any $\theta \in (0, 1)$ we have

$$\begin{aligned} \mathbb{P}\left\{\lambda_{\min}\left(\mathbf{A}^T\mathbf{A}\right) \leq \theta m\mu_{\min}\right\} &\leq n\left[\frac{\exp[\theta - 1]}{\theta^\theta}\right]^{m\mu_{\min}/R}, \\ &= n \exp\left[(-(1-\theta) - \theta\log(\theta))\left(\frac{m\mu_{\min}}{R}\right)\right]. \end{aligned}$$

Notice that $\theta > 0$ and

$$2\log(\theta) \geq 2\left(1 - \frac{1}{\theta}\right) = \frac{2(\theta - 1)}{\theta} \geq \frac{(\theta + 1)(\theta - 1)}{\theta} = \frac{\theta^2 - 1}{\theta},$$

and so

$$\begin{aligned}\mathbb{P}\left\{\lambda_{\min}\left(\mathbf{A}^T\mathbf{A}\right) \leq \theta m\mu_{\min}\right\} &\leq n\exp\left[\left(-(1-\theta)-\theta\log(\theta)\right)\left(\frac{m\mu_{\min}}{R}\right)\right], \\ &\leq n\exp\left[\left(-(1-\theta)-\theta\frac{\theta^2-1}{2\theta}\right)\left(\frac{m\mu_{\min}}{R}\right)\right], \\ &= n\exp\left[-\frac{1}{2}\left(\theta-1\right)^2\left(\frac{m\mu_{\min}}{R}\right)\right].\end{aligned}$$

For $\mu_{\min} \neq 0$, we let $\epsilon = n\exp\left[-\left(\theta-1\right)^2\frac{m\mu_{\min}}{2R}\right]$ and obtain,

$$\theta = \sqrt{\frac{2R}{m\mu_{\min}}\log\left(\frac{n}{\epsilon}\right)} + 1.$$

Therefore,

$$\mathbb{P}\left\{\lambda_{\min}\left(\mathbf{A}^T\mathbf{A}\right) \leq \sqrt{2Rm\mu_{\min}\log\left(\frac{n}{\epsilon}\right)} + m\mu_{\min}\right\} \leq \epsilon,$$

for $\epsilon \in \left(n\exp\left[-\frac{m\mu_{\min}}{2R}\right]\ , n\right)$ ∎

For the tail bound in Theorem 4.10 to be meaningful, m has to be sufficiently large compared to n. Also, $\mu_{\min}$ is required to be non-zero, which implies that all independent variables (ξ_i's) are linearly independent. In such cases, the smallest eigenvalue $\lambda_{\min}\left(\mathbf{A}^T\mathbf{A}\right)$ is at least $\mathcal{O}(\sqrt{nm\log n} + m)$ with high probability.

4.3.2 Complexity Analysis

In this section, we will use the probabilistic bounds of L to study the complexity of solving ERM. We focus only on FISTA for illustrative purpose and clear presentation of the idea of the proposed approach. But the approach developed in this section can be applied to other algorithms as well.

By the assumption that $\mathbf{A}$ is a random matrix, we also have the solution $\mathbf{x}_\star$ as a random vector. Notice that the study of randomization of $\mathbf{x}_\star$ is not covered this chapter. In particular, if the statistical properties of $\mathbf{x}_\star$ can be obtained, existing optimization algorithms might not be needed to solve the ERM problem. Therefore, in this chapter, we remove this consideration by

denoting a constant M such that $\|\mathbf{x}_0 - \mathbf{x}_\star\|^2 \leq M$. In such case, we have the FISTA convergence rate

$$F(\mathbf{x}_k) - F(\mathbf{x}_\star) \leq \frac{2\eta L M}{(k+1)^2}, \tag{4.20}$$

where $\mathbf{x}_\star$ is the solution of (4.1), and $\eta > 1$ is the parameter which is used in the backtracking stepsize strategy.

By Corollary 4.9, we know

$$\max\left\{\gamma\mu_{\max}, \frac{\gamma n}{m}\right\} \leq \frac{\gamma}{m}\mathbb{E}\left[\|\mathbf{A}\|^2\right] \leq 2\gamma\mu_{\max} + \frac{\gamma}{m}(c^2 n \log(n) + c^2 n), \tag{4.21}$$

and we are now in the position for the following theorem.

Theorem 4.11 *Consider the composite program (4.2) with f that is in the form of (4.4), where $\mathbf{A}$ is an $m \times n$ random matrix in which its rows are independent samples of some random variables $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$ with covariance matrix $\boldsymbol{\Sigma}$. Suppose $|\xi_i| \leq c$ almost surely for $i = 1, 2, \cdots, n$ and $\|\mathbf{x}_0 - \mathbf{x}_\star\|^2 \leq M$ for some positive constant M. Then the expected convergence of FISTA with backtracking stepsize strategy is*

$$\mathbb{E}[F(\mathbf{x}_k) - F(\mathbf{x}_\star)] \leq \frac{2\eta M}{(k+1)^2}\left(2\gamma\mu_{\max} + \frac{\gamma}{m}(c^2 n \log(n) + c^2 n)\right), \tag{4.22}$$

where $\mu_{\max} = \lambda_{\max}(\boldsymbol{\Sigma})$, $\eta > 1$ is the parameter which is used in the backtracking stepsize strategy, and the expectation is taken with respect to the uncertainty of $\mathbf{A}$.

Proof This result can be obtained by combining results in Proposition 4.1, Corollary 4.9, and (4.20) (or equivalently see (4.5)). ∎

In (4.21), the lower bound of $(\gamma/m)\mathbb{E}\left[\|\mathbf{A}\|^2\right]$ is linear in n/m, and upper bound is nearly-linear in n/m. This suggests that the average complexity of ERM is bounded by the ratio of the dimensions to the amount of data. In particular, problems with overdetermined systems ($m \gg n$) can be solved more efficiently than problems with underdetermined systems ($m < n$).

Another critical factor of the complexity is $\mu_{\max} = \lambda_{\max}(\mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is the covariance matrix of the rows of $\mathbf{A}$. In the ideal situation of regression analysis, all inputs should be statistically linearly independent. In such cases, since we assume the diagonal elements of $\mathbf{\Sigma}$ are 1's, $\mu_{\max} = 1$. It is, however, almost impossible to ensure this situation for practical applications. In practice, since $\mathbf{\Sigma} \in \mathbb{R}^{n\times n}$, $\mu_{\max} = \lambda_{\max}(\mathbf{\Sigma}) = \|\mathbf{\Sigma}\|$ is likely to increase as n increases.

Similarly we can compute the probabilistic lower bound of L in the case that m is sufficiently larger than n. Using Theorem 4.10, we can show that L is bounded above by

$$\mathcal{O}\left(\sqrt{(n\log n)/m} + \mu_{\min}\right).$$

We emphasize the lower bound of L is not equivalent to the lower bound of the complexity. However, since the stepsize of first order method algorithms is proportional to $1/L$, this result indicates that high dimensional problems might have smaller stepsize in order to guarantee convergence.

4.4 PUG: Probabilistic Upper-bound Guided stepsize strategy

The tail bounds in Section 4.3, as a by-product of the upper bound in Section 4.3.1, can also be used in algorithms. As mentioned in the introduction, L is an important quantity in the stepsize strategy since the stepsize is usually inversely proportional to L. However, in large scale optimization, the computational cost of evaluating $\|\mathbf{A}\|^2$ is very expensive. One could use backtracking techniques to avoid the evaluation of the Lipschitz constant; in each iteration, we find a large enough constant $\tilde{L}$ such that it satisfies the properties of the Lipschitz constant locally. In the case of FISTA [BT09], for the k^{th} iteration with incumbent $\mathbf{x}_k$ one has to find a $\tilde{L}$ such that

$$F\left(p_{\tilde{L}}(\mathbf{x}_k)\right) \leq Q_{\tilde{L}}\left(p_{\tilde{L}}(\mathbf{x}_k), \mathbf{x}_k\right), \tag{4.23}$$

where,

$$Q_{\tilde{L}}(\mathbf{x},\mathbf{y}) \triangleq f(\mathbf{y}) + \langle \mathbf{x}-\mathbf{y}, \nabla f(\mathbf{y})\rangle + \frac{\tilde{L}}{2}\|\mathbf{x}-\mathbf{y}\|^2 + g(\mathbf{x}),$$

and $p_{\tilde{L}}(\mathbf{y}) \triangleq \arg\min_{\mathbf{x}}\{Q_{\tilde{L}}(\mathbf{x},\mathbf{y}) : \mathbf{x} \in \mathbb{R}^n\}$. Equation (4.23) is identical to the Lipschitz constant condition (4.6) with specifically $\mathbf{y} = p_{\tilde{L}}(\mathbf{x}_k)$ and $\mathbf{x} = \mathbf{x}_k$. Therefore, (4.23) is a less restrictive condition compared to the Lipschitz constant condition (4.6). This indicates that $\tilde{L}$ could be much smaller than L, and so it yields to a larger stepsize. On the other hand, for $\tilde{L} \geq L$, it is guaranteed that the local Lipschitz constant condition will be satisfied. In both cases, when L is intractable, we would not be able to distinguish the two cases by just having $\tilde{L}$ that satisfies (4.23).

As we can see, finding a good $\tilde{L}$ involves a series of function evaluations. In the next section, we will review the commonly used stepsize strategies.

4.4.1 Current Stepsize Strategies

To the best of our knowledge, current strategies fall into four categories:

(i). A fixed stepsize from estimation of $\|\mathbf{A}\|^2$.

(ii). Backtracking-type methods with initial guess $\tilde{L}_0$, and monotonic increases $\tilde{L} = \eta^p \tilde{L}_0$ when it does not satisfy the Lipschitz condition locally ($\eta > 1$, $p = 0, 1, \dots$). See [BT09] for details.

(iii). Adaptive-type methods with initial guess $\tilde{L}_0$. Suppose $\tilde{L}_k$ is used for the k^{th} iteration, then find the smallest p such that $\tilde{L}_{k+1} = 2^p \tilde{L}_k$ satisfies the Lipschitz condition locally ($p = -1, 0, 1, \dots$). See Nesterov's universal gradient methods [Nes15] for details.

(iv). Adaptive stepsize strategies for a specific algorithm. See [GK13] for example.

Theorem 4.12 *Suppose $\tilde{L}$ is used as an initial guess for the k^{th} iteration, and we select the smallest $q \in \mathbb{N}$ such that $\tilde{L}_k = \eta^q \tilde{L}$ satisfies the local condition, for $\eta \geq 1$. To guarantee*

convergence, it requires

$$q \geq \max\left\{\frac{1}{\log \eta}\left(\log L - \log \tilde{L}\right), 0\right\},$$

which is also the number of function evaluations required. We have

$$\begin{aligned} L \leq \tilde{L}_k \leq \eta L, \qquad & \textit{if} \quad \tilde{L} \leq L, \\ L \leq \tilde{L}_k = \tilde{L}, \qquad & \textit{if} \quad L \leq \tilde{L}. \end{aligned}$$

Proof To guarantee convergence, it requires q such that $\tilde{L}_k = \eta^q \tilde{L} \geq L$. If $\tilde{L} \leq L$, q should be selected such that $\eta^q \tilde{L} \leq \eta L$; otherwise $q - 1$ will be large enough to be selected, i.e. $\tilde{L}_k = \eta^{q-1}\tilde{L} \geq L$. ∎

Theorem 4.12 covers the setting of choice (i)-(iii), also referred to as the fixed stepsize strategy, backtracking method, and Nesterov's adaptive method, respectively. For fixed stepsize strategies, $\tilde{L} \geq L$ is selected for all iterations, which yields $q = 0$, and thus checking the local condition is not required [BT09]. For backtracking methods, $\tilde{L} = \tilde{L}_{k-1}$ and $\eta > 1$ is a parameter of the strategy. Since $\tilde{L}_k$ is monotonically increasing in k, q is monotonically decreasing. Therefore, q at the k^{th} iteration is equivalent to the total number of (extra) function evaluations for the rest of the iterations.

On the other hand, for Nesterov's adaptive method, $\tilde{L} = \tilde{L}_{k-1}/2$ and $\eta = 2$. $\tilde{L}_k$ is not monotonically increasing in k, and in each iteration, q is the number of function evaluations in the worst case. Notice that once the worst case occurs (having q function evaluations) in the k^{th} iteration, q will be smaller since $\tilde{L}_k$ is sufficiently large. In Nesterov's universal gradient methods [Nes15], Nesterov proved that for k iterations, the number of function evaluations is bounded by $\mathcal{O}(2k)$.

Theorem 4.12 illustrates the trade-off between three aspects: aggressiveness of initial guess $\tilde{L}$, recovering rate η, and the convergence rate. Methods with small (aggressive) initial guess $\tilde{L}$ have the possibility to result in larger stepsize. However, it will yield a larger q, the number of function evaluations in the worst case. One could reduce q by setting a larger η, and so $\tilde{L}$ could scale quickly towards L, but it will generate a slower rate of convergence (ηL). If one

wants to preserve a good convergence rate (small η) with small number of function evaluations (small q), then $\tilde{L}$ could not be too small. In that case one has to give up on the opportunity of having large stepsizes. The fixed stepsize strategy is the extreme case of minimizing q by giving up the opportunity of having larger stepsizes.

The proposed stepsize strategy PUG tries to reduce $\tilde{L}$ as (iii), but guides $\tilde{L}$ to increase reasonably and quickly when it fails to satisfy the local condition. In particular, by replacing L with its probabilistic upper bound, aggressive $\tilde{L}$ and fast recovering rate are allowed without slowing the convergence. This above feature does not obey the trade-off that constrains choice (i)-(iii). Also, PUG is flexible compared to (iv). It can be applied to all algorithms that require L, as well as mini-batch and block-coordinate-type algorithms which require submatrix of $\mathbf{A}$.

4.4.2 PUG

In this section, we will use the tail bounds to develop PUG. Using equation (4.14), we first define the upper bound at different confidence level,

$$L \leq \mathcal{U}(\epsilon) \triangleq 2\gamma\mu_{\max} - \frac{\gamma R}{m}\log\left(\frac{\epsilon}{n}\right), \tag{4.24}$$

with probability of at least $1-\epsilon$. We point out that the probabilistic upper bound (4.14) does not rely on the assumption that the dataset is normalized with mean zero and unit variance, and so it is applicable to all types of datasets. The basic idea of PUG is to use the result in the following theorem.

Theorem 4.13 *Suppose $\tilde{L}$ is used as an initial guess for the k^{th} iteration, and we denote*

$$\eta_{PUG,N} = \left(\frac{\mathcal{U}(\epsilon)}{\tilde{L}}\right)^{1/N},$$

where $\mathcal{U}(\epsilon)$ is defined as in (4.24). If we select the smallest $q \in \mathbb{N}$ such that $\tilde{L}_k = \eta_{PUG,N}^q \tilde{L}$ satisfies the local condition, then with probability of at least $1-\epsilon$, it requires $q = N$ to guarantee

Algorithm 4.1 PUG

Input: $\tilde{L}_k$ from last iteration
Initialization: Set $\tilde{L} = \tilde{L}_k/2$, $\epsilon = \min\{0.1, \epsilon_0\}$ (Require: ϵ_0 small enough such that $\mathcal{U}(\epsilon) > \tilde{L}$)
Set $\eta_{\text{PUG}} = \sqrt{\mathcal{U}(\epsilon)/\tilde{L}}$
while $\tilde{L}$ does not satisfy Lipschitz constant condition locally **do**
  Set $\tilde{L} = \eta_{\text{PUG}}\tilde{L}$
end while
Output: Lipschitz constant $\tilde{L}_{k+1} = \tilde{L}$

convergence. In particular, we have

$$L \leq \tilde{L}_k \leq \mathcal{U}(\epsilon), \qquad \textit{if} \quad \tilde{L} \leq L,$$
$$L \leq \tilde{L}_k = \tilde{L}, \qquad \textit{if} \quad L \leq \tilde{L},$$

with probability of at least $1 - \epsilon$.

Proof To guarantee convergence, it requires q such that $\tilde{L}_k = \eta^q_{\text{PUG},N}\tilde{L} \geq L$. When $q = N$, $\tilde{L}_k = \mathcal{U}(\epsilon) \geq L$ with probability of at least $1 - \epsilon$. ∎

Theorem 4.13 shows the potential advantage of PUG. With any initial guess $\tilde{L}$, PUG is able to scale $\tilde{L}$ quickly towards L without interfering with the probabilistic convergence rate. This unique feature allows an aggressive initial guess $\tilde{L}$ as Nesterov's adaptive strategy without low recovering rate nor slow convergence rate. Algorithm 4.1 provided details of PUG with $N = 2$, which is chosen in order to be comparable with the Nesterov's adaptive method. We point out that,

$$\mathcal{U}(\epsilon) \to \infty \quad \text{as} \quad \epsilon \to 0.$$

Therefore, the convergence of FISTA is guaranteed with PUG, even in the extreme case that $L \leq \mathcal{U}(\epsilon)$ with $\epsilon \approx 0$.

In the case where computing $\mu_{\max}$ is impractical, it could be bounded by

$$\mu_{\max} = \lambda_{\max}(\mathbf{\Sigma}) = \left\|\mathbb{E}[\boldsymbol{\xi}\boldsymbol{\xi}^T]\right\| \leq \mathbb{E}\left[\left\|\boldsymbol{\xi}\boldsymbol{\xi}^T\right\|\right] = \mathbb{E}\left[\boldsymbol{\xi}^T\boldsymbol{\xi}\right] = \sum_{i=1}^{n}\mathbb{E}\left[\xi_i^2\right] = \sum_{i=1}^{n}\left(\mathrm{Var}(\xi_i) + (\mathbb{E}[\xi_i])^2\right). \tag{4.25}$$

With the assumption that ξ_i's have zero mean and unit variance, $\mu_{\max} \leq n$. For $\mathbf{A}$ that does not satisfy these assumptions due to different normalization process of the data, (4.25) could be used to bound $\mu_{\max}$. For the R in (4.24), one could use $c^2 n$ as in Corollary 4.9, or $\max_i \mathbf{a}_i^T\mathbf{a}_i$ since $\lambda_{\max}\left[\boldsymbol{\xi}\boldsymbol{\xi}^T\right] = \|\boldsymbol{\xi}\boldsymbol{\xi}^T\| = \boldsymbol{\xi}^T\boldsymbol{\xi}$.

4.4.3 Convergence Bounds: Regular Strategies vs. PUG

Different stepsize strategies would lead to different convergence rates even for the same algorithm. Since PUG is based on the probabilistic upper bound $\mathcal{U}(\epsilon)$ in (4.24), it leads to a probabilistic convergence of FISTA.

Theorem 4.14 *Consider the composite program (4.2) with f that is in the form of (4.4), where $\mathbf{A}$ is an $m \times n$ random matrix in which its rows are independent samples of some random variables $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$ with covariance matrix $\mathbf{\Sigma}$. Suppose $|\xi_i| \leq c$ almost surely for $i = 1, 2, \cdots, n$ and $\|\mathbf{x}_0 - \mathbf{x}_\star\|^2 \leq M$ for some positive constant M. Then the convergence of FISTA with PUG (Algorithm 4.1) is*

$$F(\mathbf{x}_k) - F(\mathbf{x}_\star) \leq \frac{2M}{(k+1)^2}\left(2\gamma\mu_{\max} - \frac{\gamma R}{m}\log\left(\frac{\epsilon}{n}\right)\right), \tag{4.26}$$

with probability at least $1 - \epsilon$, where $\mu_{\max} = \lambda_{\max}(\mathbf{\Sigma})$.

Proof This result can be obtained by combining results in Proposition 4.1, Lemma 4.7, and the same argument as in the proof of convergence in [BT09]. ∎

When using regular stepsize strategies, FISTA results in convergence rates that are in the form of (4.20) with different η's ($\eta > 1$). For a backtracking strategy, η would be an user-specified

parameter. It is clear from (4.20) that convergence is better when η is close to 1. However, it would take more iterations and more function evaluations to find a satisfying stepsize, and these costs are not captured in (4.20). In the case of Nesterov's adaptive strategy [Nes15], $\eta = 2$. Using the same analysis as in Section 4.3.2, L should be replaced with the upper bound in (4.21) for the average case, or $\mathcal{U}(\epsilon)$ in (4.24) for the probabilistic case. For the probabilistic case, those convergences are of the same order as in the case of using PUG, as shown in (4.26).

Therefore, PUG is competitive compared to other stepsize strategies in the probabilistic case. The strength of PUG comes from the fact that it is adaptive with strong theoretical guarantee that with high probability, $\tilde{L}_k$ will quickly be accepted at each iteration.

4.4.4 Mini-batch Algorithms and Block-coordinate Algorithms

For mini-batch algorithms, each iteration is performed using only a subset of the whole training set. Therefore, in each iteration, we consider a matrix that contains the corresponding subset. This matrix is a submatrix of $\mathbf{A}$ with the same structure, and therefore it is also a random matrix with smaller size $\bar{m}$-by-n, where $\bar{m} < m$. Using the existing results, we can conclude that the associated $\mathcal{U}(\epsilon)$ in each iteration would be larger than those in full-batch algorithms. As a result, the guaranteed stepsize for mini-batch algorithms tends to be smaller than full-batch algorithms.

On the other hand, block-coordinate algorithms do not update all dimensions at once in each iteration. Rather, a subset of dimensions will be selected to perform the update. In such a setting, we only consider the variables (columns of $\mathbf{A}$) that are associated with the selected coordinates. We should consider a submatrix that is formed by columns of $\mathbf{A}$. This submatrix itself is also a random matrix with smaller size m-by-$\bar{n}$, where $\bar{n} < n$. Using the existing results, the guaranteed stepsize for block-coordinate algorithms tends to be larger.

Thus, with minor modifications PUG can be applied to mini-batch and block-coordinate algorithms.

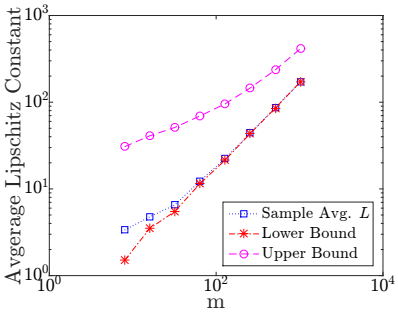


Figure 4.1: Case I, $m = n$

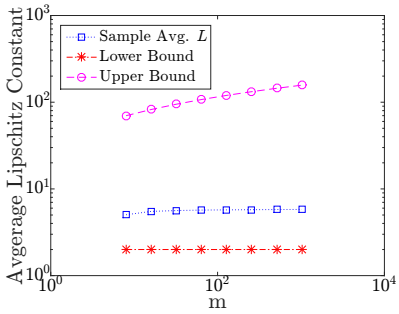


Figure 4.2: Case II, $2m = n$

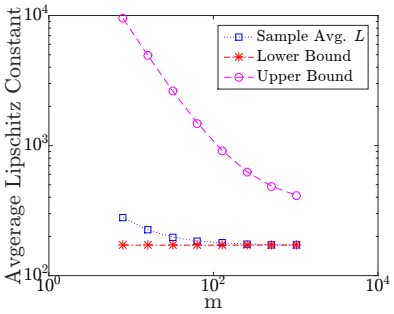


Figure 4.3: Case III, $n = 1024$

4.5 Numerical Experiments

In the first part of this section, we will apply the bounds from Section 4.3 to illustrate the relationship between different parameters and L. Then, we will perform the PUG on two regression examples.

4.5.1 Numerical Simulations for Average L

We consider three cases, and in each case we simulate $\mathbf{A}$'s in different dimension m's and n's. Each configuration is simulated with 1000 instances, and we study the sample average behaviors of L.

In the first case, we consider the most complicated situation and create random vector such that its entries are not identical nor independent. We use a mixture of three types of random variables (exponential, uniform, and multivariate normal) to construct the matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$. The rows of $\mathbf{A}$ are independent samples of $\boldsymbol{\xi}^T = (\xi_1, \xi_2, \cdots, \xi_n)$. We divide $\mathbf{A}$ into three parts with n_1, n_2, and n_3 columns. Note that $n_1 = n_2 = n_3 = n/3$ up to rounding errors. We assign $\boldsymbol{\xi}$ with the elements where

$$\xi_j \sim \begin{cases} Exp(1) - 1 & \text{if } j \le n_1, \\ \mathcal{U}(-\sqrt{3}, \sqrt{3}) & \text{if } n_1 < j \le n_1 + n_2, \end{cases} \tag{4.27}$$

and $(\xi_{n_1+n_2+1}, \xi_{n_1+n_2+2}, \cdots, \xi_n) \sim \mathcal{N}(\mathbf{0}_{n_3\times 1}, \hat{\boldsymbol{\Sigma}})$. $\hat{\boldsymbol{\Sigma}}$ is a $n_3 \times n_3$ matrix with 1 on the diagonal

and 0.5 otherwise. $\xi_1, \xi_2, \cdots, \xi_{n_1+n_2}$ are independent.

The scaling factors of the uniform distribution and exponential distribution are used to normalize the uniform random variables ξ_j such that $\mathbb{E}[\xi_j] = 0$, and $\mathbb{E}[\xi_j^2] = 1$. Some entries of $\mathbf{A}$ are normally distributed or exponentially distributed, and we approximate the upper bound of the entries with $c = 3$. From statistics, we know that with very high probability, this approximation is valid.

In Figure 4.1, we plot the sample average Lipschitz constant over 1000 instances. As expected, the expected Lipschitz constant is "trapped" between its lower and upper bound. We can see that the expected L increases when m and n increases with the ratio n/m is fixed. This phenomenon is due the fact that $\mu_{\max} = \lambda_{\max}(\mathbf{\Sigma})$ increases as n increases.

To further illustrate this, we consider the second case. The setting in this case is the same as the first case except that we replace $\hat{\mathbf{\Sigma}}$ with $\mathbf{I}_n$. So, all the variables are linearly independent. In the case, $\mu_{\max} = 1$ regardless the size of the $\mathbf{A}$. The ratio $n/m = 2$ is fixed in this example. From Figure 4.2, the sample average L does not increase rapidly as the size of $\mathbf{A}$ increases. These results match with the bound (4.21).

In the last case, we investigate the effect of the ratio n/m. The setting is same as the first case, but we keep $n = 1024$ and experiment with different m's. From Figure 4.3, the sample average L decreases as m increases. This result suggests that a large dataset is favorable in terms of complexity, especially for large-scale (large n) ERM problems.

4.5.2 Regularized Logistics Regression

We implement FISTA with three different stepsize strategies (i) the regular backtracking stepsize strategy, (ii) the Nesterov's adaptive stepsize strategy, and (iii) the proposed adaptive stepsize strategy PUG. We compare the three strategies with an example in a ℓ_1 regularized logistic regression problem, in which we solve the convex optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{m}\sum_{i=1}^{m} \log(1+\exp(-b_i\mathbf{x}^T\mathbf{a}_i)) + \omega\|\mathbf{x}\|_1.$$

	Backtracking	Nesterov	PUG
T	1.00x	0.31x	**0.28x**
nIter	1.00x	**0.21x**	0.25x
nFunEva	1.00x	0.28x	**0.27x**
Avg. $\tilde{L}$	1.00x	**0.16x**	0.24x

Table 4.1: Gisette

	Backtracking	Nesterov	PUG
T	1.00x	1.04x	**0.78x**
nIter	1.00x	0.69x	**0.61x**
nFunEva	1.00x	0.92x	**0.71x**
Avg. $\tilde{L}$	1.00x	**0.54x**	0.68x

Table 4.2: YearPredictionMSDt

We use the dataset *gisette* for $\mathbf{A}$ and $\mathbf{b}$. *Gisette* is a handwritten digits dataset from the NIPS 2003 feature selection challenge. The matrix $\mathbf{A}$ is a 6000×5000 dense matrix, and so we have $n = 5000$ and $m = 6000$. The parameter ω is chosen to be the same as [LSS14, YHL12]. *Gisette* can be found at *https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets*. We chose $\tilde{L}_0 = 1$ for all three stepsize strategies. For the backtracking stepsize strategy, we chose $\eta = 1.5$.

Table 4.1 shows the performances of the three stepsize strategies. T is the scaled computational time, nIter is the scaled number of iterations, nFunEva is the scaled number of function evaluations, and Avg. $\tilde{L}$ is the average of $\tilde{L}$ used. This result encourages the two adaptive stepsize strategies as the number of iterations needed and the computational time are significantly smaller compared to the regular backtracking algorithm. This is due to the fact that $\tilde{L}$ could be a lot smaller than the Lipschitz constant L in this example, and so the two adaptive strategies provide more efficient update for FISTA. As shown in Table 4.1, even though Nesterov's strategy yields smaller numbers of iterations, it leads to higher numbers of function evaluations and so it takes more time than PUG.

4.5.3 Regularized Linear Regression

We also compare the three strategies with an example in a ℓ_1 regularized linear regression problem, a.k.a LASSO, in which we solve the convex optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2m}\sum_{i=1}^{m}(\mathbf{x}^T\mathbf{a}_i - b_i)^2 + \omega\|\mathbf{x}\|_1.$$

We use the dataset *YearPredictionMSDt* (testing dataset) for $\mathbf{A}$ and $\mathbf{b}$. *YearPredictionMSDt* has a 51630×90 dense matrix $\mathbf{A}$, and so we have $n = 90$ and $m = 51630$. The parameter ω is chosen to be 10^{-6}. *YearPredictionMSD* can be found at

https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets.

We chose $\tilde{L}_0 = 1$ for all three stepsize strategies. For the backtracking stepsize strategy, we chose $\eta = 1.5$.

Table 4.2 shows the performance of the three stepsize strategies, and the structure is same as Table 4.1. Unlike *Gisette*, adaptive strategies failed to provide small $\tilde{L}$ compared to L. Nesterov's strategy could not take the advantage of its adaptive feature. In particular, compared to backtracking strategy, even though Nesterov's strategy yielded a 31% reduction in terms of number of iterations, the number of function evaluations is only 8% better than the backtracking strategy. This explains the reason why Nesterov's strategy did not outperform the backtracking strategy in this example. PUG, on the other hand, maintains good performance due to the fact that it requires fewer numbers of iterations.

4.6 Conclusions and Perspectives

The analytical results in this chapter show the relationship between the Lipschitz constant and the training set of an ERM problem. These results provide insightful information about the complexity of ERM problems, as well as opening up opportunities for new stepsize strategies for optimization problems.

One interesting extension could be to apply the same approach to different machine learning models, such as neural networks, deep learning, etc.

Chapter 5

Multilevel Methods for Unconstrained Convex Optimization

Life is like riding a bicycle. To keep your balance you must keep moving.

Albert Einstein

Building upon multigrid methods, the framework of multilevel optimization methods was developed to solve structured optimization problems, including problems in optimal control [GMS+10], image processing [PLRR], etc. In this chapter, we give a broader view of the multilevel framework and establish some connections between multilevel algorithms and the other approaches. An interesting case of the so called Galerkin model is further studied. By studying three different case studies of the Galerkin model, we take the first step to show how the structure of optimization problems could improve the convergence of multilevel algorithms.

5.1 Introduction

Multigrid methods are considered as the standard approach in solving differential equations [BHM00, Hac03, HYB17, Str07, TOS01, Wes92]. When solving a differential equation using

numerical methods, an approximation of the solution is obtained on a mesh via discretization. The computational cost of solving the discretized problem, however, varies and it depends on the choice of the mesh size used. Therefore, by considering different mesh sizes, a hierarchy of discretized models can be defined. In general, a more accurate solution can be obtained with a smaller mesh size chosen, which results in a discretized problem in higher dimensions. We shall follow the traditional terminologies in the multigrid community and call a *fine model* to be the discretization in which its solution is sufficiently close to the solution of the original differential equation; otherwise we call it *coarse model* [BHM00]. The main idea of multigrid methods is to make use of the geometric similarity between different discretizations. In particular, during the iterative process of computing a solution of the fine model, one replaces part of the computations with the information from coarse models. The advantages of using multigrid methods are twofold. Firstly, coarse models are in the lower dimensions compared to the fine model, and so the computational cost is reduced. Secondly and interestingly, the directions generated by coarse model and fine model are in fact complementary. It has been shown that using the fine model is effective in reducing the high frequency components of the residual (error) but ineffective in reducing and alternating the low frequency components. Those low frequency components, however, will become high frequency after dimensional reduction. Thus, they could be eliminated effectively using coarse models [BHM00, Str07].

This idea of multigrid was extended to optimization. Nash [Nas00] proposed a multigrid framework for unconstrained infinite-dimensional convex optimization problems. Examples of such problems could be found in the area of optimal control. Following the idea of Nash, many multigrid optimization methods were further developed [Nas00, Nas14, LN13, LN05, KM16, WG09, GST08]. In particular, Wen and Goldfarb [WG09] provided a line search-based multigrid optimization algorithm under the framework in [Nas00], and further extended the framework to nonconvex problems. Gratton et al. [GST08] provided a sophisticated trust-region version of multigrid optimization algorithms, in which they called it multiscale algorithm, and in the later developments [WG09], the name multilevel algorithm is used. In this chapter, we will consistently use the name *multilevel algorithms* for all these optimization algorithms, but we emphasize that the terms multilevel, multigrid, and multiscale were used interchangeably in

different literatures. On the other hand, we keep the name *multigrid methods* for the conventional multigrid methods that solve linear or nonlinear equations that are discretizations arising from partial differential equations (PDEs).

It is worth mentioning that different multilevel algorithms were developed beyond infinite-dimensional problems, see for example Markov decision processes [HP14] (Chapter 3 in the thesis), image deblurring [PLRR], and face recognition [HPZ15]. The above algorithms all have the same aim: to speed up the computations by making use of the geometric similarity between different models in the hierarchy.

The numerical performance of multilevel algorithms has been satisfying. In particular, both of the line-search based [WG09] and trust-region based [GMS$^+$10] algorithms outperform standard methods when solving infinite-dimensional problems. Numerical results show that multilevel algorithms can take the advantage of the geometric similarity between different discretizations just as the original multigrid methods.

However, to the best of our knowledge, no theoretical result is able to show the advantages of using multilevel optimization algorithms. For the line-search based algorithm, Wen and Goldfarb [WG09] proved a sublinear convergence rate for strongly convex problems and convergence for nonconvex problems. Gratton et al. [GST08] proved that their trust-region based multilevel algorithm requires the same order of number of iterations as compared to the gradient descent.

Building upon the above developments, in this chapter, we aim to address three fundamental issues with the current multilevel optimization framework. Firstly, under the general framework of multilevel optimization, could we connect classical optimization algorithms with the recently developed multilevel optimization algorithms? Secondly, could we extend the current analysis and explain why multilevel optimization algorithms outperform standard methods for some classes of problems (e.g. infinite-dimensional problems)? Thirdly, how do we construct a coarse model when the hierarchy is not obvious?

The contributions of this chapter are:

- We provide a more complete view of line search multilevel algorithms, and in particular,

we connect the general framework of the multilevel algorithm with classical optimization algorithms, such as variable metric methods and block-coordinate type methods. We also make a connection with the stochastic variance reduced gradient (SVRG) algorithm [JZ13].

- We analyze the multilevel algorithm with the Galerkin model. The key feature of the Galerkin model is that a coarse model is created from the first and second order information of the fine model. The name "Galerkin model" is given in [GST08] since this is related to the Galerkin approximation in algebraic multigrid methods [Stü01]. We will call this algorithm the **G**alerkin-based **A**lgebraic **M**ultilevel **A**lgorithm (**GAMA**). A global convergence analysis of GAMA is provided.

- We propose to use the composite rate for analysis of the local convergence of GAMA. As we will show later, neither linear convergence nor quadratic convergence is suitable when studying the local convergence due to the broadness of GAMA.

- We study the composite rate of GAMA in a case study of infinite dimensional optimization problems. We show that the linear component of the composite rate is inversely proportional to the smoothness of the residual, which agrees with the findings in conventional multigrid methods.

- We show that GAMA can be set up as Newton's method in lower dimensions with low rank approximation to Hessians. This is done by a low rank approximation method called the naïve Nyström method. We show how the dimensions of the coarse model and the spectrum of the eigenvalues would affect the composite rate.

- GAMA can also be set up as Newton's method with block-diagonal approximation of the Hessians. We define a class of objective functions with weakly-connected Hessians. That is, the Hessians of the function have the form of a linear combination of a block-diagonal matrix and a general matrix which its entries are in $\mathcal{O}(\delta)$, for $\delta \ll 1$. We show how δ would vary the composite rate, and at the limit $\delta \to 0$, GAMA would achieve the quadratic rate of convergence.

The rest of this chapter is structured as follows: In Section 5.2 we provide background material and introduce different variants of multilevel algorithms. We also show that several existing optimization algorithms are in fact special cases under the general framework of multilevel algorithms. In Section 5.3, we study the convergence of GAMA. We first derive the global convergence rate of GAMA, and then show that GAMA exhibits composite convergence when the current incumbent is sufficiently close to the optimum. Composite convergence rate is defined as a linear combination of linear convergence and quadratic convergence, and we denote r_1 and r_2 as the coefficient of linear rate and quadratic rate, respectively. Using these results, in Section 5.4 we derive the complexity of both GAMA and Newton's method. When r_1 is sufficiently small, we show that GAMA has less complexity compared to Newton's method. In Sections 5.5-5.7, three special cases of GAMA are considered. We compute r_1 in each case and show the relationship between r_1 and the structure of the problem. In Section 5.5, we study problems arising from discretizations of one-dimensional PDE problems; in Section 5.6 we study problems where the low rank approximations of Hessians are sufficiently accurate; in Section 5.7 we study problems where the Hessians of the objective function are nearly block-diagonal. In Section 5.8 we illustrate the convergence of GAMA using several numerical examples, including variational optimization problems and machine learning problems.

5.2 Multilevel Models

In this section a broad view of the general multilevel framework will be provided. We start with basic settings and the core idea of multilevel algorithms in [GST08, LN05, WG09], then we show that the general multilevel framework covers several optimization algorithms, including the variable metric methods, block-coordinate descent, and stochastic variance reduced gradient. At the end of this section we provide the settings and details of the core topic of this chapter - Galerkin model.

5.2.1 Basic Settings

In this chapter we are interested in solving,

$$\min_{\mathbf{x}_h \in \mathbb{R}^N} f_h(\mathbf{x}_h), \tag{5.1}$$

where $\mathbf{x}_h \in \mathbb{R}^N$, and function $f_h : \mathbb{R}^N \to \mathbb{R}$ is continuous, differentiable, and strongly convex.

We first clarify the use of the subscript h. Throughout this chapter, the lower case h represents that this is associated with the ***fine*** (exact) model. To use multilevel methods, one needs to formulate a hierarchy of models, and models with lower dimensions (resolutions) called the ***coarse*** models. To avoid the unnecessary complications, in this chapter we consider only two models in the hierarchy: fine and coarse. In the same manner of using subscript h, we assign the upper case H to represent the association with coarse model. We assign N and n ($n \leq N$) to be the dimensions of the fine model and the coarse model, respectively. For instance, any vector that is within the space $\mathbb{R}^N$ is denoted with subscript h, and similarly, any vector with subscript H is within the space $\mathbb{R}^n$.

Assumption 5.1 *There exists constants μ_h, L_h, and M_h such that*

$$\mu_h \mathbf{I} \preccurlyeq \nabla^2 f_h(\mathbf{x}_h) \preccurlyeq L_h \mathbf{I}, \quad \forall \mathbf{x}_h \in \mathbb{R}^N, \tag{5.2}$$

and

$$\|\nabla^2 f_h(\mathbf{x}_h) - \nabla^2 f_h(\mathbf{y}_h)\| \leq M_h \|\mathbf{x}_h - \mathbf{y}_h\|, \quad \forall \mathbf{x}_h, \mathbf{y}_h \in \mathbb{R}^N. \tag{5.3}$$

Equation (5.2) implies

$$\|\nabla f_h(\mathbf{x}_h) - \nabla f_h(\mathbf{y}_h)\| \leq L_h \|\mathbf{x}_h - \mathbf{y}_h\|, \quad \forall \mathbf{x}_h, \mathbf{y}_h \in \mathbb{R}^N.$$

The above assumption of the objective function will be used throughout this chapter, and it is common when studying second order algorithms.

Multilevel methods require mapping information across different dimensions. To this end, we define a matrix $\mathbf{P} \in \mathbb{R}^{N\times n}$ to be the prolongation operator which maps information from coarse to fine, and we define a matrix $\mathbf{R} \in \mathbb{R}^{n\times N}$ to be the restriction operator which maps information from fine to coarse. We make the following assumption on $\mathbf{P}$ and $\mathbf{R}$.

Assumption 5.2 *The restriction operator* $\mathbf{R}$ *is the transpose of the prolongation operator* $\mathbf{P}$ *up to a constant* c*. That is,*

$$\mathbf{P} = c\mathbf{R}^T, \qquad c > 0.$$

Without loss of generality, we take $c = 1$ throughout this chapter to simplify the use of notation for the analysis. We also assume any useful (non-zero) information in the coarse model will not become zero after prolongation and make the following assumption.

Assumption 5.3 *The prolongation operator* $\mathbf{P}$ *has full column rank, and so*

$$\mathit{rank}(\mathbf{P}) = n.$$

Notice that Assumption 5.2 and 5.3 are standard assumptions for multilevel methods [BHM00, Hac03, WG09]. Since $\mathbf{P}$ has full column rank, we define the pseudoinverse and its norm

$$\mathbf{P}^+ = (\mathbf{R}\mathbf{P})^{-1}\mathbf{R}, \quad \text{and} \quad \nu_L = \|\mathbf{P}^+\|. \tag{5.4}$$

The coarse model is constructed in the following manner. Suppose in the k^{th} iterations we have an incumbent solution $\mathbf{x}_{h,k}$ and gradient $\nabla f_{h,k} \triangleq \nabla f_h(\mathbf{x}_{h,k})$. Then the corresponding coarse model is,

$$\min_{\mathbf{x}_H\in\mathbb{R}^n} \phi_H(\mathbf{x}_H) \triangleq f_H(\mathbf{x}_H) + \langle \mathbf{v}_H, \mathbf{x}_H - \mathbf{x}_{H,0}\rangle, \tag{5.5}$$

where,

$$\mathbf{v}_H \triangleq -\nabla f_{H,0} + \mathbf{R}\nabla f_{h,k},$$

$\mathbf{x}_{H,0} = \mathbf{R}\mathbf{x}_{h,k}$, and $f_H : \mathbb{R}^n \to \mathbb{R}$. Similar to $\nabla f_{h,k}$, we denote $\nabla f_{H,0} \triangleq \nabla f_H(\mathbf{x}_{H,0})$ and

$\nabla\phi_{H,0} \triangleq \nabla\phi_H(\mathbf{x}_{H,0})$ to simplify notation. Similar notation will be used consistently unless it is specified otherwise. We emphasize the construction of coarse model (5.5) is common in the line of multilevel optimization research and it is not original in this chapter. See for example [GST08, LN05, WG09]. Note that when constructing the coarse model (5.5), one needs to add an additional linear term on $f_H(\mathbf{x}_H)$. This linear term ensures the following is satisfied,

$$\nabla\phi_{H,0} = \mathbf{R}\nabla f_{h,k}. \tag{5.6}$$

For infinite-dimensional optimization problems, one can define f_h and f_H using discretization with different mesh sizes. In general, f_h is the function that is sufficiently close to the original problem, and that can be achieved using small mesh sizes. Based on geometric similarity between discretizations with different meshes, $f_h \approx f_H$ even though $n \leq N$.

However, we want to emphasize $f_h \approx f_H$ is not a necessary requirement when using multilevel methods. In principle, $f_H(\mathbf{x}_H)$ can be any function. Galerkin model, as we will show later, is a quadratic model where f_H is chosen to be an approximation of the Hessian of f_h.

5.2.2 The General Multilevel Algorithm

The main idea of multilevel algorithms is to use the coarse model to compute search directions. We call such direction the ***coarse correction step***. When using coarse correction steps, we compute the direction by solving the corresponding coarse model (5.5) and perform the update

$$\mathbf{x}_{h,k+1} = \mathbf{x}_{h,k} + \alpha_{h,k}\hat{\mathbf{d}}_{h,k},$$

with

$$\hat{\mathbf{d}}_{h,k} \triangleq \mathbf{P}(\mathbf{x}_{H,\star} - \mathbf{x}_{H,0}), \tag{5.7}$$

where $\mathbf{x}_{H,\star}$ is the solution of the coarse model, and $\alpha_{h,k} \in \mathbb{R}^+$ is the stepsize. We clarify that the "hat" in $\hat{\mathbf{d}}_{h,k}$ is used to identify a coarse correction step. The subscript h in $\hat{\mathbf{d}}_{h,k}$ is used because $\hat{\mathbf{d}}_{h,k} \in \mathbb{R}^N$.

We should emphasize that $\mathbf{x}_{H,\star}$ in (5.7) can be replaced by $\mathbf{x}_{H,r}$ for $r = 1, 2, \ldots,$ i.e. the incumbent solution of the coarse mode (5.5) after r^{th} iterations. However, for the purpose of this chapter and simplicity, we ignore this case unless there is extra specification, and we let (5.7) be the coarse correction step.

It is known that the coarse correction step $\hat{\mathbf{d}}_{h,k}$ is a descent direction if f_H is convex. The following lemma states this argument rigorously. Even though the proof is provided in [WG09], we provide it with our notation for the completeness of this chapter.

Lemma 5.4 ([WG09]) *If f_H is a convex function, then the coarse correction step is a descent direction. In particular, in the k^{th} iteration,*

$$\nabla f_{h,k}^T \hat{\mathbf{d}}_{h,k} \leq \phi_{H,\star} - \phi_{H,0} \leq 0.$$

Proof

$$\begin{aligned} \nabla f_{h,k}^T \hat{\mathbf{d}}_{h,k} &= \nabla f_{h,k}^T \mathbf{R}^T (\mathbf{x}_{H,\star} - \mathbf{x}_{H,0}) \\ &= (\mathbf{R}\nabla f_{h,k})^T (\mathbf{x}_{H,\star} - \mathbf{x}_{H,0}) \\ &= \nabla \phi_{H,0}^T (\mathbf{x}_{H,\star} - \mathbf{x}_{H,0}) \\ &\leq \phi_{H,\star} - \phi_{H,0}, \end{aligned}$$

as required. ∎

The last inequality holds because ϕ_H is a convex function. Even though Lemma 5.4 states that $\hat{\mathbf{d}}_{h,k}$ is a descent direction, using coarse correction step solely is not sufficient to solve the fine model (5.1).

Proposition 5.5 *Suppose $\nabla f_{h,k} \neq 0$ and $\nabla f_{h,k} \in null(\mathbf{R})$, then the coarse correction step*

$$\hat{\mathbf{d}}_{h,k} = 0.$$

Proof From (5.6), $\mathbf{x}_{H,\star} = \mathbf{x}_{H,0}$ when $\mathbf{R}\nabla f_{h,k} = 0$. Thus, $\hat{\mathbf{d}}_{h,k} = \mathbf{P}(\mathbf{x}_{H,\star} - \mathbf{x}_{H,0}) = 0$. ∎

Recall that $\mathbf{R} \in \mathbb{R}^{n\times N}$, and so for $n < N$, a coarse correction step could be zero and make no progress even when the first order necessary condition $\nabla f_h = 0$ has not been satisfied.

Fine Correction Step

Two approaches can be used when the coarse correction step is not progressing nor effective. The first approach is to compute directions using standard optimization methods. We call such step the ***fine correction step***. As opposed to coarse correction step $\hat{\mathbf{d}}_{h,k}$, we abandon the use of "hat" for all fine correction steps and denote them as $\mathbf{d}_{h,k}$'s.

Classical examples of $\mathbf{d}_{h,k}$'s are steps that are computed by standard methods such as gradient descent method, quasi-Newton method, etc. We perform fine correction steps when coarse correction steps are not effective. That is,

$$\|\mathbf{R}\nabla f_{h,k}\| < \kappa\|\nabla f_{h,k}\| \quad \text{or} \quad \|\mathbf{R}\nabla f_{h,k}\| < \epsilon, \tag{5.8}$$

where $\kappa \in (0, \min(1, \|\mathbf{R}\|))$, and $\epsilon \in (0, 1)$. The above criteria prevent using coarse model when $\mathbf{x}_{H,0} \approx \mathbf{x}_{H,\star}$, i.e. the coarse correction step $\hat{\mathbf{d}}_{h,k}$ is close to $\mathbf{0}$. We point out that these criteria were also proposed in [WG09]. We also make the following assumption on the fine correction step throughout this chapter.

Assumption 5.6 *There exists strictly positive constants $\zeta_1, \zeta_2 > 0$ such that*

$$\|\mathbf{d}_{h,k}\| \le \zeta_1\|\nabla f_{h,k}\|, \quad \textit{and} \quad -\nabla f_{h,k}^T\mathbf{d}_{h,k} \ge \zeta_2\|\nabla f_{h,k}\|^2,$$

where $\mathbf{d}_{h,k}$ is a fine correction step. As a consequence, there exists a constant $\Lambda_h > 0$ such that

$$f_{h,k} - f_{h,k+1} \ge \Lambda_h\|\nabla f_{h,k}\|^2,$$

where $f_{h,k+1}$ is updated using a fine correction step.

As we will show later, Assumption 5.6 is not restrictive, and Λ_h is known for well-known cases

like gradient descent, Newton method, etc. Using the combination of fine and coarse correction steps is the standard approach in multilevel methods, especially for PDE-based optimization problems [GST08, LN05, WG09].

Multiple P's and R's

The second approach to overcome issue of ineffective coarse correction step is by creating multiple coarse models with different $\mathbf{P}$'s and $\mathbf{R}$'s.

Proposition 5.7 *Suppose $\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_p$ are all restriction operators that satisfy Assumption 5.2 and 5.3, where $\mathbf{R}_i \in \mathbb{R}^{n_i \times N}$ for $i = 1, 2, \ldots, p$. Denote $\mathcal{S}$ to be a set that contains the rows of $\mathbf{R}_i$'s in $\mathbb{R}^N$, for $i = 1, 2, \ldots, p$. If*

$$span(\mathcal{S}) = \mathbb{R}^N,$$

then for $\nabla f_{h,k} \neq 0$ there exists at least one $\mathbf{R}_j \in \{\mathbf{R}_i\}_{i=1}^p$ such that

$$\hat{\mathbf{d}}_{h,k} \neq 0 \quad \textit{and} \quad \nabla f_{h,k}^T \hat{\mathbf{d}}_{h,k} < 0,$$

where $\hat{\mathbf{d}}_{h,k}$ is the coarse correction step computed using $\mathbf{R}_j$.

Proof Since $\text{span}(\mathcal{S}) = \mathbb{R}^N$, then for $\nabla f_{h,k} \neq 0$, there exists one $\mathbf{R}_j$ such that $\mathbf{R}_j \nabla f_{h,k} \neq 0$. So the corresponding coarse model would have $\mathbf{x}_{H,\star} \neq \mathbf{x}_{H,0}$, and thus $\hat{\mathbf{d}}_{h,k_j} \neq 0$. ∎

Proposition 5.7 shows that if the rows of the restriction operators $\mathbf{R}_i$'s span $\mathbb{R}^N$, then at least one coarse correction step from these restriction operators would be nonzero and thus effective. In each iteration, one could use a similar idea as in (5.8) to rule out ineffective coarse models. However, this checking process could be expensive for large scale problems with large p (number of restriction operators). To omit this checking process, one could choose the following alternatives.

i. **Cyclical approach**: choose $\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_p$ in order at each iteration, and choose $\mathbf{R}_1$ after $\mathbf{R}_p$.

ii. **Probabilistic approach**: assign a probability mass function with $\{\mathbf{R}_i\}_{i=1}^p$ as a sample space, and choose the coarse model randomly based on the mass function. The mass function has to be strictly positive for each $\mathbf{R}_i$.

We point out that this idea of using multiple coarse models is related to domain decomposition methods, which solve (non-)linear equations arising from PDEs. Domain decomposition methods partition the problem domain into several sub-domains, and thus decompose the original problem into several smaller problems. We refer the readers to [CS94] for more details about domain decomposition methods.

In Section 5.2.3, we will show that using multiple $\mathbf{P}$'s and $\mathbf{R}$'s is not new in the optimization research community. Using the above multilevel framework, one can re-generate the block-coordinate descent.

5.2.3 Connection with Variable Metric Methods

Using the above multilevel framework, in the rest of this section we will introduce different versions of multilevel algorithms: variable metric methods, block-coordinate descent, and stochastic variance reduced gradient. At the end of this section we will introduce the Galerkin model, which is an interesting case of the multilevel framework.

Recall that for variable metric methods, the direction $\mathbf{d}_{h,k}$ is computed by solving

$$\begin{aligned} \mathbf{d}_{h,k} &= \arg\min_{\mathbf{d}} \frac{1}{2}\langle \mathbf{d}, \mathbf{Q}\mathbf{d}\rangle + \langle \nabla f_{h,k}, \mathbf{d}\rangle, \\ &= -\mathbf{Q}^{-1}\nabla f_{h,k}. \end{aligned} \tag{5.9}$$

where $\mathbf{Q} \in \mathbb{R}^{N\times N}$ is a positive definite matrix. When $\mathbf{Q} = \mathbf{I}$, $\mathbf{d}_{h,k}$ is the gradient descent search direction. When $\mathbf{Q} = \nabla^2 f_{h,k}$, $\mathbf{d}_{h,k}$ is the search direction by Newton's method. When $\mathbf{Q}$ is an approximation of the Hessian, then $\mathbf{d}_{h,k}$ is the quasi-Newton search direction.

To show the connections between multilevel methods and variable metric methods, consider the following f_H.

$$f_H(\mathbf{x}_H) = \frac{1}{2}\langle \mathbf{x}_H - \mathbf{x}_{H,0}, \mathbf{Q}_H(\mathbf{x}_H - \mathbf{x}_{H,0})\rangle, \tag{5.10}$$

where $\mathbf{Q}_H \in \mathbb{R}^{n\times n}$, and $\mathbf{x}_{H,0} = \mathbf{R}\mathbf{x}_{h,k}$ as defined in (5.5). Applying the definition of the coarse model (5.5), we obtain

$$\min_{\mathbf{x}_H\in\mathbb{R}^n} \phi_H(\mathbf{x}_H) = \frac{1}{2}\langle \mathbf{x}_H - \mathbf{x}_{H,0}, \mathbf{Q}_H(\mathbf{x}_H - \mathbf{x}_{H,0})\rangle + \langle \mathbf{R}\nabla f_{h,k}, \mathbf{x}_H - \mathbf{x}_{H,0}\rangle. \tag{5.11}$$

Thus from the definition in (5.7), the associated coarse correction step is,

$$\hat{\mathbf{d}}_{h,k} = \mathbf{P}\left(\arg\min_{\mathbf{d}_H\in\mathbf{R}^n} \underbrace{\frac{1}{2}\langle \mathbf{d}_H, \mathbf{Q}_H\mathbf{d}_H\rangle + \langle \mathbf{R}\nabla f_{h,k}, \mathbf{d}_H\rangle}_{\mathbf{d}_H = \mathbf{x}_H - \mathbf{x}_{H,0}}\right) = -\mathbf{P}\mathbf{Q}_H^{-1}\mathbf{R}\nabla f_{h,k}. \tag{5.12}$$

Therefore, with this specific f_H in (5.10), the resulting coarse model (5.11) is analogous to variable metric methods. In a naïve case where $n = N$ and $\mathbf{P} = \mathbf{R} = \mathbf{I}$, the corresponding coarse correction step (5.12) would be the same as gradient descent direction, Newton direction, and quasi-Newton direction for $\mathbf{Q}_H$ that is identity matrix, Hessian, and approximation of Hessian, respectively.

5.2.4 Connection with Block-coordinate Descent

Interestingly, the coarse model (5.11) is also related to block-coordinate type methods. Suppose we have p coarse models with prolongation and restriction operators, $\{\mathbf{P}_i\}_{i=1}^p$ and $\{\mathbf{R}_i\}_{i=1}^p$, respectively. For each coarse model, we let (5.10) be the corresponding f_H with $\mathbf{Q}_H = \mathbf{I}$, and we further restrict our setting with the following properties.

1. $\mathbf{P}_i \in \mathbb{R}^{N\times n_i}$, $\forall i = 1, 2, \ldots, p$.
2. $\mathbf{P}_i = \mathbf{R}_i^T$, $\forall i = 1, 2, \ldots, p$.
3. $[\mathbf{P}_1\ \mathbf{P}_2 \ldots \mathbf{P}_p] = \mathbf{I}$.

From (5.12), the above setting results in $\hat{\mathbf{d}}_{h,k_i} = -\mathbf{P}_i\mathbf{R}_i\nabla f_{h,k}$, where $\hat{\mathbf{d}}_{h,k_i}$ is the coarse correction step for the i^{th} model. Notice that

$$(\mathbf{P}_i\mathbf{R}_i\nabla f_{h,k})_j = \begin{cases} (\nabla f_{h,k})_j & \text{if } \sum_{q=1}^{i-1} n_q < j \leq \sum_{q=1}^{i} n_q, \\ 0 & \text{otherwise .} \end{cases}$$

Therefore, $\hat{\mathbf{d}}_{h,k_i}$ is equivalent to a block-coordinate descent update [BT13]. When $n_i = 1$, for $i = 1, 2, \ldots, p$, it becomes a coordinate descent method. When $1 < n_i < N$, for $i = 1, 2, \ldots, p$, it becomes a block-coordinate descent. When the $\mathbf{P}_i$'s and $\mathbf{R}_i$'s are chosen using the cyclical approach, then it would be a cyclical (block)-coordinate descent. When the $\mathbf{P}_i$'s and $\mathbf{R}_i$'s are chosen using the probabilistic approach, then it would be a randomized (block)-coordinate descent method.

5.2.5 Connection with SVRG

The multilevel framework is also related to the Stochastic Variance Reduced Gradient (SVRG) and its variants [GGR16, JZ13, MNJ16], which is a state-of-the-art algorithm for structured machine learning problems. Suppose the fine model has the following form

$$\min_{\mathbf{x}_h \in \mathbb{R}^N} f_h(\mathbf{x}_h) = \frac{1}{M}\sum_{i=1}^{M} f_i(\mathbf{x}_h).$$

We denote a set, $\mathcal{S}_H \subseteq \{1, 2, \ldots, M\}$ with $|\mathcal{S}_H| = m$, and construct the following coarse model

$$\min_{\mathbf{x}_H \in \mathbb{R}^N} f_H(\mathbf{x}_H) = \frac{1}{m}\sum_{i \in \mathcal{S}_H} f_i(\mathbf{x}_H).$$

In this particular case where $\mathbf{x}_h, \mathbf{x}_H \in \mathbb{R}^N$, no dimension is reduced, and we let $\mathbf{P} = \mathbf{R} = \mathbf{I}$. In the k^{th} iteration with incumbent $\mathbf{x}_k$, the coarse model is

$$\min_{\mathbf{x}_H \in \mathbb{R}^n} \frac{1}{m}\sum_{i \in \mathcal{S}_H} f_i(\mathbf{x}_H) + \left\langle -\frac{1}{m}\sum_{i \in \mathcal{S}_H} \nabla f_i(\mathbf{x}_{h,k}) + \frac{1}{M}\sum_{i=1}^{M} \nabla f_i(\mathbf{x}_{h,k}), \mathbf{x}_H - \mathbf{x}_{h,k} \right\rangle.$$

Suppose gradient descent is applied for K steps to solve the above coarse model, then

$$\mathbf{x}_{H,j} = \mathbf{x}_{H,j-1} - \alpha_{H,j}\left(\frac{1}{m}\sum_{i\in\mathcal{S}_H}\nabla f_i(\mathbf{x}_{H,j-1}) - \frac{1}{m}\sum_{i\in\mathcal{S}_H}\nabla f_i(\mathbf{x}_{h,k}) + \frac{1}{M}\sum_{i=1}^{M}\nabla f_i(\mathbf{x}_{h,k})\right),$$

for $j = 1, 2, \ldots, K$. The above update is the key step in SVRG and its variants. In particular, when $m = K_d = 1$, the above setting is the same as the original SVRG in [JZ13] with 1 inner iteration. Even though the coarse model is in the same dimension as the fine model, the cost of computing function values and gradients is much cheaper when $m \ll M$.

5.2.6 The Galerkin Model

We end this section with the core topic of this chapter - the Galerkin model. The Galerkin coarse model is a special case of (5.11) where,

$$\mathbf{Q}_H = \nabla_H^2 f_{h,k} \triangleq \mathbf{R}\nabla^2 f_{h,k}\mathbf{P}, \tag{5.13}$$

and so the Galerkin (coarse) model is,

$$\min_{\mathbf{x}_H\in\mathbb{R}^n} \phi_H(\mathbf{x}_H) = \frac{1}{2}\langle \mathbf{x}_H - \mathbf{x}_{H,0}, \nabla_H^2 f_{h,k}(\mathbf{x}_H - \mathbf{x}_{H,0})\rangle + \langle \mathbf{R}\nabla f_{h,k}, \mathbf{x}_H - \mathbf{x}_{H,0}\rangle. \tag{5.14}$$

According to (5.12), the corresponding coarse correction step is

$$\hat{\mathbf{d}}_{h,k} = -\mathbf{P}[\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}]^{-1}\mathbf{R}\nabla f_{h,k} = -\mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla f_{h,k}. \tag{5.15}$$

The Galerkin model is closely related to algebraic multigrid methods which solve (non-)linear equations arising from PDEs. Algebraic multigrid methods are used when the computation or implementation of f_H is difficult (see e.g. [Stü01]). In the context of multilevel optimization, to the best of our knowledge, this is first mentioned in [GST08] by Gratton, Sartenaer, and Toint. In [GST08] a trust-region type multilevel method is proposed to solve PDE-based optimization problems, and the Galerkin model is described as a "radical strategy". In a later paper from Gratton et al. [GMS+10], the trust-region type multilevel method is tested numerically, and

Galerkin model provides good numerical results.

It is worth mentioning that the above coarse correction step is equivalent to the solution of the system of linear equations,

$$\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}\mathbf{d}_H = -\mathbf{R}\nabla f_{h,k}. \tag{5.16}$$

which is the general case of the Newton's method in which $\mathbf{P} = \mathbf{R} = \mathbf{I}$. Using Assumption 5.3, we can show that $\nabla_H^2 f_{h,k}$ is positive definite, and so equation (5.16) has a unique solution.

Proposition 5.8 $\mathbf{R}\nabla^2 f_h(\mathbf{x}_h)\mathbf{P}$ *is positive definite, and in particular,*

$$\mu_h \nu_L^{-2}\mathbf{I} \preceq \mathbf{R}\nabla^2 f_h(\mathbf{x}_h)\mathbf{P} \preceq L_h \nu_U^2 \mathbf{I}$$

where $\nu_U \triangleq \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$ *and* $\nu_L = \|\mathbf{P}^+\|$ *as defined in (5.4).* L_h *and* μ_h *are defined in Assumption 5.1.*

Proof

$$\mathbf{x}^T \left(\mathbf{R}\nabla^2 f_h(\mathbf{x}_h)\mathbf{P}\right)\mathbf{x} = (\mathbf{P}\mathbf{x})^T \nabla^2 f_h(\mathbf{x}_h)(\mathbf{P}\mathbf{x}) \leq L_h \|\mathbf{P}\mathbf{x}\|^2 \leq L_h \nu_U^2 \|\mathbf{x}\|^2.$$

Also,

$$\mathbf{x}^T \left(\mathbf{R}\nabla^2 f_h(\mathbf{x}_h)\mathbf{P}\right)\mathbf{x} = (\mathbf{P}\mathbf{x})^T \nabla^2 f_h(\mathbf{x}_h)(\mathbf{P}\mathbf{x}) \geq \mu_h \|\mathbf{P}\mathbf{x}\|^2 \geq \frac{\mu_h}{\|\mathbf{P}^+\|^2}\|\mathbf{x}\|^2 = \frac{\mu_h}{\nu_L^2}\|\mathbf{x}\|^2.$$

So we obtain the desired result. ∎

5.3 Convergence of GAMA

In this section we will analyze GAMA that is stated as Algorithm 5.1. The fine correction steps in Algorithm 5.1 are deployed by variable metric methods, and an Armijo's rule is used as stepsize strategy for both fine and coarse correction steps. We emphasize that Algorithm

Algorithm 5.1 GAMA

Input: κ, ϵ, $\rho_1 \in (0, 0.5)$, $\beta_{ls} \in (0, 1)$,
$\mathbf{P} \in \mathbb{R}^{N\times n}$ and $\mathbf{R} \in \mathbb{R}^{n\times N}$ which satisfy Assumption 5.2 and 5.3.
Initialization: $\mathbf{x}_{h,0} \in \mathbb{R}^N$
for $k = 0, 1, 2, \ldots$ **do**
Compute the direction

$$\mathbf{d} = \begin{cases} \hat{\mathbf{d}}_{h,k} \text{ in (5.15)} & \text{if } \|\mathbf{R}\nabla f_{h,k}\| > \kappa\|\nabla f_{h,k}\| \text{ and } \|\mathbf{R}\nabla f_{h,k}\| > \epsilon, \\ \mathbf{d}_{h,k} \text{ in (5.9)} & \text{otherwise.} \end{cases}$$

Find the smallest $q \in \mathbb{N}$ such that for stepsize $\alpha_{h,k} = \beta_{ls}^q$,

$$f_h(\mathbf{x}_{h,k} + \alpha_{h,k}\mathbf{d}) \leq f_{h,k} + \rho_1\alpha_{h,k}\nabla^T f_{h,k}\mathbf{d}.$$

Update

$$\mathbf{x}_{h,k+1} \triangleq \mathbf{x}_{h,k} + \alpha_{h,k}\mathbf{d}.$$

end for

5.1 is the basic version of GAMA, but the general techniques of analysis in this section could be applied to its variants which we introduced in Section 5.2. The results in this section will be used in Section 5.3 to compare the complexity between GAMA and Newton's method.

We will first show that Algorithm 5.1 achieves a sublinear rate of convergence. In particular, we will show that for $k = 0, 1, \ldots,$

$$f_{h,k} - f_{h,\star} \leq \frac{C}{2+k},$$

for some constant C which we will specify later in this section. We then analyze the maximum number of coarse correction steps that would be taken by Algorithm 5.1, and the condition under which the coarse correction steps yield quadratic reduction in the gradients in the subspace. At the end of this section, we will provide the composite convergence rate for the coarse correction steps.

To provide convergence properties when coarse correction steps are used, the following quantity will be used:

$$\chi_{H,k} \triangleq [(\mathbf{R}\nabla f_{h,k})^T[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla f_{h,k}]^{1/2}.$$

Notice that $\chi_{H,k}$ is analogous to the Newton decrement, which is used to study the convergence

of Newton's method [BV04]. In particular, the defined $\chi_{H,k}$ has the following properties.

1. $\nabla f_{h,k}^T \hat{\mathbf{d}}_{h,k} = -\chi_{H,k}^2$.
2. $\hat{\mathbf{d}}_{h,k}^T \nabla^2 f_{h,k} \hat{\mathbf{d}}_{h,k} = \chi_{H,k}^2$.

We omit the proofs of the above properies since these can be done by using direct computation and the definition of $\chi_{H,k}$.

5.3.1 The Worst Case $\mathcal{O}(1/k)$ Convergence

We will show that Algorithm 5.1 will achieve a sublinear rate of convergence. We will deploy the techniques from [BT13] and [BV04]. Starting with the following lemma, we state reduction in function value using coarse correction steps. We would like to clarify that even though GAMA is considered as a special case in [WG09], we take advantage of this simplification and specification to provide analysis with results that are easier to interpret. In particular, the analysis of stepsizes $\alpha_{h,k}$'s in [WG09] relies on the maximum number of iterations taken. This result is unfavourable and unnecessary for the settings we consider.

Lemma 5.9 *The coarse correction step $\hat{\mathbf{d}}_{h,k}$ in Algorithm 5.1 will lead to reduction in function value*

$$f_{h,k} - f_h(\mathbf{x}_{h,k} + \alpha_{h,k}\hat{\mathbf{d}}_{h,k}) \geq \frac{\rho_1 \kappa^2 \beta_{ls} \mu_h}{L_h^2} \|\nabla f_{h,k}\|^2,$$

where ρ_1, κ, and β_{ls} are user-defined parameters in Algorithm 5.1. L_h and μ_h are defined in Assumption 5.1.

Proof By mean value theorem,

$$\begin{aligned} f(\mathbf{x}_{h,k} + \alpha\hat{\mathbf{d}}_{h,k}) &\leq f_{h,k} + \alpha\langle \nabla f_{h,k}, \hat{\mathbf{d}}_{h,k}\rangle + \frac{L_h}{2}\alpha^2 \|\hat{\mathbf{d}}_{h,k}\|^2, \\ &\leq f_{h,k} - \alpha\chi_{H,k}^2 + \frac{L_h}{2\mu_h}\alpha^2 \chi_{H,k}^2, \end{aligned}$$

since

$$\mu_h \|\hat{\mathbf{d}}_{h,k}\|^2 \leq \hat{\mathbf{d}}_{h,k}^T \nabla^2 f(x_k) \hat{\mathbf{d}}_{h,k} = \chi_{H,k}^2.$$

Notice that $\hat{\alpha} = \mu_h / L_h$, we have

$$-\hat{\alpha} + \frac{L_h}{2\mu_h}\hat{\alpha}^2 = -\hat{\alpha} + \frac{L_h}{2\mu_h}\frac{\mu_h}{L_h}\hat{\alpha} = -\frac{1}{2}\hat{\alpha},$$

and

$$\begin{aligned} f(\mathbf{x}_{h,k} + \hat{\alpha}\hat{\mathbf{d}}_{h,k}) &\leq f_{h,k} - \frac{\hat{\alpha}}{2}\chi_{H,k}^2, \\ &\leq f_{h,k} + \frac{\hat{\alpha}}{2}\nabla f_{h,k}^T \hat{\mathbf{d}}_{h,k}, \\ &< f_{h,k} + \rho_1 \hat{\alpha} \nabla f_{h,k}^T \hat{\mathbf{d}}_{h,k}, \end{aligned}$$

which satisfies the Armijo's rule. Therefore, the line search will return a stepsize $\alpha_{h,k} \geq \hat{\alpha} = (\beta_{ls}\mu_h)/L_h$. Using the fact that

$$\frac{1}{L_h}\|\mathbf{R}\nabla f(x_k)\|^2 \leq (\mathbf{R}\nabla f(x_k))^T [\nabla_H^2 f(x_k)]^{-1} \mathbf{R}\nabla f(x_k) = \chi_{H,k}^2,$$

we obtain

$$\begin{aligned} f(\mathbf{x}_{h,k} + \alpha_{h,k}\hat{\mathbf{d}}_{h,k}) - f_{h,k} &\leq \rho_1 \alpha_{h,k} \nabla f_{h,k}^T \hat{\mathbf{d}}_{h,k}, \\ &\leq -\rho_1 \hat{\alpha} \chi_{H,k}^2, \\ &\leq -\rho_1 \frac{\beta_{ls}\mu_h}{L_h^2}\|\mathbf{R}\nabla f_{h,k}\|^2, \\ &\leq -\frac{\rho_1 \kappa^2 \beta_{ls}\mu_h}{L_h^2}\|\nabla f_{h,k}\|^2, \qquad \text{(from (5.8))} \end{aligned}$$

as required. ∎

Using the result in Lemma 5.9, we derive the guaranteed reduction in function value in the following two lemmas.

Lemma 5.10 *Let* $\Lambda \triangleq \min\left\{\Lambda_h, \frac{\rho_1\kappa^2\beta_{ls}\mu_h}{L_h^2}\right\}$. *Then the step* **d** *in Algorithm 5.1 will lead to*

$$f_{h,k} - f_{h,k+1} \geq \Lambda\|\nabla f_{h,k}\|^2,$$

where ρ_1, κ, *and* β_{ls} *are user-defined parameters in Algorithm 5.1.* L_h *and* μ_h *are defined in Assumption 5.1.* Λ_h *is defined in Assumption 5.6.*

Proof This is a direct result from Lemma 5.9 and Assumption 5.6. ∎

Lemma 5.11 *Suppose*

$$\mathcal{R}(\mathbf{x}_{h,0}) \triangleq \max_{\mathbf{x}_h\in\mathbb{R}^N}\{\|\mathbf{x}_h - \mathbf{x}_{h,\star}\| : f_h(\mathbf{x}_h) \leq f_h(\mathbf{x}_{h,0})\}.$$

Then step in Algorithm 5.1 will guarantee

$$f_{h,k} - f_{h,k+1} \geq \frac{\Lambda}{\mathcal{R}^2(\mathbf{x}_{h,0})}\left(f_{h,k} - f_{h,\star}\right)^2,$$

where Λ *is defined in Lemma 5.10.*

Proof By convexity, for $k = 0, 1, 2, \ldots,$

$$\begin{aligned} f_{h,k} - f_{h,\star} &\leq \langle\nabla f_{h,k}, \mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\rangle, \\ &\leq \|\nabla f_{h,k}\|\ \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|, \\ &\leq \mathcal{R}(\mathbf{x}_{h,0})\|\nabla f_{h,k}\|. \end{aligned}$$

Using Lemma 5.10, we have

$$\begin{aligned} f_{h,k} - f_{h,\star} &\leq \mathcal{R}(\mathbf{x}_{h,0})\sqrt{\Lambda^{-1}\left(f_{h,k} - f_{h,k+1}\right)}, \\ \left(\frac{f_{h,k} - f_{h,\star}}{\mathcal{R}(\mathbf{x}_{h,0})}\right)^2 &\leq \Lambda^{-1}\left(f_{h,k} - f_{h,k+1}\right), \\ \Lambda\left(\frac{f_{h,k} - f_{h,\star}}{\mathcal{R}(\mathbf{x}_{h,0})}\right)^2 &\leq f_{h,k} - f_{h,k+1}, \end{aligned}$$

as required. ∎

The constant Λ in Lemma 5.11 depends on Λ_h, which is introduced in Assumption 5.6. This constant depends on both the fine correction step chosen and the user-defined parameter ρ_1 in Armijo's rule. For instance,

$$\Lambda_h = \begin{cases} \dfrac{\rho_1 \mu_h}{L_h^2} & \text{if} \quad \mathbf{d}_{h,k} = -[\nabla^2 f_{h,k}]^{-1} \nabla f_{h,k}, \\ \dfrac{\rho_1}{L_h} & \text{if} \quad \mathbf{d}_{h,k} = -\nabla f_{h,k}. \end{cases}$$

In order to derive the convergence rate in this section, we use the following lemma on nonnegative scalar sequences.

Lemma 5.12 ([BT13]) *Let $\{A_k\}_{k\geq 0}$ be a nonnegative sequence of the real numbers satisfying*

$$A_k - A_{k+1} \geq \gamma A_k^2, \quad k = 0, 1, 2, \ldots,$$

and

$$A_0 \leq \frac{1}{q\gamma}$$

for some positive γ and q. Then

$$A_k \leq \frac{1}{\gamma(k+q)}, \quad k = 0, 1, 2, \ldots,$$

and so

$$A_k \leq \frac{1}{\gamma k}, \quad k = 0, 1, 2, \ldots.$$

Proof see Lemma 3.5 in [BT13]. ∎

Combining the above results, we obtain the rate of convergence.

Theorem 5.13 *Let $\{\mathbf{x}_k\}_{k\geq 0}$ be the sequence that is generated by Algorithm 5.1. Then,*

$$f_{h,k} - f_{h,\star} \leq \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda} \frac{1}{2+k},$$

where Λ and $\mathcal{R}(\cdot)$ are defined as in Lemma 5.10 and 5.11, respectively.

Proof Notice that by Lemma 5.11

$$f_{h,k} - f_{h,k+1} \geq \frac{\Lambda}{\mathcal{R}^2(\mathbf{x}_{h,0})} \left(f_{h,k} - f_{h,\star}\right)^2$$

and so

$$\left(f_{h,k} - f_{h,\star}\right) - \left(f_{h,k+1} - f_{h,\star}\right) \geq \frac{\Lambda}{\mathcal{R}^2(\mathbf{x}_{h,0})} \left(f_{h,k} - f_{h,\star}\right)^2.$$

Also, we have

$$\begin{aligned} f_{h,0} - f_{h,\star} \leq \frac{L_h}{2}\|\mathbf{x}_{h,0} - \mathbf{x}_{h,\star}\|^2 \leq \frac{L_h}{2}\mathcal{R}^2(\mathbf{x}_{h,0}) \leq \frac{L_h^2\mathcal{R}^2(\mathbf{x}_{h,0})}{2\mu_h} &\leq \frac{L_h^2\mathcal{R}^2(\mathbf{x}_{h,0})}{2\mu_h\beta_{ls}\kappa^2\rho_1}, \\ &\leq \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{2\Lambda}, \end{aligned}$$

where the first two inequalities hold because of descent lemma and the definition of $\mathcal{R}(\cdot)$ in Lemma 5.11, and the last inequality holds because of the definition of Λ in Lemma 5.10. We also use the fact that $\mu_h \leq L_h$ and $\beta_{ls}, \kappa, \rho_1 \leq 1$.

Let $A_k \triangleq f_{h,k} - f_{h,\star}$, $\gamma \triangleq \dfrac{\Lambda}{R^2(\mathbf{x}_{h,0})}$, and $q \triangleq 2$. By applying Lemma 5.12, we have

$$f_{h,k} - f_{h,\star} \leq \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda}\frac{1}{2+k},$$

as required. ∎

Theorem 5.13 provides the sublinear convergence of Algorithm 5.1. We emphasize that the rate is inversely proportional to $\Lambda = \min\{\Lambda_h, \rho_1\kappa^2\mu_h/L_h^2\}$, and so small κ would result in low convergence. Therefore, even though κ could be arbitrary small, it is not desirable in terms of worst case complexity. Note that κ is a user-defined parameter for determining whether coarse correction step would be used. If κ is chosen to be too large, then it is less likely that the coarse correction step would be used. In the extreme case where $\kappa \geq \|\mathbf{R}\|$, coarse correction

step would not be deployed because

$$\|\mathbf{R}\nabla f_{h,k}\| \leq \|\mathbf{R}\|\|\nabla f_{h,k}\|,$$

and so Algorithm 5.1 reduces to the standard variable metric method. Therefore, there is a trade-off between the worst case complexity and the likelihood that coarse correction step would be deployed.

Bear in mind that one can deploy GAMA without using any fine correction step, as stated in Section 5.2.2. In this case the criterion (5.8) would not be used, but we clarify that the analysis in this section is still valid as long as we assume there are constants κ, ϵ such that criterion (5.8) is always satisfied.

5.3.2 Maximum Number of Iterations of Coarse Correction Step

We now discuss the maximum number of coarse correction steps in Algorithm 5.1. The following lemma will state the sufficient conditions for not taking any coarse correction step.

Lemma 5.14 *No coarse correction step in Algorithm 5.1 will be taken when*

$$\|\nabla f_{h,k}\| \leq \frac{\epsilon}{\nu_U},$$

where $\nu_U = \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$*, and* ϵ *is a user-defined parameter in Algorithm 5.1.*

Proof Recall that in Algorithm 5.1, the coarse step is only taken when $\|\mathbf{R}\nabla f_{h,k}\| > \epsilon$. We have,

$$\|\mathbf{R}\nabla f_{h,k}\| \leq \nu_U\|\nabla f_{h,k}\| \leq \nu_U\frac{\epsilon}{\nu_U} = \epsilon,$$

and so no coarse correction step will be taken. ∎

The above lemma states the condition when the coarse correction step would not be performed. We then investigate the maximum number of iterations to achieve that sufficient condition.

Lemma 5.15 *Let $\{\mathbf{x}_k\}_{k\geq 0}$ be a sequence generated by Algorithm 5.1. Then, $\forall \bar{\epsilon}, \bar{k} > 0$ such that,*

$$\bar{k} \geq \left(\frac{1}{\bar{\epsilon}}\right)^2 \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda^2} - 2,$$

we obtain

$$\|\nabla f_h(\mathbf{x}_{h,\bar{k}})\| \leq \bar{\epsilon},$$

where Λ and $\mathcal{R}(\cdot)$ are defined as in Lemma 5.10 and 5.11, respectively.

Proof We know that

$$\Lambda \|\nabla f_{h,k}\|^2 \;\leq\; f_{h,k} - f_{h,k+1}.$$

Also, we have,

$$f_{h,k} - f_{h,\star} \leq \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda} \frac{1}{2+k}.$$

Therefore,

$$\begin{aligned} \|\nabla f_{h,k}\|^2 &\leq \frac{1}{\Lambda}\left(f_{h,k} - f_{h,k+1}\right), \\ &\leq \frac{1}{\Lambda}\left(f_{h,k} - f_{h,\star}\right), \\ &\leq \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda^2} \frac{1}{2+k}. \end{aligned}$$

For

$$k = \left(\frac{1}{\bar{\epsilon}}\right)^2 \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda^2} - 2,$$

we have

$$\|\nabla f_{h,k}\| \leq \sqrt{\frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda^2} \frac{1}{2+k}} \leq \sqrt{\frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda^2} (\bar{\epsilon})^2 \frac{\Lambda^2}{\mathcal{R}^2(\mathbf{x}_{h,0})}} = \bar{\epsilon},$$

as required. ∎

By integrating the above results, we obtain the maximum number of iterations to achieve $\|\nabla f_{h,k}\| \leq \epsilon/\nu_U$. That is, no coarse correction step will be taken after

$$\left(\frac{\nu_U}{\epsilon}\right)^2 \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda^2} - 2 \quad \text{iterations.}$$

Notice that the smaller ϵ, the more coarse correction step will be taken. Depending on the choice of $\mathbf{d}_{h,k}$, the choice of ϵ could be different. For example, if $\mathbf{d}_{h,k}$ is chosen as the Newton step where $\mathbf{d}_{h,k} = -[\nabla^2 f_{h,k}]^{-1}\nabla f_{h,k}$, one good choice of ϵ could be $3\nu_U(1-2\rho_1)\mu_h^2/L_h$ if μ_h and L_h are known. This is because Newton's method achieves quadratic rate of convergence when $\|\nabla f_{h,k}\| \leq 3(1-2\rho_1)\mu_h^2/L_h$ [BV04]. Therefore, for such ϵ, no coarse correction step would be taken when the Newton method performs in its quadratically convergent phase.

5.3.3 Quadratic Phase in Subspace

We now state the required condition for the stepsize to achieve $\alpha_{h,k} = 1$, and then we will show that when $\|\mathbf{R}\nabla f_{h,k}\|$ is sufficiently small, the coarse correction step would reduce $\|\mathbf{R}\nabla f_{h,k}\|$ quadratically. The results below are analogous to the analysis of the Newton's method in [BV04].

Lemma 5.16 *Suppose the coarse correction step $\hat{\mathbf{d}}_{h,k}$ in Algorithm 5.1 is taken, then $\alpha_{h,k} = 1$ when*

$$\|\mathbf{R}\nabla f_{h,k}\| \leq \eta = \frac{3\mu_h^2}{M_h}(1-2\rho_1),$$

where ρ_1 is a user-defined parameter in Algorithm 5.1. M_h and μ_h are defined in Assumption 5.1.

Proof By Lipschitz continuity (5.3),

$$\|\nabla^2 f_h(\mathbf{x}_{h,k} + \alpha\hat{\mathbf{d}}_{h,k}) - \nabla^2 f_{h,k}\| \leq \alpha M_h \|\hat{\mathbf{d}}_{h,k}\|,$$

which implies

$$\|\hat{\mathbf{d}}_{h,k}^T(\nabla^2 f_h(\mathbf{x}_{h,k}+\alpha\hat{\mathbf{d}}_{h,k})-\nabla^2 f_{h,k})\hat{\mathbf{d}}_{h,k}\| \le \alpha M_h\|\hat{\mathbf{d}}_{h,k}\|^3.$$

Let $\tilde{f}(\alpha)=f_h(\mathbf{x}_{h,k}+\alpha\hat{\mathbf{d}}_{h,k})$. Then the above inequality can be rewritten as

$$|\tilde{f}''(\alpha)-\tilde{f}''(0)| \le \alpha M_h\|\hat{\mathbf{d}}_{h,k}\|^3,$$

and so

$$\tilde{f}''(\alpha) \le \tilde{f}''(0)+\alpha M_h\|\hat{\mathbf{d}}_{h,k}\|^3.$$

Since $\tilde{f}''(0)=\hat{\mathbf{d}}_{h,k}^T\nabla^2 f_{h,k}\hat{\mathbf{d}}_{h,k}=\chi_{H,k}^2$,

$$\tilde{f}''(\alpha) \le \chi_{H,k}^2+\alpha M_h\|\hat{\mathbf{d}}_{h,k}\|^3.$$

By integration,

$$\tilde{f}'(\alpha) \le \tilde{f}'(0)+\alpha\chi_{H,k}^2+(\alpha^2/2)M_h\|\hat{\mathbf{d}}_{h,k}\|^3.$$

Similarly, $\tilde{f}'(0)=\nabla f_{h,k}^T\hat{\mathbf{d}}_{h,k}=-\chi_{H,k}^2$, and so

$$\tilde{f}'(\alpha) \le -\chi_{H,k}^2+\alpha\chi_{H,k}^2+(\alpha^2/2)M_h\|\hat{\mathbf{d}}_{h,k}\|^3.$$

Integrating the above inequality, we obtain

$$\tilde{f}(\alpha) \le \tilde{f}(0)-\alpha\chi_{H,k}^2+(\alpha^2/2)\chi_{H,k}^2+(\alpha^3/6)M_h\|\hat{\mathbf{d}}_{h,k}\|^3.$$

Recall that $\mu_h\|\hat{\mathbf{d}}_{h,k}\|^2 \le \hat{\mathbf{d}}_{h,k}^T\nabla^2 f_{h,k}\hat{\mathbf{d}}_{h,k}=\chi_{H,k}^2$; thus,

$$\tilde{f}(\alpha) \le \tilde{f}(0)-\alpha\chi_{H,k}^2+\frac{\alpha^2}{2}\chi_{H,k}^2+\frac{\alpha^3 M_h}{6\mu_h^{3/2}}\chi_{H,k}^3.$$

Let $\alpha = 1$,

$$\begin{aligned}\tilde{f}(1) - \tilde{f}(0) &\leq -\chi_{H,k}^2 + \frac{1}{2}\chi_{H,k}^2 + \frac{M_h}{6\mu_h^{3/2}}\chi_{H,k}^3 \\ &= -\left(\frac{1}{2} - \frac{M_h}{6\mu_h^{3/2}}\chi_{H,k}\right)\chi_{H,k}^2.\end{aligned}$$

Using the fact that

$$\|\mathbf{R}\nabla f_{h,k}\| \leq \eta = \frac{3\mu_h^2}{M_h}(1 - 2\rho_1)$$

and

$$\chi_{H,k} = ((\mathbf{R}\nabla f_{h,k})^T[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla f_{h,k})^{1/2} \leq \frac{1}{\sqrt{\mu_h}}\|\mathbf{R}\nabla f_{h,k}\|,$$

we have

$$\chi_{H,k} \leq \frac{3\mu_h^{3/2}}{M_h}(1 - 2\rho_1) \quad \Longleftrightarrow \quad \rho_1 \leq \frac{1}{2} - \frac{M_h}{6\mu_h^{3/2}}\chi_{H,k}.$$

Therefore,

$$\tilde{f}(1) - \tilde{f}(0) \leq -\rho_1\chi_{H,k}^2 = \rho_1\nabla f_{h,k}^T\hat{\mathbf{d}}_{h,k},$$

and we have $\alpha_{h,k} = 1$ when $\|\mathbf{R}\nabla f_{h,k}\| \leq \eta$. ∎

The above lemma yields the following theorem.

Theorem 5.17 *Suppose the coarse correction step $\hat{\mathbf{d}}_{h,k}$ in Algorithm 5.1 is taken and $\alpha_{h,k} = 1$, then*

$$\|\mathbf{R}\nabla f_{h,k+1}\| \leq \frac{\nu_U^3\nu_L^4 M_h}{2\mu_h^2}\|\mathbf{R}\nabla f_{h,k}\|^2,$$

where M_h and μ_h are defined in Assumption 5.1, $\nu_U = \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$ and $\nu_L = \|\mathbf{P}^+\|$.

Proof Since $\alpha_{h,k} = 1$, we have

$$\begin{aligned}
\|\mathbf{R}\nabla f_{h,k+1}\| &= \|\mathbf{R}\nabla f_h(\mathbf{x}_{h,k} + \hat{\mathbf{d}}_{h,k}) - \mathbf{R}\nabla f_{h,k} + \mathbf{R}\nabla^2 f_{h,k}\mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla f_{h,k}\| \\
&\leq \|\mathbf{R}\|\ \|\nabla f_h(\mathbf{x}_{h,k} + \hat{\mathbf{d}}_{h,k}) - \nabla f_{h,k} - \nabla^2 f_{h,k}\hat{\mathbf{d}}_{h,k}\| \\
&\leq \nu_U \left\| \int_0^1 (\nabla^2 f_h(\mathbf{x}_{h,k} + t\hat{\mathbf{d}}_{h,k}) - \nabla^2 f_{h,k})\hat{\mathbf{d}}_{h,k}\ dt \right\| \\
&\leq \nu_U \frac{M_h}{2}\|\hat{\mathbf{d}}_{h,k}\|^2.
\end{aligned}$$

Notice that

$$\begin{aligned}
\|\hat{\mathbf{d}}_{h,k}\| &= \|\mathbf{P}[\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}]^{-1}\mathbf{R}\nabla f_{h,k}\| \\
&\leq \|\mathbf{P}\|\ \|[\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}]^{-1}\|\ \|\mathbf{R}\nabla f_{h,k}\| \\
&\leq \frac{\nu_U \nu_L^2}{\mu_h}\|\mathbf{R}\nabla f_{h,k}\|.
\end{aligned}$$

Thus,

$$\|\mathbf{R}\nabla f_{h,k+1}\| \leq \frac{\nu_U^3 \nu_L^4 M_h}{2\mu_h^2}\|\mathbf{R}\nabla f_{h,k}\|^2,$$

as required. ∎

The above theorem states the quadratic convergence of $\|\nabla f_{h,k}\|$ within the subspace range($\mathbf{R}$). However, it does not give insight on the convergence behaviour on the full space $\mathbb{R}^N$. To address this, we study the composite rate of convergence in the next section.

5.3.4 Composite Convergence Rate

At the end of this section, we study the convergence properties of the coarse correction step when the incumbent is sufficiently close to the solution. In particular, we deploy the idea of composite convergence rate in [EM15], and consider the convergence of the coarse correction step as a combination of linear and quadratic convergence.

The reason for proving composite convergence is due to the broadness of GAMA. Suppose in the naïve case when $\mathbf{P} = \mathbf{R} = \mathbf{I}$, then the coarse correction step in GAMA becomes Newton's

method. In this case we expect quadratic convergence when the incumbent is sufficiently close to the solution. On the other hand, suppose $\mathbf{P}$ is any column of $\mathbf{I}$ and $\mathbf{R} = \mathbf{P}^T$, then the coarse correction step is a (weighted) coordinate descent direction, as described in Section 5.2.4. One should expect not more than linear convergence in that case. Therefore, both quadratic convergence and linear convergence are not suitable for GAMA, and one needs a combination of them. In this chapter, we propose to use composite convergence, and show that it can better explain the convergence of different variants of GAMA.

We would like to emphasize the difference between our setting and [EM15]. To the best of our knowledge, composite convergence rate was used in [EM15] to study subsample Newton methods for machine learning problems without dimensionality reduction. In this chapter, the class of problems that we consider is not restricted to machine learning, and we focus on the Galerkin model, which is a reduced dimension model. The results presented in this section are not direct results of the approach in [EM15]. In particular, if the exact analysis of [EM15] is taken, the derived composite rate would not be useful in our setting, because the coefficient of the linear component would be greater than 1.

Theorem 5.18 *Suppose the coarse correction step $\hat{\mathbf{d}}_{h,k}$ in Algorithm 5.1 is taken and $\alpha_{h,k} = 1$, then*

$$\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| \leq \|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\| + \frac{M_h \nu_U^2 \nu_L^2}{2\mu_h}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2, \quad (5.17)$$

where M_h and μ_h are defined in Assumption 5.1, $\nu_U = \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$ and $\nu_L = \|\mathbf{P}^+\|$. The operator ∇_H^2 is defined in (5.13).

Proof Denote

$$\tilde{\mathbf{Q}} = \int_0^1 \nabla^2 f(\mathbf{x}_{h,\star} - t(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}))\mathrm{d}t,$$

we have

$$\begin{aligned}
\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star} &= \mathbf{x}_{h,k} - \mathbf{x}_{h,\star} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla f_{h,k} \\
&= \mathbf{x}_{h,k} - \mathbf{x}_{h,\star} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\tilde{\mathbf{Q}}(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}) \\
&= \left(\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\tilde{\mathbf{Q}}\right)(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}) \\
&= \left(\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\right)(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}) \\
&\quad + \left(\mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\tilde{\mathbf{Q}}\right)(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}) \\
&= \left(\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\right)(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}) \\
&\quad + \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\left(\nabla^2 f_{h,k} - \tilde{\mathbf{Q}}\right)(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}),
\end{aligned}$$

where the last equality holds since

$$\left(\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\right)\mathbf{PR} = \mathbf{PR} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\mathbf{PR} = \mathbf{PR} - \mathbf{PR} = \mathbf{0}.$$

Note that

$$\|\nabla^2 f_{h,k} - \tilde{\mathbf{Q}}\| = \left\|\nabla^2 f_{h,k} - \int_0^1 \nabla^2 f(\mathbf{x}_{h,\star} - t(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}))\mathrm{d}t\right\| \leq \frac{M_h}{2}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|.$$

Therefore,

$$\begin{aligned}
\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| &\leq \|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\| \\
&\quad + \|\mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\|\frac{M_h}{2}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2, \\
&\leq \|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\| \\
&\quad + \frac{M_h\nu_U^2\nu_L^2}{2\mu_h}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2,
\end{aligned}$$

as required. ∎

Theorem 5.18 provides the composite convergence rate for the coarse correction step. However, some terms remain unclear, in particular $\|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|$. Notice that in the case

when $\text{rank}(\mathbf{P}) = N$ (i.e. $\mathbf{P}$ is invertible),

$$\begin{aligned}\|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\| &= \|\mathbf{I} - \mathbf{P}[\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|, \\ &= \|\mathbf{I} - \mathbf{P}\mathbf{P}^{-1}[\nabla^2 f_{h,k}]^{-1}\mathbf{R}^{-1}\mathbf{R}\nabla^2 f_{h,k}\|, \\ &= 0.\end{aligned}$$

It is intuitive to consider that $\|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|$ should be small and less than 1 when $\text{rank}(\mathbf{P})$ is close to but not equal to N. However, the above intuition is not true, and we prove this in the following lemma.

Lemma 5.19 *Suppose rank*$(\mathbf{P}) \neq N$*, then*

$$1 \leq \|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\| \leq \sqrt{\frac{L_h}{\mu_h}},$$

where L_h and μ_h are defined in Assumption 5.1. The operator ∇_H^2 is defined in (5.13).

Proof Since $\nabla^2 f_{h,k}$ is a positive definite matrix, consider the eigendecomposition of $\nabla^2 f_{h,k}$,

$$\nabla^2 f_{h,k} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T,$$

where $\mathbf{\Sigma}$ is a diagonal matrix containing the eigenvalues of $\nabla^2 f_{h,k}$, and $\mathbf{U}$ is a orthogonal matrix where its columns are eigenvectors of $\nabla^2 f_{h,k}$.

In this proof, we rely on results in orthogonal projection for real matrices (see [HMT11] for more details). An orthogonal projector is an symmetric matrix such that $\mathbf{\Gamma}^2 = \mathbf{\Gamma}$, which implies $0 \preceq \mathbf{\Gamma} \preceq \mathbf{I}$. For a full column rank matrix $\mathbf{M}$, it unique orthogonal projector is

$$\mathbf{\Gamma}_\mathbf{M} = \mathbf{M}(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T,$$

with $\text{range}(\mathbf{\Gamma}_\mathbf{M}) = \text{range}(\mathbf{M})$. The matrix $\mathbf{I} - \mathbf{\Gamma}_\mathbf{M}$ is also an orthogonal projector.

Using the above results and definitions, we have

$$\begin{aligned}
& \mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k} \\
= \; & \mathbf{I} - \mathbf{P}[\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}]^{-1}\mathbf{R}\nabla^2 f_{h,k}, \\
= \; & \mathbf{U}\boldsymbol{\Sigma}^{-1/2}\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T - \mathbf{U}\boldsymbol{\Sigma}^{-1/2}\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}[\mathbf{R}\mathbf{U}\boldsymbol{\Sigma}^{1/2}\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}]^{-1}\mathbf{R}\mathbf{U}\boldsymbol{\Sigma}^{1/2}\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T, \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(by eigendecomposition)} \\
= \; & \mathbf{U}\boldsymbol{\Sigma}^{-1/2}\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T \\
& -\mathbf{U}\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P})[(\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P})^T(\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P})]^{-1}(\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P})^T\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T, \\
& \qquad\qquad \text{(grouping } (\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P})\text{'s to be in the form of orthogonal projector)} \\
= \; & \mathbf{U}\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T,
\end{aligned}$$

where $\boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}}$ is the orthogonal projection operator onto the range of $\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}$, and so

$$\begin{aligned}
\|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\| &= \|\mathbf{U}\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\|, \\
&= \|\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\|.
\end{aligned}$$

For the upper bound, we have

$$\|\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\| \leq \|\boldsymbol{\Sigma}^{-1/2}\|\|(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\|\|\boldsymbol{\Sigma}^{1/2}\| \leq \sqrt{\frac{L_h}{\mu_h}},$$

since $\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}}$ is an orthogonal projector and $\|(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\| \leq 1$. For the lower bound, we have

$$\begin{aligned}
\|\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\| &= \|\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\|, \\
&= \|\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\|, \\
&\leq \|\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\|\|\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\|, \\
&= \|\boldsymbol{\Sigma}^{-1/2}(\mathbf{I} - \boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\boldsymbol{\Sigma}^{1/2}\|^2.
\end{aligned}$$

The assumption $\text{rank}(\mathbf{P}) \neq N$ implies that $\text{range}(\mathbf{P}) \neq \text{range}(\mathbf{I})$ and so $\text{range}(\boldsymbol{\Gamma}_{\boldsymbol{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}}) \neq$

range($\mathbf{I}$). Thus,

$$\mathbf{I} \neq \mathbf{\Gamma}_{\mathbf{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}} \quad \text{and} \quad \|\mathbf{\Sigma}^{-1/2}(\mathbf{I} - \mathbf{\Gamma}_{\mathbf{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\mathbf{\Sigma}^{1/2}\| \neq 0.$$

Therefore, $1 \leq \|\mathbf{\Sigma}^{-1/2}(\mathbf{I} - \mathbf{\Gamma}_{\mathbf{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\mathbf{\Sigma}^{1/2}\|$, as required. ∎

Lemma 5.19 clarifies the fact that the term $\|\mathbf{I}-\mathbf{P}[\nabla^2_H f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|$ is at least 1 when $n < N$. This fact reduces the usefulness of the composite convergence rate in Theorem 5.18. In Section 5.5-5.7, we will investigate different Galerkin models, and show that $\|(\mathbf{I}-\mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$ is sufficiently small in those cases.

5.4 Complexity Analysis

In this section we will perform the complexity analysis for both the Newton's method and GAMA. Our complexity analysis for Newton's method is a variant of the results in [BV04, Kan52, Pol87]. The main difference is that in this chapter we focus on the complexity that yield $\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| \leq \epsilon_h$ accuracy instead of $\|\nabla f_{h,k}\| \leq \epsilon_h$. This choice is made for simpler comparison with GAMA. At the end of this section, we compare the complexity of Newton's method and GAMA, and we will state the condition for which GAMA has lower complexity.

5.4.1 Complexity Analysis: Newton's Method

It is known that for Newton's method, the algorithm enters its quadratic convergence phase when $\alpha_{h,k} = 1$, with

$$\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| \leq \frac{M_h}{2\mu_h}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2.$$

The above equation, however, does not guarantee that $\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\|$ is a contraction. To obtain this guarantee, it requires

$$\frac{M_h}{2\mu_h}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| < 1 \quad \Longleftrightarrow \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| < \frac{2\mu_h}{M_h}. \tag{5.18}$$

Moreover, $\alpha_{h,k} = 1$ when

$$\|\nabla f_{h,k}\| \leq 3(1-2\rho_1)\frac{\mu_h^2}{L_h}. \tag{5.19}$$

In what follows we will first prove the number of iterations needed to satisfy condition (5.18)-(5.19) (called the damped Newton phase), and we will then compute the number of iterations needed in the quadratically convergent phase. To this end, we define the following two variables:

- k_d: The number of iterations in the **d**amped Newton phase.
- k_q: The number of iterations in the **q**uadratically convergent phase.

Thus, the total number of iterations needed is $k_d + k_q$.

Lemma 5.20 *Suppose Newton's method is performed. Then conditions (5.18)-(5.19) are satisfied after*

$$k_d \geq \left(\frac{1}{\epsilon_N}\right)^2 \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda_N^2} - 2$$

iterations, where

$$\epsilon_N \triangleq \underbrace{\min\left\{\frac{3}{2}(1-2\rho_1), \delta\right\}}_{\triangleq \eta_N} \frac{2\mu_h^2}{M_h},\ \forall \delta \in (0,1), \qquad \Lambda_N \triangleq \frac{\rho_1 \beta_{ls} \mu_h}{L_h^2}.$$

Note that ρ_1 and β_{ls} are user-defined parameters in Armijo's rule as Algorithm 5.1; M_h, L_h, and μ_h are defined in Assumption 5.1; $\mathcal{R}(\cdot)$ is defined in Lemma 5.11.

Proof It is known that for Newton's method

$$f_{h,k+1} - f_{h,k} \leq -\Lambda_N \|\nabla f_{h,k}\|^2.$$

Using the above equation together with the proofs of Lemma 5.11 and Theorem 5.13, we obtain

$$f_{h,k} - f_{h,\star} \leq \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda_N} \frac{1}{2+k}.$$

Therefore, using the proof of Lemma 5.15, it takes a finite number of iterations, k_d, to achieve $\|\nabla f_{h,k_d}\| \leq \epsilon_N$ for $\epsilon_N > 0$, and

$$k_d \leq \left(\frac{1}{\epsilon_N}\right)^2 \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda_N^2} - 2.$$

By convexity and the definition of ϵ_N, we obtain

$$\|\mathbf{x}_{h,k_d} - \mathbf{x}_{h,\star}\| \leq \frac{1}{\mu_h}\|\nabla f_{h,k_d}\| \quad \leq \quad \frac{1}{\mu_h}\epsilon_N = \frac{1}{\mu_h}\min\left\{\frac{3}{2}(1-2\rho_1), \delta\right\}\frac{2\mu_h^2}{M_h} < \frac{2\mu_h}{M_h}.$$

So we obtain the desired result. ∎

Lemma 5.20 gives k_d, the number of iterations required in order to enter the quadratic phase. In the following lemma we derive k_q.

Lemma 5.21 *Suppose Newton's method is performed and $\|\nabla f_{h,0}\| \leq \epsilon_N$, where ϵ_N is defined in Lemma 5.20. Then, for ϵ_h and k_q such that*

$$\epsilon_h \in (0,1), \quad \textit{and} \quad k_q \geq \frac{1}{\log 2}\log\left(\frac{\log\left(\frac{M_h\epsilon_h}{2\mu_h}\right)}{\log \eta_N}\right) - 1,$$

we obtain $\|\mathbf{x}_{h,k_q} - \mathbf{x}_{h,\star}\| \leq \epsilon_h$. Note that M_h and η_N are defined in Assumption 5.1 and Lemma 5.20, respectively.

Proof Given that

$$\|\mathbf{x}_{h,0} - \mathbf{x}_{h,\star}\| \leq \frac{1}{\mu_h}\|\nabla f_{h,0}\| \leq \frac{\epsilon_N}{\mu_h} \leq \eta_N\frac{2\mu_h}{M_h},$$

we have

$$\begin{aligned}
\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| &\leq \frac{M_h}{2\mu_h}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2 \\
&\leq \left(\frac{M_h}{2\mu_h}\right)\left(\frac{M_h}{2\mu_h}\right)^2 \|\mathbf{x}_{h,k-1} - \mathbf{x}_{h,\star}\|^4 \\
&\leq \left(\frac{M_h}{2\mu_h}\right)^{\sum_{j=0}^{k} 2^j} \|\mathbf{x}_{h,0} - \mathbf{x}_{h,\star}\|^{2^{k+1}} \\
&= \left(\frac{M_h}{2\mu_h}\right)^{2^{k+1}-1} \left(\eta_N \frac{2\mu_h}{M_h}\right)^{2^{k+1}} \\
&= \frac{2\mu_h}{M_h}\eta_N^{2^{k+1}}.
\end{aligned}$$

To achieve the desired accuracy, we require

$$\begin{aligned}
\frac{2\mu_h}{M_h}\eta_N^{2^{k_q+1}} &\leq \epsilon_h, \\
2^{k_q+1}\log\eta_N &\leq \log\left(\frac{M_h\epsilon_h}{2\mu_h}\right), \\
(k_q+1)\log 2 &\geq \log\left(\frac{\log\left(\frac{M_h\epsilon_h}{2\mu_h}\right)}{\log\eta_N}\right), \\
k_q &\geq \frac{1}{\log 2}\log\left(\frac{\log\left(\frac{M_h\epsilon_h}{2\mu_h}\right)}{\log\eta_N}\right) - 1.
\end{aligned}$$

So we obtain the desired result. ∎

Combining the results in Lemma 5.20 and Lemma 5.21, we obtain the complexity of Newton's method.

Theorem 5.22 *Suppose Newton's method is performed and $k = k_d + k_q$, where k_d and k_q are defined as in Lemma 5.20 and Lemma 5.21, respectively. Then we obtain the ϵ_h-accuracy $\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| \leq \epsilon_h$ with the complexity*

$$\mathcal{O}\left((k_d + k_q)N^3\right).$$

Proof The total complexity is the number of iterations, $k_d + k_q$, multiply by the cost per

iteration, which is $\mathcal{O}(N^3)$. ∎

5.4.2 Complexity Analysis: GAMA

We follow the same strategy to compute the complexity of GAMA. In order to avoid unnecessary complications in notations, in the section, we let r_1 and r_2 to be the composite rate in which

$$\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| \leq r_1\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| + r_2\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2, \tag{5.20}$$

when

$$\|\mathbf{R}\nabla f_{h,k}\| \leq \frac{3\mu_h^2}{M_h}(1 - 2\rho_1). \tag{5.21}$$

For $\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\|$ in (5.20) to be a contraction, we need $r_1 < 1$ and

$$\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| < \frac{1 - r_1}{r_2}. \tag{5.22}$$

We clarify that the above form in (5.20) is not exactly in the same form of the composite rate in Section 5.3.4, where $\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$ is used instead of $\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|$. The latter case is used solely for simpler analysis, and does not contradict with the results presented in Section 5.3.4; in particular, one can simply let

$$r_1 = \|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|\frac{\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|}{\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|}. \tag{5.23}$$

In order to guarantee convergence, we simply assume one fine correction step is taken after a fixed number of coarse correction steps. For the purpose of simplifying analysis, we make the following assumptions on the fine correction step taken.

Assumption 5.23 *The coarse correction step of Algorithm 5.1 has the following properties:*

1. *One fine correction step is taken for every K_H coarse correction steps.*
2. *The computational cost of each fine correction step is $\mathcal{O}(C_h)$, for some C_h.*

3. *When the composite rate (5.20) applies for the coarse correction steps, the fine correction step satisfies*

$$\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| \leq \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|.$$

Recall that GAMA only achieves the composite rate when condition (5.21) is satisfied, as stated in Lemma 5.16 and Theorem 5.18. When (5.21) does not hold, a global sublinear rate of convergence is still guaranteed, as concluded in Theorem 5.13. We shall call the former case and the latter case as composite convergent phase and sublinear convergent phase, respectively.

In the following lemma, we compute the number of iterations needed for both composite convergent phase and sublinear convergent phase. Similar to the case of Newton's method, we define the following notation:

- k_s: The number of iterations in the **s**ublinear convergent phase.
- k_c: The number of iterations in the **c**omposite convergent phase.

Thus, the total number of iterations of GAMA would be $k_s + k_c$.

Lemma 5.24 *Suppose Algorithm 5.1 is performed and Assumption 5.23 holds. Then conditions (5.21)-(5.22) are satisfied after*

$$k_s \geq \left(\frac{1}{\epsilon_G}\right)^2 \frac{\mathcal{R}^2(\mathbf{x}_{h,0})}{\Lambda^2} - 2$$

iterations, where

$$\epsilon_G \triangleq \min\left\{\frac{3\mu_h^2}{\nu_U M_h}(1-2\rho_1), \delta\right\},\ \forall \delta \in \left(0, \frac{\mu_h(1-r_1)}{r_2}\right).$$

Note that ρ_1 is a user-defined parameter in Algorithm 5.1; $\nu_U = \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$; M_h and μ_h are defined in Assumption 5.1; Λ and $\mathcal{R}(\cdot)$ are defined in Lemma 5.10 and 5.11, respectively; r_1 and r_2 are defined in (5.20).

Proof Using the result in Lemma 5.15, we obtain

$$\|\nabla f_{h,k_s}\| \leq \epsilon_G.$$

We then show that the above condition is sufficient for $\alpha_{h,k_s} = 1$. By definitions,

$$\|\mathbf{R}\nabla f_{h,k_s}\| \leq \nu_U \|\nabla f_{h,k_s}\| \leq \nu_U \epsilon_G \leq \nu_U \frac{3\mu_h^2}{\nu_U M_h}(1-2\rho_1) = \frac{3\mu_h^2}{M_h}(1-2\rho_1).$$

By Lemma 5.16, $\alpha_{h,k_s} = 1$. On the other hand,

$$\|\mathbf{x}_{h,k_s} - \mathbf{x}_{h,\star}\| \leq \frac{1}{\mu_h}\|\nabla f_{h,k_s}\| \quad < \quad \frac{1}{\mu_h}\frac{\mu_h(1-r_1)}{r_2} = \frac{1-r_1}{r_2}.$$

Therefore, we obtain the desired result. ∎

Lemma 5.24 gives the number of iterations required in the sublinear convergent phase, k_s. In the following lemma, we derive k_c.

Lemma 5.25 *Suppose Algorithm 5.1 is performed, Assumption 5.23 holds, and*

$$\|\nabla f_{h,0}\| \leq \epsilon_G,$$

where ϵ_G is defined in Lemma 5.24. Then for ϵ_h and k_c such that

$$\epsilon_h \in (0,1), \text{ and } k_c \geq \frac{1+1/K_H}{r_1-1}\left(\log\left(\frac{\mu_h\epsilon_h}{\epsilon_G}\right) - \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + \frac{r_2\epsilon_G}{\mu_h \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)}\right),$$

we obtain $\|\mathbf{x}_{h,k_c} - \mathbf{x}_{h,\star}\| \leq \epsilon_h$. Note that μ_h and K_H are defined in Assumption 5.1 and Assumption 5.23, respectively; r_1 and r_2 are defined in (5.20).

Proof Based on Assumption 5.23, if k coarse correction steps are needed, k/K_H fine correction steps would be taken. The total number of searches would then be $k(1+1/K_H)$. Therefore, we first neglect the use of fine correction steps, and consider this factor at the end of the proof by multiplying $(1+1/K_H)$.

We obtain

$$\|\mathbf{x}_{h,0} - \mathbf{x}_{h,\star}\| \leq \frac{1}{\mu_h}\|\nabla f_{h,0}\| \leq \frac{\epsilon_G}{\mu_h}.$$

and $\frac{\epsilon_G}{\mu_h} < \frac{1-r_1}{r_2}$, based on the definition of ϵ_G. Since $\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|$ is a contraction,

$$\|\mathbf{x}_{h,0} - \mathbf{x}_{h,\star}\| \geq \|\mathbf{x}_{h,1} - \mathbf{x}_{h,\star}\| \geq \|\mathbf{x}_{h,2} - \mathbf{x}_{h,\star}\| \geq \dots$$

Based on the above notations, observations, and the fact that the composite rate holds, we obtain

$$\begin{aligned}
\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| &\leq (r_1 + r_2\|\mathbf{x}_{h,k-1} - \mathbf{x}_{h,\star}\|)\,\|\mathbf{x}_{h,k-1} - \mathbf{x}_{h,\star}\| \\
&\leq \left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)\|\mathbf{x}_{h,k-1} - \mathbf{x}_{h,\star}\| \\
&\leq \left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^k\|\mathbf{x}_{h,0} - \mathbf{x}_{h,\star}\| \\
&= \left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^k\frac{\epsilon_G}{\mu_h}.
\end{aligned}$$

We denote $r(k) \triangleq r_1 + r_2\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^k\frac{\epsilon_G}{\mu_h}$ and we obtain

$$\begin{aligned}
\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| &\leq r_1\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| + r_2\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2 \\
&\leq (r_1 + r_2\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|)\,\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| \\
&\leq r(k)\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| \\
&\leq \left(\prod_{j=0}^{k} r(j)\right)\|\mathbf{x}_{h,0} - \mathbf{x}_{h,\star}\| \\
&\leq \left(\prod_{j=0}^{k} r(j)\right)\frac{\epsilon_G}{\mu_h}.
\end{aligned}$$

Therefore, it is sufficient to achieve ϵ_h-accuracy when

$$\begin{aligned}
\left(\prod_{j=0}^{k} r(j)\right)\frac{\epsilon_G}{\mu_h} &\leq \epsilon_h, \\
\prod_{j=0}^{k}\left(r_1 + r_2\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^j \frac{\epsilon_G}{\mu_h}\right) &\leq \frac{\mu_h\epsilon_h}{\epsilon_G}, \\
\sum_{j=0}^{k}\log\left(r_1 + r_2\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^j \frac{\epsilon_G}{\mu_h}\right) &\leq \log\left(\frac{\mu_h\epsilon_h}{\epsilon_G}\right), \\
\sum_{j=1}^{k+1}\log\left(r_1 + r_2\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^{j-1} \frac{\epsilon_G}{\mu_h}\right) &\leq \log\left(\frac{\mu_h\epsilon_h}{\epsilon_G}\right).
\end{aligned}$$

Using calculus, we know that

$$\begin{aligned}
&\sum_{j=1}^{k+1}\log\left(r_1 + r_2\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^{j-1} \frac{\epsilon_G}{\mu_h}\right) && \text{(this is a monotonic series)} \\
\leq\ & \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + \int_1^{k+1}\log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^{x-1}\right)\mathrm{d}x \\
\leq\ & \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + \int_1^{k+1}(r_1 - 1) + r_2\frac{\epsilon_G}{\mu_h}\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^{x-1}\mathrm{d}x && (\log(x) \leq x - 1) \\
\leq\ & \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + k(r_1 - 1) + \frac{r_2\frac{\epsilon_G}{\mu_h}}{r_1 + r_2\frac{\epsilon_G}{\mu_h}}\int_1^{k+1}\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^x \mathrm{d}x \\
\leq\ & \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + k(r_1 - 1) + \frac{r_2\frac{\epsilon_G}{\mu_h}}{r_1 + r_2\frac{\epsilon_G}{\mu_h}}\left(\frac{\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^x}{\log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)}\right|_{x=1}^{x=k+1} && \text{(from integration)} \\
\leq\ & \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + k(r_1 - 1) + \frac{r_2\frac{\epsilon_G}{\mu_h}}{\log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)}\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)^k - \frac{r_2\frac{\epsilon_G}{\mu_h}}{\log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)} \\
\leq\ & \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + k(r_1 - 1) - \frac{r_2\epsilon_G}{\mu_h\log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)} && \left(\text{since } r_1 + r_2\frac{\epsilon_G}{\mu_h} < 1\right).
\end{aligned}$$

So, it is sufficient to achieve ϵ_h-accuracy if

$$\begin{aligned}
\log\left(\frac{\mu_h\epsilon_h}{\epsilon_G}\right) &\geq \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + k(r_1-1) - \frac{r_2\epsilon_G}{\mu_h \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)}, \\
k(r_1-1) &\leq \log\left(\frac{\mu_h\epsilon_h}{\epsilon_G}\right) - \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + \frac{r_2\epsilon_G}{\mu_h \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)}, \\
k &\geq \frac{1}{r_1-1}\left(\log\left(\frac{\mu_h\epsilon_h}{\epsilon_G}\right) - \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right) + \frac{r_2\epsilon_G}{\mu_h \log\left(r_1 + r_2\frac{\epsilon_G}{\mu_h}\right)}\right).
\end{aligned}$$

So we obtain the desired result. ∎

Although the result of Lemma 5.25 states the number of iterations needed for composite convergent phase, the derived result is difficult to interpret. To this end, in the following lemma, we study a special case of Lemma 5.25.

Lemma 5.26 *Consider the setting as in Lemma 5.25 with*

$$\epsilon_G = \min\left\{\frac{3\mu_h^2}{\nu_U M_h}(1-2\rho_1), \frac{\mu_h(1-r_1)}{2r_2}\right\}.$$

Then for ϵ_h and k_c such that

$$\epsilon_h \in (0,1) \quad \textit{and} \quad k_c \geq \frac{1+1/K_H}{1-r_1}\left(\log\left(\frac{3\mu_h}{\nu_U M_h \epsilon_h}\right) + 1\right),$$

we obtain $\|\mathbf{x}_{h,k_c} - \mathbf{x}_{h,\star}\| \leq \epsilon_h$. Note that M_h and μ_h are defined in Assumption 5.1; K_H is defined in Assumption 5.23; r_1 is defined in (5.20).

Proof By definition,

$$\epsilon_G \leq \frac{\mu_h(1-r_1)}{2r_2} \Rightarrow r_2\frac{\epsilon_G}{\mu_h} \leq \frac{1-r_1}{2} \quad \text{and} \quad r_1 + r_2\frac{\epsilon_G}{\mu_h} \leq \frac{1+r_1}{2}.$$

Also,

$$\epsilon_G \leq \frac{3\mu_h^2}{\nu_U M_h}(1-2\rho_1) \Rightarrow \frac{\epsilon_G}{\mu_h} \leq \frac{3\mu_h}{\nu_U M_h}(1-2\rho_1) \leq \frac{3\mu_h}{\nu_U M_h}.$$

Thus, using the results in Lemma 5.25, it is sufficient when

$$\begin{aligned} k_c &\geq \frac{1+1/K_H}{r_1-1}\left(\log\left(\frac{\nu_U M_h \epsilon_h}{3\mu_h}\right) - \log\left(\frac{1+r_1}{2}\right) + \frac{1-r_1}{2\log\left(\frac{1+r_1}{2}\right)}\right), \\ &= \frac{1+1/K_H}{1-r_1}\left(\log\left(\frac{3\mu_h}{\nu_U M_h \epsilon_h}\right) + \log\left(\frac{1+r_1}{2}\right) + \frac{r_1-1}{2\log\left(\frac{1+r_1}{2}\right)}\right). \end{aligned}$$

Since

$$\frac{r_1-1}{2\log\left(\frac{1+r_1}{2}\right)} < 1 \quad \text{and} \quad \log\left(\frac{1+r_1}{2}\right) < 0 \quad \text{for} \quad 0 < r_1 < 1,$$

it is sufficient when

$$k_c \geq \frac{1+1/K_H}{1-r_1}\left(\log\left(\frac{3\mu_h}{\nu_U M_h \epsilon_h}\right) + 1\right).$$

So we obtain the desired result. ∎

Lemma 5.26 provides a better picture of the convergence when composite rate holds. One can see that the number of iterations required, k_c, is clearly inverse proportional to $1 - r_1$.

Theorem 5.27 *Suppose Algorithm 5.1 is performed, Assumption 5.23 holds, and $k = k_s + k_c$, where k_s and k_c are defined in Lemma 5.24 and 5.25. Then we obtain the ϵ_h-accuracy $\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| \leq \epsilon_h$ with complexity*

$$\mathcal{O}\left(\frac{k_s+k_c}{1+1/K_H} n^3 + \frac{1/K_H(k_s+k_c)}{1+1/K_H} C_h\right),$$

where K_H and C_h are defined in Assumption 5.23.

Proof The total complexity is the number of iterations, $k_s + k_c$, multiply by the cost per iteration. Based on Assumption 5.23, $\frac{k_s+k_c}{1+1/K_H}$ coarse correction steps and $\frac{1/K_H(k_s+k_c)}{1+1/K_H}$ fine correction steps are taken. The computational cost of each coarse correction step and fine correction step is $\mathcal{O}(n^3)$ and $\mathcal{O}(C_h)$, respectively. ∎

5.4.3 Comparison: Newton v.s. Multilevel

Using the derived complexity results, we now compare the complexity of Newton's method and GAMA. We conclude this section by stating the condition for which GAMA has lower complexity.

Theorem 5.28 *Suppose Assumption 5.23 holds, then for sufficiently large enough N, the complexity of Algorithm 5.1 is lower than the complexity of Newton's method. In particular, if ϵ_G in Lemma 5.24 is chosen to be*

$$\epsilon_G \triangleq \min\left\{\frac{3\mu_h^2}{\nu_U M_h}(1-2\rho_1), \frac{\mu_h(1-r_1)}{2r_2}\right\},$$

then the complexity of Algorithm 5.1 is lower than the complexity of Newton's method when

$$r_1 \leq 1 - \frac{(K_H n^3 + C_h)(1+1/K_H)\left(\log\left(\frac{3\mu_h}{\nu_U M_h \epsilon_h}\right)+1\right)}{N^3(K_H+1)(k_d+k_q) - (K_H n^3 + C_h)k_s}, \tag{5.24}$$

for

$$N^3(K_H+1)(k_d+k_q) - (K_H n^3 + C_h)k_s > 0.$$

Note that ρ_1 is a user-defined parameter in Algorithm 5.1; $\nu_U = \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$; M_h and μ_h are defined in Assumption 5.1; K_H and C_h are defined in Assumption 5.23; r_1 and r_2 are defined in (5.20); k_d, k_q, and k_s are defined in Lemma 5.20, 5.21, and 5.24, respectively.

Proof When the complexity of Algorithm 5.1 is less than Newton's method, we have

$$\begin{aligned}
\frac{k_s+k_c}{1+1/K_H}n^3 + \frac{1/K_H(k_s+k_c)}{1+1/K_H}C_h &\leq (k_d+k_q)N^3,\\
k_s n^3 + k_c n^3 + \frac{1}{K_H}(k_s+k_c)C_h &\leq \left(1+\frac{1}{K_H}\right)(k_d+k_q)N^3,\\
k_s\left(n^3+\frac{C_h}{K_H}\right) + k_c\left(n^3+\frac{C_h}{K_H}\right) &\leq \left(1+\frac{1}{K_H}\right)(k_d+k_q)N^3,\\
\left(1+\frac{1}{K_H}\right)(k_d+k_q)N^3 - k_s\left(n^3+\frac{C_h}{K_H}\right) &\geq k_c\left(n^3+\frac{C_h}{K_H}\right).
\end{aligned}$$

From the first inequality we can see that it is satisfied when N is sufficiently large. Using the definition of k_c in Lemma 5.26, we obtain

$$\begin{aligned}
\frac{1+1/K_H}{1-r_1}\left(\log\left(\frac{3\mu_h}{\nu_U M_h \epsilon_h}\right)+1\right) &\leq \left(\frac{(K_H+1)N^3}{K_H n^3 + C_h}\right)(k_d+k_q) - k_s, \\
\frac{1}{1-r_1} &\leq \frac{\left(\frac{(K_H+1)N^3}{K_H n^3+C_h}\right)(k_d+k_q)-k_s}{(1+1/K_H)\left(\log\left(\frac{3\mu_h}{\nu_U M_h\epsilon_h}\right)+1\right)}, \\
1-r_1 &\geq \frac{(1+1/K_H)\left(\log\left(\frac{3\mu_h}{\nu_U M_h\epsilon_h}\right)+1\right)}{\left(\frac{(K_H+1)N^3}{K_H n^3+C_h}\right)(k_d+k_q)-k_s}, \\
r_1 &\leq 1 - \frac{(1+1/K_H)\left(\log\left(\frac{3\mu_h}{\nu_U M_h\epsilon_h}\right)+1\right)}{\left(\frac{(K_H+1)N^3}{K_H n^3+C_h}\right)(k_d+k_q)-k_s},
\end{aligned}$$

as required. ∎

Theorem 5.28 shows that when the dimension of the fine model, N, is sufficiently large, GAMA has lower computational complexity. The condition (5.24) requires a sufficiently small r_1 in the composite rate (5.20). This result agrees with the intuition with the following reasoning: when $r_1 \ll 1$, it implies that GAMA converges with very fast linear rate, which could outperform Newton's method because of the cheaper per-iteration cost.

We shall further study the condition (5.24). Assume the cost of fine correction step is at most in the same order of the coarse correction step, i.e. $C_h = \mathcal{O}(n^3)$. Then, by fixing all the quantities except N, the condition (5.24) can be recognized asymptotically as

$$r_1 \leq 1 - \mathcal{O}\left(\frac{1}{N^3}\right).$$

Thus, as $N \to \infty$, the above condition holds even when $r_1 \approx 1$. This condition is relaxed quickly because N grows in cube.

From equation (5.23), recall that $r_1 \ll 1$ is equivalent to

$$\|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|\ \|(\mathbf{I}-\mathbf{PR})(\mathbf{x}_{h,k}-\mathbf{x}_{h,\star})\| \ll \|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|.$$

From the above expression, one can see that a small $\|(\mathbf{I}-\mathbf{PR})(\mathbf{x}_{h,k}-\mathbf{x}_{h,\star})\|$ is equivalent to small r_1. In the following three sections, we will consider three cases of GAMA and derive the bounds for $\|(\mathbf{I}-\mathbf{PR})(\mathbf{x}_{h,k}-\mathbf{x}_{h,\star})\|$ for each case. In particular, we show how the magnitude of $\|(\mathbf{I}-\mathbf{PR})(\mathbf{x}_{h,k}-\mathbf{x}_{h,\star})\|$ varies depending on the structure of the problems and the parameters chosen.

5.5 PDE-based Problems: One-dimensional Case

In this section, we study the Galerkin model that arises from PDE-based problems. We begin with introducing the basic setting, and then we analyze the coarse correction step in this specific case. Building upon the composite rate in Section 5.3.4, at the end of this section we re-derive the composite rate with a more insightful bound of $\|(\mathbf{I}-\mathbf{PR})(\mathbf{x}_{h,k}-\mathbf{x}_{h,\star})\|$. As mentioned in Section 5.3, this quantity is critical in analyzing the performance and complexity of GAMA.

Since the conventional multigrid methods were originally developed for solving (non-)linear equations arising from PDEs, most research on multilevel optimization algorithms has been focusing on solving the discretizations of infinite-dimensional problems [GMS+10, GST08, KM16, LN05, Nas00, WG09]. As mentioned before, the Galerkin model in optimization was first mentioned in [GST08] and later tested numerically in [GMS+10]. We point out that from a theoretical perspective, the Galerkin model has been only considered as one special case of the general multilevel framework, and it has not been shown to have any particular advantage. For the trust-region based multilevel algorithm in [GST08], it has the same order of complexity bound as pure gradient descent. For the line-search based multilevel algorithm in [WG09], the convergence rate was proven to be sublinear for strongly convex problems, which agrees with our results in Section 5.3.

For the simplicity of the analysis, we consider specifically the one-dimensional case, i.e. the decision variable of the infinite-dimensional problems is a functional in $\mathbb{R}$. We further assume that the decision variable is discretized uniformly over $[0,1]$ with value 0 on the boundary. We would like to clarify that the approach of analysis in this section could be applied to more

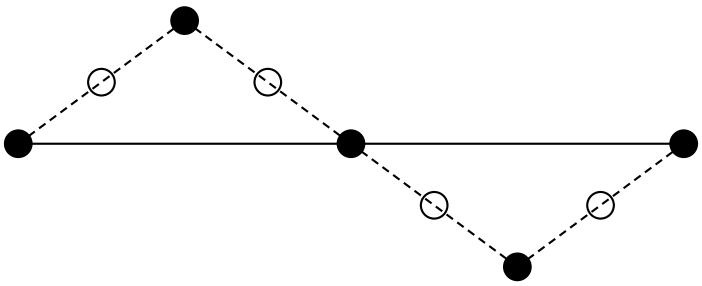


Figure 5.1: **P** in (5.25)

general and high dimensional settings.

5.5.1 Galerkin Model by One-dimensional Interpolations

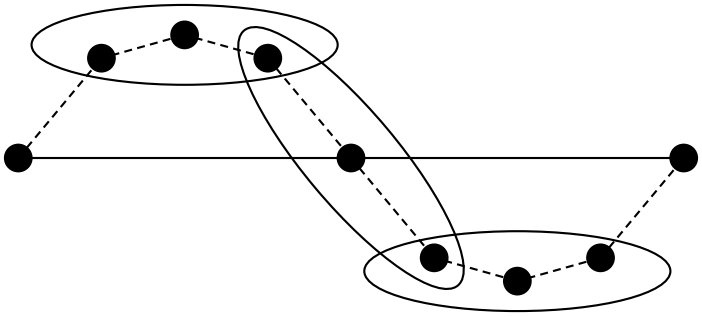


Figure 5.2: **R** in (5.26)

For one-dimensional problems, we consider the standard linear prolongation operator and restriction operator. Based on the traditional setting in multigrid research, we define the following Galerkin model.

- N is an even number,
- the (fine) discretized decision variable is in $\mathbb{R}^{N-1}$, and
- the coarse model is in $\mathbb{R}^{N/2-1}$.

For the interpolation operator $\mathbf{P} \in \mathbb{R}^{(N-1)\times(N/2-1)}$, we consider

$$\mathbf{P} = \frac{1}{2}\begin{pmatrix} 1 & & & & \\ 2 & & & & \\ 1 & 1 & & & \\ & 2 & & & \\ & 1 & & & \\ & & \ddots & 1 & \\ & & & 2 & \\ & & & 1 & \end{pmatrix}, \tag{5.25}$$

and the restriction operator

$$\mathbf{R} = \frac{1}{2}\mathbf{P}^T \in \mathbb{R}^{(N/2-1)\times(N-1)}. \tag{5.26}$$

Notice that the $\mathbf{P}$ and $\mathbf{R}$ in (5.25) and (5.26) have geometric meanings, and they are one of the standard pairs of operators in multilevel and multigrid methods [BHM00]. As shown in Figure 5.1, $\mathbf{P}$ is an interpolation operator such that one point is interpolated linearly between every two points. On the other hand, from Figure 5.2, $\mathbf{R}$ performs a restriction by doing weighted average onto every three points. These two operators assume the boundary condition is zero for both end points. We emphasize that the approach of convergence analysis in this section is not restricted for this specific pair of $\mathbf{P}$ and $\mathbf{R}$. We believe the general approach could be applied to interpolation type operators, especially operators that are designed for PDE-based optimization problems.

5.5.2 Analysis

With the definitions (5.25) and (5.26), we investigate the convergence behaviour of the coarse correction step. The analytical tool we use in this section is Taylor's expansion. To deploy this technique, we consider interpolations over the elements of vectors. In particular, we consider interpolations that are twice continuously differentiable with the following definition.

Definition 5.29 *For any vector* $\mathbf{r} \in \mathbb{R}^{N-1}$, *we denote* $\mathcal{F}_{\mathbf{r}}^{N-1}$ *to be the set of twice continuously differentiable functions such that* $\forall w \in \mathcal{F}_{\mathbf{r}}^{N-1}$,

$$w(0) = w(1) = 0, \quad \textit{and} \quad w_i = w(y_i) = (\mathbf{r})_i,$$

where $y_i = i/N$ *for* $i = 1, 2, \ldots, N-1$.

Using the definitions (5.25) and (5.26), we can estimate the "information loss" via interpolations using the following proposition.

Proposition 5.30 *Suppose* $\mathbf{P}$ *and* $\mathbf{R}$ *are defined in (5.25) and (5.26), respectively. For any vector* $\mathbf{r}_h \in \mathbb{R}^{N-1}$, *we denote* $(\mathbf{r}_h)_0 = (\mathbf{r}_h)_N = 0$ *and obtain*

$$(\mathbf{PRr}_h)_j = \begin{cases} \frac{1}{4}((\mathbf{r}_h)_{j-1} + 2(\mathbf{r}_h)_j + (\mathbf{r}_h)_{j+1}) & \textit{if } j \textit{ is even,} \\ \frac{1}{8}((\mathbf{r}_h)_{j-2} + 2(\mathbf{r}_h)_{j-1} + 2(\mathbf{r}_h)_j + 2(\mathbf{r}_h)_{j+1} + (\mathbf{r}_h)_{j+2}) & \textit{if } j \textit{ is odd,} \end{cases}$$

for $j = 1, 2, \ldots, N-1$.

Proof By the definition of $\mathbf{R}$ and $\mathbf{P}$, we have

$$(\mathbf{Rr}_h)_j = \frac{1}{4}((\mathbf{r}_h)_{2j-1} + 2(\mathbf{r}_h)_{2j} + (\mathbf{r}_h)_{2j+1}), \quad 1 \le j \le \frac{n}{2} - 1.$$

So

$$(\mathbf{PRr}_h)_j = (\mathbf{Rr}_h)_{j/2} = \frac{1}{4}((\mathbf{r}_h)_{j-1} + 2(\mathbf{r}_h)_j + (\mathbf{r}_h)_{j+1}) \quad \text{if } j \text{ is even,}$$

and

$$\begin{aligned} (\mathbf{PRr}_h)_j &= \frac{1}{2}\left((\mathbf{Rr}_h)_{(j-1)/2} + (\mathbf{Rr}_h)_{(j+1)/2}\right), \\ &= \frac{1}{8}((\mathbf{r}_h)_{j-2} + 2(\mathbf{r}_h)_{j-1} + 2(\mathbf{r}_h)_j + 2(\mathbf{r}_h)_{j+1} + (\mathbf{r}_h)_{j+2}) \quad \text{if } j \text{ is odd.} \end{aligned}$$

So we obtain the desired result. ∎

Using the above proposition and Taylor's expansion, we obtain the following lemma.

Lemma 5.31 *Suppose* $\mathbf{P}$ *and* $\mathbf{R}$ *are defined in (5.25) and (5.26), respectively. For any vector* $\mathbf{r}_h \in \mathbb{R}^{N-1}$,

$$\|(\mathbf{I}-\mathbf{PR})\mathbf{r}_h\|_\infty \leq \min_{w\in\mathcal{F}_{\mathbf{r}_h}^{N-1}} \max_{y\in[0,1]} |w''(y)| \frac{3}{4N^2}.$$

Note that the definition of $\mathcal{F}_{\mathbf{r}_h}^{N-1}$ *follows from Definition 5.29.*

Proof Using Proposition 5.30 and Taylor's Theorem, in the case that j is even, we obtain

$$\begin{aligned}
\frac{1}{4}((\mathbf{r}_h)_{j-1} + 2(\mathbf{r}_h)_j + (\mathbf{r}_h)_{j+1}) &= \frac{1}{4}\left(w\left(y_{j-1}\right) + 2w\left(y_j\right) + w\left(y_{j+1}\right)\right), \\
&= w\left(y_j\right) + \frac{w''(y_{c1})}{8}\frac{1}{N^2} + \frac{w''(y_{c2})}{8}\frac{1}{N^2}, \\
&= (\mathbf{r}_h)_j + \frac{w''(y_{c1}) + w''(y_{c2})}{8}\frac{1}{N^2},
\end{aligned}$$

where $w(\cdot) \in \mathcal{F}_{\mathbf{r}_h}^{N-1}$, $y_{j-1} \leq y_{c1} \leq y_j$, and $y_j \leq y_{c2} \leq y_{j+1}$. Similarly, in the case that j is odd, we have

$$\begin{aligned}
&\frac{1}{8}((\mathbf{r}_h)_{j-2} + 2(\mathbf{r}_h)_{j-1} + 2(\mathbf{r}_h)_j + 2(\mathbf{r}_h)_{j+1} + (\mathbf{r}_h)_{j+2}) \\
&\qquad = (\mathbf{r}_h)_j + \frac{4w''(y_{c3}) + 2w''(y_{c4}) + 2w''(y_{c5}) + 4w''(y_{c6})}{16}\frac{1}{N^2}, \quad (5.27)
\end{aligned}$$

where $y_{j-2} \leq y_{c3} \leq y_j$, $y_{j-1} \leq y_{c4} \leq y_j$, $y_j \leq y_{c5} \leq y_{j+1}$, and $y_j \leq y_{c6} \leq y_{j+2}$. Therefore,

$$\|(\mathbf{I}-\mathbf{PR})\mathbf{r}_h\|_\infty \leq \max_{y\in[0,1]} |w''(y)| \frac{3}{4N^2} \quad \text{for} \quad \forall w(\cdot) \in \mathcal{F}_{\mathbf{r}_h}^{N-1}.$$

So we obtain the desired result. ∎

Lemma 5.31 provides an upper bound of $\|(\mathbf{I}-\mathbf{PR})\mathbf{r}_h\|_\infty$, for any $\mathbf{r}_h \in \mathbf{R}^{N-1}$. This result can be used to derive the upper bound of $\|(\mathbf{I}-\mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$, where $\mathbf{r}_h = \mathbf{x}_{h,k} - \mathbf{x}_{h,\star}$. As we can see, if $|w''(y)| = \mathcal{O}(1)$, where $w \in \mathcal{F}_{\mathbf{r}_h}^{N-1}$, then $\|(\mathbf{I}-\mathbf{PR})\mathbf{r}_h\|_\infty = \mathcal{O}(N^{-2})$. This can be explained by the fact that when the mesh size is fine enough (i.e. large N), linear interpolation and restriction provide very good estimations of the fine model.

In the following lemma, we provide an upper bound of $|w''|$ in terms of the original vector $\mathbf{r}_h$. The idea is to specify the interpolation method in which we construct w, and we will use cubic spline in particular. Cubic spline is one of the standard interpolation methods, and the output interpolated function w satisfies the setting in Definition 5.29 and Lemma 5.31.

Lemma 5.32 *Suppose* $\mathbf{P}$ *and* $\mathbf{R}$ *are defined in (5.25) and (5.26), respectively. For any vector* $\mathbf{r}_h \in \mathbb{R}^{N-1}$*, we obtain*

$$\|(\mathbf{I} - \mathbf{PR})\mathbf{r}_h\|_\infty \leq \frac{9}{4N^2}\|\mathbf{A}\mathbf{r}_h\|_\infty,$$

where

$$\mathbf{A} = N^2 \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & \ddots & \ddots & \\ & & \ddots & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}.$$

Proof We follow the notation in Definition 5.29. For $w \in \mathcal{F}_{\mathbf{r}_h}^{N-1}$ that is constructed via cubic spline, in the interval (y_i, y_{i+1}), we have

$$w(y) = Aw_i + Bw_{i+1} + Cw_i'' + Dw_{i+1}'',$$

where

$$\begin{aligned} A &= \frac{y_{i+1} - y}{y_{i+1} - y_i}, \\ B &= \frac{y - y_i}{y_{i+1} - y_i}, \\ C &= \frac{1}{6}(A^3 - A)(y_{i+1} - y_i)^2, \\ D &= \frac{1}{6}(B^3 - B)(y_{i+1} - y_i)^2. \end{aligned}$$

It is known from [PTVF96] that

$$\frac{\mathrm{d}^2 w}{\mathrm{d}y^2} = Aw_i'' + Bw_{i+1}'' \tag{5.28}$$

and

$$\frac{y_i - y_{i-1}}{6} w''_{i-1} + \frac{y_{i+1} - y_{i-1}}{3} w''_i + \frac{y_{i+1} - y_i}{6} w''_{i+1} = \frac{w_{i+1} - w_i}{y_{i+1} - y_i} - \frac{w_i - w_{i-1}}{y_i - y_{i-1}} \tag{5.29}$$

for $i = 1, 2, \ldots, N-1$. Using the above equation (5.28), at the interval (y_i, y_{i+1}), we obtain

$$\begin{aligned} \left| \frac{\mathrm{d}^2 w}{\mathrm{d}y^2} \right| = \left| A w''_i + B w''_{i+1} \right| &= \left| \frac{y_{i+1} - y}{y_{i+1} - y_i} w''_i + \frac{y - y_i}{y_{i+1} - y_i} w''_{i+1} \right| \\ &\leq \left| \frac{y_{i+1} - y}{y_{i+1} - y_i} \right| |w''_i| + \left| \frac{y - y_i}{y_{i+1} - y_i} \right| |w''_{i+1}| \\ &\leq \max\{|w''_i|, |w''_{i+1}|\}. \end{aligned}$$

Suppose $j \in \arg\max_i \{|w''_i|\}_i$, then from (5.29) and the fact that $y_{j+1} - y_j = 1/N$,

$$\begin{aligned} \frac{y_{j+1} - y_{j-1}}{3} w''_j &= \frac{w_{j+1} - w_j}{y_{j+1} - y_j} - \frac{w_j - w_{j-1}}{y_j - y_{j-1}} - \frac{y_j - y_{j-1}}{6} w''_{j-1} - \frac{y_{j+1} - y_j}{6} w''_{j+1}, \\ \frac{2}{3N} w''_j &= N(w_{j+1} - w_j) - N(w_j - w_{j-1}) - \frac{1}{6N} w''_{j-1} - \frac{1}{6N} w''_{j+1}, \\ 2 w''_j &= 3N^2 (w_{j+1} - 2w_j + w_{j-1}) - \frac{1}{2} w''_{j-1} - \frac{1}{2} w''_{j+1}. \end{aligned}$$

Thus,

$$\begin{aligned} |2w''_j| &\leq 3N^2 |w_{j+1} - 2w_j + w_{j-1}| + \frac{1}{2}|w''_{j-1}| + \frac{1}{2}|w''_{j+1}|, \\ 2|w''_j| &\leq 3N^2 |w_{j+1} - 2w_j + w_{j-1}| + \frac{1}{2}|w''_j| + \frac{1}{2}|w''_j|, \\ |w''_j| &\leq 3N^2 |w_{j+1} - 2w_j + w_{j-1}|. \end{aligned}$$

Therefore,

$$|w''_i| \leq \max_i 3N^2 |w_{i+1} - 2w_i + w_{i-1}|,$$

and so,

$$\|(\mathbf{I} - \mathbf{PR})\mathbf{r}_h\|_\infty \leq \max_{y \in [0,1]} |w''(y)| \frac{3}{4N^2} \leq \max_i \frac{9|w_{i+1} - 2w_i + w_{i-1}|}{4} = \frac{9}{4N^2} \|\mathbf{A}\mathbf{r}_h\|_\infty, \tag{5.30}$$

as required. ∎

Lemma 5.32 provides the discrete version of the result presented in Lemma 5.31. The matrix $\mathbf{A}$ is the discretized Laplacian operator, which is equivalent to twice differentiation using finite difference with a uniform mesh.

5.5.3 Convergence

With all the results, we revisit the composite convergence rate with the following corollary.

Corollary 5.33 *Suppose* $\mathbf{P}$ *and* $\mathbf{R}$ *are defined in (5.25) and (5.26), respectively. If the coarse correction step* $\hat{\mathbf{d}}_{h,k}$ *in (5.15) is taken with* $\alpha_{h,k} = 1$*, then*

$$\begin{aligned}\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| &\leq \sqrt{\frac{L_h}{\mu_h}} \min_{w \in \mathcal{F}^{N-1}_{\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}}} \max_{y\in[0,1]} |w''(y)| \frac{3}{4N^{3/2}} + \frac{M_h \nu_U^2 \nu_L^2}{2\mu_h} \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2, \\ &\leq \frac{9}{4N^{3/2}} \sqrt{\frac{L_h}{\mu_h}} \|\mathbf{A}(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\| + \frac{M_h \nu_U^2 \nu_L^2}{2\mu_h} \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2,\end{aligned}$$

where $\mathbf{A}$ *is defined in Lemma 5.32. Note that* M_h*,* L_h*, and* μ_h *are defined in Assumption 5.1,* $\nu_U = \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$*, and* $\nu_L = \|\mathbf{P}^+\|$*.*

Proof

$$\begin{aligned}\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| &\leq \|\mathbf{I} - \mathbf{P}[\nabla^2_H f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\| \|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\| \\ &\quad + \frac{M_h \nu_U^2 \nu_L^2}{2\mu_h} \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2 \qquad \text{(from Theorem 5.18)} \\ &\leq \sqrt{\frac{L_h}{\mu_h}} \min_{w \in \mathcal{F}^{N-1}_{\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}}} \max_{y\in[0,1]} |w''(y)| \frac{3}{4N^{3/2}} + \frac{M_h \nu_U^2 \nu_L^2}{2\mu_h} \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2 \\ &\qquad \text{(from Lemma 5.19 and Lemma 5.31)} \\ &\leq \frac{9}{4N^{3/2}} \sqrt{\frac{L_h}{\mu_h}} \|\mathbf{A}(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\| + \frac{M_h \nu_U^2 \nu_L^2}{2\mu_h} \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2 \qquad \text{(from (5.30)),}\end{aligned}$$

as required. ∎

Corollary 5.33 provides the convergence of using Galerkin model for PDE-based problems that we considered. This result shows the complementarity of the fine correction step and the coarse

correction step. Suppose the fine correction step can effectively reduce $\|\mathbf{A}(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$, then the coarse correction step could yield a major reduction based on the result shown in Corollary 5.33.

5.6 Low Rank Approximation using Nyström Method

In this section, we focus on the Galerkin model that is based on a low rank approximation of the Hessian matrix. We begin with an introduction of low rank approximation and the Nyström method. Then we make the connection between the Nyström method and the coarse correction step in (5.15). Finally, we re-derive the composite rate with more insightful bounds of both $\|\mathbf{I} - \mathbf{P}[\nabla^2_H f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|$ and $\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$.

Before introducing the obscure connection between low rank approximation and Galerkin model, let's start with the setting and consider a symmetric positive semi-definite matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$. The best low rank approximation of $\mathbf{A}$ with rank q can be obtained by solving the following optimization problem

$$\min_{\mathbf{A}_q \in \mathbb{R}^{N\times N}} \|\mathbf{A} - \mathbf{A}_q\|, \quad \text{s.t. } \operatorname{rank}(\mathbf{A}_q) = q. \tag{5.31}$$

It is known that the above problem can be solved via eigendecomposition. However, eigendecomposition is computationally expensive. In the context of optimization, the cost for each iteration of Newton's method is not more expensive than performing eigendecomposition. If a Galerkin model is constructed via eigendecomposition, one could apply Newton's method instead.

Although computing the exact solution of (5.31) is unfavorable, we could seek for its approximation. Nyström method was originally developed to numerically approximate eigenfunctions, and the idea was applied later in the machine learning community for the low rank optimization problem [WS01]. It provides a suboptimal solution of the low rank approximation with cheaper computational cost.

Nyström method is performed by the column selection procedure. Consider a set $\mathcal{Q} = \{1, 2, \ldots, N\}$,

and suppose a subset $\mathcal{Q}_1 \subseteq \mathcal{Q}$ with n elements. We denote q_i as the i^{th} element of $\mathcal{Q}_1$, for $i = 1, 2, \ldots, n$. Then one can approximate $\mathbf{A} \in \mathbb{R}^{N\times N}$ using the following procedures.

1. Define a matrix $\mathbf{A}_1 \in \mathbb{R}^{n\times N}$ such that the i^{th} row of $\mathbf{A}_1$ is the q_i^{th} row of $\mathbf{A}$.
2. Define a matrix $\mathbf{A}_2 \in \mathbb{R}^{N\times n}$ such that the i^{th} column of $\mathbf{A}_2$ is the q_i^{th} column of $\mathbf{A}$.
3. Define a matrix $\mathbf{A}_3 \in \mathbb{R}^{n\times n}$ such that $(\mathbf{A}_3)_{i,j}$ is the element of $\mathbf{A}$ in q_i^{th} row and q_j^{th} column.
4. Compute the pseudo-inverse $\mathbf{A}_3^+$.
5. Compute the low rank approximation of $\mathbf{A}$ by $\mathbf{A}_2\mathbf{A}_3^+\mathbf{A}_1$.

Equivalently, the above procedure can be described by using a matrix $\mathbf{S} \in \mathbb{R}^{N\times n}$ such that the i^{th} column of $\mathbf{S}$ is the q_i^{th} column of the identity matrix $\mathbf{I}$. The output of the above procedure is the same as

$$\mathbf{A} \approx \mathbf{A}_2\mathbf{A}_3^+\mathbf{A}_1 = \mathbf{AS}[\mathbf{S}^T\mathbf{AS}]^+\mathbf{S}^T\mathbf{A}. \tag{5.32}$$

Much research have been focused on developing Nyström method based on different methods on selecting the subset $\mathcal{Q}_1$ [DM05, Git13, SS00, WS01]. In this chapter, we consider the naïve Nyström method in which elements in $\mathcal{Q}_1$ are selected uniformly without replacement from $\mathcal{Q}$.

5.6.1 Galerkin Model by Naïve Nyström Method

Now we are in the position to show how Nyström method can be used to develop Galerkin model. The approximation (5.32) is highly similar to the coarse correction step in a multilevel algorithm.

Definition 5.34 *Consider a set $\mathcal{Q} = \{1, 2, \ldots, N\}$, and an n elements subset $\mathcal{Q}_1$ in which the elements are selected randomly, and uniformly without replacement from $\mathcal{Q}$. Denote q_i as the i^{th} element of $\mathcal{Q}_1$. Then the prolongation operator, $\mathbf{P} \in \mathbb{R}^{N\times n}$, and restriction operator, $\mathbf{R} \in \mathbb{R}^{n\times N}$, are generated using naïve Nyström method if*

i. The i^{th} column of $\mathbf{P}$ is the q_i^{th} column of the identity matrix $\mathbf{I}$.

ii. $\mathbf{R} = \mathbf{P}^T$.

Definition 5.34 defines the prolongation and restriction operators that are based on naïve Nyström method. One can see the analogy by substituting $\mathbf{S} = \mathbf{P}$, $\mathbf{S}^T = \mathbf{R}$, and $\mathbf{A} = \nabla^2 f_{h,k}$ in equation (5.32). Under the setting of naïve Nyström method, $\mathbf{P}$ is full column rank, and so Assumption 5.3 is satisfied. Moreover, different from the assumption that $\mathbf{A}$ is positive semi-definite in (5.32), $\nabla^2 f_{h,k}$ is positive definite as stated in Assumption 5.1, and so it is guaranteed to be invertible. Consider the low rank approximation (5.32) with $\mathbf{S} = \mathbf{P}$, $\mathbf{S}^T = \mathbf{R}$, and $\mathbf{A} = \nabla^2 f_{h,k}$. Multiplying $\nabla^2 f_{h,k}^{-1}$ from both left and right yields

$$\nabla^2 f_{h,k}^{-1} \approx \mathbf{P}[\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}]^{+}\mathbf{R} = \mathbf{P}[\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}]^{-1}\mathbf{R},$$

and so

$$-\nabla^2 f_{h,k}^{-1}\nabla f_{h,k} \approx -\mathbf{P}[\mathbf{R}\nabla^2 f_{h,k}\mathbf{P}]^{-1}\mathbf{R}\nabla f_{h,k} = \hat{\mathbf{d}}_{h,k}.$$

Thus, the coarse correction step $\hat{\mathbf{d}}_{h,k}$ is an approximation of Newton step. We emphasize that naïve Nyström method is effective in practice, and computationally inexpensive to perform (uniform sampling without replacement).

It is worth mentioning that the coarse correction step is highly related to block-coordinate descent algorithms. In fact, $\mathbf{P}$ and $\mathbf{R}$ from Definition 5.34 can be used to derive block-coordinate descent algorithms, as described in Section 5.2.4. The coarse correction step in this section is different from first order block-coordinate descent type methods because GAMA uses the Hessian $\nabla^2 f_{h,k}$ instead of the identity matrix in the coarse model (5.11).

Interestingly, similar works have been done from the perspective of block coordinate methods for machine learning problems. In particular, Gower et al. [GGR16] recently developed a stochastic block BFGS for solving objective functions that have the form of sum of functions. The coarse correction step we study in this section is a special case of the stochastic block BFGS: when the previous approximated inverse Hessian is set to zero and when all functions

(in the summation) are used to compute Hessians. On the other hand, the proposed coarse correction step is also studied by Qu et al. [QRTF15] for the dual formulation of empirical risk minimization. In both cases, they provided (expected) linear convergence rates. Moreoever, due to different sources of motivation, they did not mention that Nyström is used inherently within the search direction.

5.6.2 Analysis

We are now in the position to analyze the two important factors in the composition convergence rate, $\|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|$ and $\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$. The analytical tool we use is concentration inequality. The following Chernoff bounds will be used to analyze $\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$.

Theorem 5.35 ([Tro11]) *Let $\mathcal{Q}$ be a finite set of positive numbers, and suppose*

$$\max_{q\in\mathcal{Q}} q \leq B.$$

Sample $\{q_1, q_2, \ldots, q_l\}$ uniformly at random from $\mathcal{Q}$ without replacement. Compute

$$s = l \cdot \mathbb{E}(q_1).$$

Then

$$\mathbb{P}\left\{\sum_j q_j \leq (1-\sigma)s\right\} \leq \left(\frac{e^{-\sigma}}{(1-\sigma)^{1-\sigma}}\right)^{s/B} \qquad \textit{for} \quad \sigma \in [0,1), \quad \textit{and}$$
$$\mathbb{P}\left\{\sum_j q_j \geq (1+\sigma)s\right\} \leq \left(\frac{e^{\sigma}}{(1+\sigma)^{1+\sigma}}\right)^{s/B} \qquad \textit{for} \quad \sigma \geq 0.$$

Proof See Theorem 2.1 from Tropp [Tro11]. ∎

Theorem 5.35 is useful to derive statistical bounds for $\|(\mathbf{I} - \mathbf{PR})\mathbf{r}_h\|$, for any $\mathbf{r}_h \in \mathbb{R}^N$. The results are provided in the following lemma.

Lemma 5.36 *Suppose the prolongation operator* $\mathbf{P} \in \mathbb{R}^{N\times n}$ *and the restriction operator* $\mathbf{R} \in \mathbb{R}^{n\times N}$ *are generated using the naïve Nyström method according to Definition 5.34 and* $\mathbf{r}_h \in \mathbb{R}^N$. *Then* $\forall \sigma \in [0,1)$, *we obtain*

$$\mathbb{P}\left\{\|(\mathbf{I}-\mathbf{PR})\mathbf{r}_h\| \leq \sqrt{(1-\sigma)\frac{N-n}{N}}\|\mathbf{r}_h\|\right\} \leq \left(\frac{e^{-\sigma}}{(1-\sigma)^{1-\sigma}}\right)^{\frac{N-n}{N}\|\mathbf{r}_h\|^2/\|\mathbf{r}_h\|_\infty^2},$$

and $\forall \sigma \geq 0$, *we obtain*

$$\mathbb{P}\left\{\|(\mathbf{I}-\mathbf{PR})\mathbf{r}_h\| \geq \sqrt{(1+\sigma)\frac{N-n}{N}}\|\mathbf{r}_h\|\right\} \leq \left(\frac{e^{\sigma}}{(1+\sigma)^{1+\sigma}}\right)^{\frac{N-n}{N}\|\mathbf{r}_h\|^2/\|\mathbf{r}_h\|_\infty^2}.$$

Proof We denote $\mathcal{Q} = \{1, 2, \ldots, N\}$ to be a set of indices, a subset $\mathcal{Q}_1 \subset \mathcal{Q}$ such that

$$\text{range}(\mathbf{P}) = \text{span}\left(\{\mathbf{e}_j : j \in \mathcal{Q}_1\}\right),$$

and the complement $\mathcal{Q}_2 = \mathcal{Q} \setminus \mathcal{Q}_1$.

These definitions lead to

$$\|(\mathbf{I}-\mathbf{PR})\mathbf{r}_h\|^2 = \sum_{j\in\mathcal{Q}_2}(\mathbf{r}_h)_j^2,$$

since $\mathcal{Q}_2$ is a set of indices that are associated with the selected coordinates in $\mathbf{I}-\mathbf{PR}$. Therefore, $\mathcal{Q}_2$ contains $N-n$ samples from $\mathcal{Q}$ that are distributed uniformly without replacement. By applying Theorem 5.35, we obtain

$$\max_{j\in\mathcal{Q}}(\mathbf{r}_h)_j^2 = \|\mathbf{r}_h\|_\infty^2,$$

and

$$s = (N-n)\frac{1}{N}\sum_{j\in\mathcal{Q}}(\mathbf{r}_h)_j^2 = \frac{N-n}{N}\|\mathbf{r}_h\|^2.$$

By direct substitutions, we obtain the desired result. ∎

Lemma 5.36 provides bounds for $\|(\mathbf{I}-\mathbf{PR})\mathbf{r}_h\|$, for any $\mathbf{r}_h \in \mathbb{R}^N$. On the other hand, we bear in mind that Nyström method is a method of computing low rank approximations. In the following lemma, we will show that this feature is shown in the bound of $\|\mathbf{I}-\mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|$.

Lemma 5.37 *Suppose the prolongation operator $\mathbf{P} \in \mathbb{R}^{N\times n}$ and the restriction operator $\mathbf{R} \in \mathbb{R}^{n\times N}$ are generated using the naïve Nyström method according to Definition 5.34. For $p \in \{1, 2, \ldots, N\}$, let the eigendecomposition of $\nabla^2 f_{h,k}$ have the following form*

$$\nabla^2 f_{h,k} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T = \begin{pmatrix} \mathbf{U}_1 & \mathbf{U}_2 \end{pmatrix} \begin{pmatrix} \mathbf{\Sigma}_1 & \\ & \mathbf{\Sigma}_2 \end{pmatrix} \begin{pmatrix} \mathbf{U}_1^T \\ \mathbf{U}_2^T \end{pmatrix},$$

where $\mathbf{\Sigma}_1 \in \mathbb{R}^{p\times p}$, $\mathbf{\Sigma}_2 \in \mathbb{R}^{(N-p)\times(N-p)}$, $\mathbf{U}_1 \in \mathbb{R}^{N\times p}$, and $\mathbf{U}_2 \in \mathbb{R}^{N\times(N-p)}$ are the sub-matrices of $\mathbf{\Sigma}$ and $\mathbf{U}$. Denote τ as the coherence of $\mathbf{U}_1$,

$$\tau \triangleq \frac{N}{p} \max_i (\mathbf{U}_1\mathbf{U}_1^T)_{ii}.$$

Then, for β, σ and n such that

$$\beta, \sigma \in (0, 1), \quad \text{and} \quad n \geq \frac{2\tau p \log\left(\frac{p}{\beta}\right)}{(1-\sigma)^2},$$

we obtain

$$\|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\| \leq \sqrt{\frac{\lambda_{p+1}(\nabla^2 f_{h,k})}{\mu_h}\left(1 + \frac{N}{n\sigma}\right)},$$

with probability at least $1-\beta$. Note that $\lambda_{p+1}(\nabla^2 f_{h,k})$ is the $p+1^{th}$ largest eigenvalue of $\nabla^2 f_{h,k}$.

Proof Following from Lemma 5.19, we have

$$\begin{aligned} \|\mathbf{I} - \mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\| &= \|\mathbf{U}\mathbf{\Sigma}^{-1/2}(\mathbf{I} - \mathbf{\Gamma}_{\mathbf{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\mathbf{\Sigma}^{1/2}\mathbf{U}^T\|, \\ &\leq \|\mathbf{U}\mathbf{\Sigma}^{-1/2}\|\|(\mathbf{I} - \mathbf{\Gamma}_{\mathbf{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\mathbf{\Sigma}^{1/2}\mathbf{U}^T\|, \\ &\leq \sqrt{\frac{1}{\mu_h}}\|(\mathbf{I} - \mathbf{\Gamma}_{\mathbf{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\mathbf{\Sigma}^{1/2}\mathbf{U}^T\|. \end{aligned}$$

Using results from Gittens [Git11], Theorem 2,

$$\|(\mathbf{I} - \mathbf{\Gamma}_{\mathbf{\Sigma}^{1/2}\mathbf{U}^T\mathbf{P}})\mathbf{\Sigma}^{1/2}\mathbf{U}^T\| \leq \sqrt{\lambda_{p+1}(\nabla^2 f_{h,k})\left(1 + \frac{N}{n\sigma}\right)},$$

with probability at least $1-\beta$. ∎

In addition to Lemma 5.19, Lemma 5.37 provides a new alternative for bounding the term $\|\mathbf{I}-\mathbf{P}[\nabla_H^2 f_{h,k}]^{-1}\mathbf{R}\nabla^2 f_{h,k}\|$. This is a direct result from the fact that Nyström is used inherently with the $\mathbf{P}$ and $\mathbf{R}$ in Definition 5.34. As we will show later, this result would improve the convergence rate if the Hessian can be well-approximated using low rank approximation.

As mentioned in [Git11, CR09], we would like to point out that the coherence τ defined in Lemma 5.19 ranges from 1 to N/p. For an $N \times p$ random orthogonal matrix in which its columns are selected uniformly among all families of p orthonormal vectors, its coherence is bounded by $\mathcal{O}(\max\{p, \log N\}/p)$ with high probability [CR09].

5.6.3 Convergence

Using the above results, we obtain the following corollaries.

Corollary 5.38 *Suppose $\mathbf{P} \in \mathbb{R}^{N\times n}$ and $\mathbf{R} \in \mathbb{R}^{n\times N}$ are generated using the naïve Nyström method according to Definition 5.34, and τ is the coherence as defined in Lemma 5.37. If the coarse correction step $\hat{\mathbf{d}}_{h,k}$ is taken with $\alpha_{h,k} = 1$, then $\forall \sigma_2 \geq 0$,*

$$\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| \leq \sqrt{\frac{L_h}{\mu_h}(1+\sigma_2)\frac{N-n}{N}}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| + \frac{M_h \nu_U^2 \nu_L^2}{2\mu_h}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2,$$

with probability at least

$$1 - (\Phi(\sigma_2))^{\frac{N-n}{N}\|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|^2/\|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|_\infty^2} \quad \textit{for} \quad \Phi(\sigma_2) = \frac{e^{\sigma_2}}{(1+\sigma_2)^{1+\sigma_2}}. \tag{5.33}$$

Note that L_h, M_h, and μ_h are defined in Assumption 5.1, $\nu_U = \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$ and $\nu_L = \|\mathbf{P}^+\|$.

Proof The result can be obtained by combining results from Lemma 5.36 with $\mathbf{r}_h = \mathbf{x}_{h,k} - \mathbf{x}_{h,\star}$, Lemma 5.19, and Theorem 5.18. ∎

Corollary 5.38 provides the probabilistic composite convergence rate. As expected, the coefficient of the linear component goes to 0 as $n \to N$. We point out that when $n = N$, the probability in (5.33) is equal to zero since $(N - n)/N = 0$. Thus, Corollary 5.38 is not meaningful at the exact limit of $n = N$. However, in this case, no dimension is reduced, and so based on Theorem 5.18, the quadratic convergence is obtained.

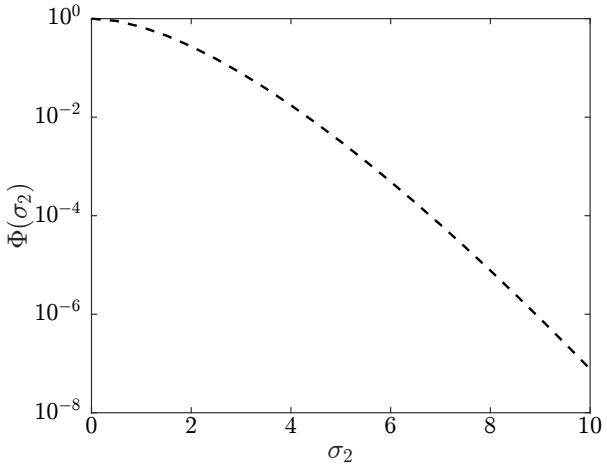


Figure 5.3: $\Phi(\sigma_2)$ in (5.33)

Figure 5.3 shows the value of $\Phi(\sigma_2)$ in (5.33), and one can see that with reasonably small σ_2, $\Phi(\sigma_2) \ll 1$. Also, since $\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2$ is the sum of squares of the error in each dimension, it is reasonable to expect that it is in $\mathcal{O}(N)$. Therefore, one could expect that

$$\frac{N-n}{N} \frac{\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2}{\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|_\infty^2} \sim \mathcal{O}(N-n),$$

and so for $n < N$, the power coefficient above should reduce $\Phi(\sigma_2)$ further.

While Corollary 5.38 illustrates how $\|(\mathbf{I} - \mathbf{PR})(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$ varies with respect to n, it does not show that using the prolongation and restriction operators that are inspired by Nyström method has any advantage when Hessians have the low rank structure. By combining result in Lemma 5.37, we obtain the following corollary.

Corollary 5.39 *Suppose* $\mathbf{P} \in \mathbb{R}^{N \times n}$ *and* $\mathbf{R} \in \mathbb{R}^{n \times N}$ *are generated using naïve Nyström method according to Definition 5.34, and* τ *is the coherence as defined in Lemma 5.37. If the coarse*

correction step $\hat{\mathbf{d}}_{h,k}$ *is taken with* $\alpha_{h,k} = 1$*, then* $\forall \beta, \sigma_1\sigma_2, p, n$ *such that*

$$\beta, \sigma_1 \in (0,1), \quad \sigma_2 \geq 0, \quad p \in \{1, 2, \ldots, N\}, \quad \textit{and} \quad n \geq \frac{2\tau p \log\left(\frac{p}{\beta}\right)}{(1-\sigma_1)^2},$$

we obtain

$$\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| \leq \sqrt{\frac{\lambda_{p+1}(\nabla^2 f_{h,k})}{\mu_h}\left(1 + \frac{N}{n\sigma_1}\right)(1+\sigma_2)\frac{N-n}{N}}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\| + \frac{M_h \nu_U^2 \nu_L^2}{2\mu_h}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2,$$

with probability at least

$$(1-\beta)\left(1 - (\Phi(\sigma_2))^{\frac{N-n}{N}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2 / \|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|_\infty^2}\right) \quad \textit{for} \quad \Phi(\sigma_2) = \frac{e^{\sigma_2}}{(1+\sigma_2)^{1+\sigma_2}}.$$

Note that L_h*,* M_h*, and* μ_h *are defined in Assumption 5.1;* $\nu_U = \max\{\|\mathbf{P}\|, \|\mathbf{R}\|\}$ *and* $\nu_L = \|\mathbf{P}^+\|$*;* $\lambda_{p+1}(\nabla^2 f_{h,k})$ *is the* $p+1^{th}$ *largest eigenvalue of* $\nabla^2 f_{h,k}$*.*

Proof The result can be obtained by combining results from Lemma 5.36 with $\mathbf{r}_h = \mathbf{x}_{h,k} - \mathbf{x}_{h,\star}$, Lemma 5.37, and Theorem 5.18. ∎

Compared to Corollary 5.38, Corollary 5.39 replaces the largest eigenvalue of $\nabla^2 f_{h,k}$, L_h, with the scaled $p+1^{\text{th}}$ largest eigenvalue, $\lambda_{p+1}(\nabla^2 f_{h,k})$, with high probability. It provides a clear advantage when there is a large gap between the p^{th} and $p+1^{\text{th}}$ eigenvalue, in particular, when

$$\mu_h \leq \lambda_N(\nabla^2 f_{h,k}) \leq \cdots \leq \lambda_{p+1}(\nabla^2 f_{h,k}) \ll \lambda_p(\nabla^2 f_{h,k}) \leq \lambda_1(\nabla^2 f_{h,k}) \leq L_h.$$

We point out that the concentration inequality is not only useful for getting a composite convergence rate, but also useful for bounding the parameter κ in Algorithm 5.1.

Lemma 5.40 *Suppose the prolongation operator* $\mathbf{P} \in \mathbb{R}^{N\times n}$ *and the restriction operator* $\mathbf{R} \in \mathbb{R}^{n\times N}$ *are generated using the naïve Nyström method according to Definition 5.34. Then* $\forall \mathbf{r}_h \in$

$\mathbb{R}^N, \forall \sigma \in [0,1)$, *we obtain*

$$\mathbb{P}\left\{\|\mathbf{PR}\mathbf{r}_{h,k}\| \leq \sqrt{(1-\sigma)\frac{n}{N}}\|\mathbf{r}_{h,k}\|\right\} \leq \left(\frac{e^{-\sigma}}{(1-\sigma)^{1-\sigma}}\right)^{\frac{n}{N}\|\mathbf{r}_{h,k}\|^2/\|\mathbf{r}_{h,k}\|_\infty^2},$$

and $\forall \sigma \geq 0$, *we have*

$$\mathbb{P}\left\{\|\mathbf{PR}\mathbf{r}_{h,k}\| \geq \sqrt{(1+\sigma)\frac{n}{N}}\|\mathbf{r}_{h,k}\|\right\} \leq \left(\frac{e^{\sigma}}{(1+\sigma)^{1+\sigma}}\right)^{\frac{n}{N}\|\mathbf{r}_{h,k}\|^2/\|\mathbf{r}_{h,k}\|_\infty^2}.$$

Proof The proof is exactly the same as in Lemma 5.36 with consideration of $\mathcal{Q}_1$ as a sample set instead. ∎

Lemma 5.40 provides the fact that with high probability

$$\|\mathbf{R}\nabla f_{h,k}\| = \|\mathbf{PR}\nabla f_{h,k}\| \geq \mathcal{O}\left(\sqrt{\frac{n}{N}}\right)\|\nabla f_{h,k}\|.$$

Note that in the analysis in Section 5.3, when the coarse correction step is taken, we assume $\|\mathbf{R}\nabla f_{h,k}\| > \kappa\|\nabla f_{h,k}\|$ for some constant κ. As stated in Lemma 5.10 and Theorem 5.13, the square of this κ is proportional to Λ, which is inversely proportional to the rate of convergence. Therefore, we shall conclude that in the setting considered in this section, with high probability the rate of convergence is inversely proportional to $\mathcal{O}(n/N)$, or equivalently, proportional to $\mathcal{O}(N/n)$.

5.7 Block Diagonal Approximation

In this section, we focus on the case that the Hessian $\nabla^2 f_{h,k}$ is approximated by block diagonal approximation. The structure of this section is similar to the last two sections: we introduce and formally define block diagonal approximation, perform analysis, and finally re-derive the composite rate in this setting.

Definition 5.41 *Suppose* $\nabla^2 f_{h,k} \in \mathbb{R}^{N\times N}$ *and* $n_1, n_2, \ldots, n_q \in \mathbb{N}$ *such that* $n_1 + n_2 + \cdots + n_q = N$. *Then the q-block diagonal approximation of* $\nabla^2 f_{h,k}$ *is defined as* $\nabla^2_B f_{h,k}$ *where*

$$(\nabla^2_B f_{h,k})_{i,j} = \begin{cases} (\nabla^2 f_{h,k})_{i,j} & \text{if} \quad \sum_{p=1}^{m-1} n_p < i,j \leq \sum_{p=1}^{m} n_p, \quad \text{for any} \;\; m = 1,2,\ldots,q, \\ 0 & \text{otherwise.} \end{cases}$$

Definition 5.41 states the formal definition of block diagonal approximation of a Hessian. That is, we only preserve the elements which are located in block diagonal positions, and set all the other elements to zeros. Recall that even though Newton's method is one of the best algorithms with quadratic convergence rate, the trade-off, however, goes into the high computational cost at each iteration: solving an N-by-N system of linear equations. By replacing the Hessian with its q-block diagonal approximation, the corresponding N-by-N system of linear equations can be decomposed by q smaller systems of linear equations, and thus lower computational cost is required.

The above block diagonal approximation approach is a special case of the incomplete Hessian Newton minimization method proposed by Xie and Ni [XN09]. In the case where $n_1 = n_2 = \cdots = n_N = 1$, this diagonal approximation is also considered in [FT15]. While it is clear that the block diagonal approximation contains partial second order information and one should expect that it performs better than first order algorithms, no theoretical indication has pointed in this direction.

5.7.1 Multiple Galerkin Models

We will show that q-block diagonal approximation from Definition 5.41 could be formulated using multiple Galerkin models. We denote the prolongation operators as $\mathbf{P}_i \in \mathbb{R}^{N\times n_i}$, for $i = 1, 2, \ldots, q$. Notice that $n_1 + n_2 + \cdots + n_q = N$, and we assume

$$[\mathbf{P}_1 \; \mathbf{P}_2 \ldots \mathbf{P}_q] = \mathbf{I}.$$

We also denote the corresponding restriction operators as $\mathbf{R}_i = \mathbf{P}_i^T$, for $i = 1, 2, \ldots, q$. Then, the block diagonal approximation can be expressed as

$$\nabla^2_{\mathrm{B}} f_{h,k} = \mathrm{diag}(\mathbf{R}_1 \nabla^2 f_{h,k} \mathbf{P}_1, \mathbf{R}_2 \nabla^2 f_{h,k} \mathbf{P}_2, \ldots, \mathbf{R}_q \nabla^2 f_{h,k} \mathbf{P}_q),$$

and the corresponding coarse correction step is defined as

$$\hat{\mathbf{d}}_{h,k} = -[\nabla^2_{\mathrm{B}} f_{h,k}]^{-1} \nabla f_{h,k} = \sum_{i=1}^{q} -\mathbf{P}_i [\mathbf{R}_i \nabla^2 f_{h,k} \mathbf{P}_i] \mathbf{R}_i \nabla f_{h,k}. \tag{5.34}$$

5.7.2 A Counterexample for General Functions

We start with a counterexample to show that it is impossible to be as good as the classical Newton's method for general functions in term of convergence. Suppose we have the following problem

$$\min_{\mathbf{x}_h \in \mathbb{R}^2} f_h(\mathbf{x}_h) \triangleq \frac{1}{2} \mathbf{x}_h^T \begin{pmatrix} 1 & -\sqrt{0.5} \\ -\sqrt{0.5} & 1 \end{pmatrix} \mathbf{x}_h + \mathbf{x}_h^T \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

The above quadratic program (QP) has the positive definite Hessian

$$\begin{pmatrix} 1 & -\sqrt{0.5} \\ -\sqrt{0.5} & 1 \end{pmatrix} = \begin{pmatrix} 1 \\ -\sqrt{0.5} \end{pmatrix} \begin{pmatrix} 1 & -\sqrt{0.5} \end{pmatrix} + \begin{pmatrix} 0 \\ \sqrt{0.5} \end{pmatrix} \begin{pmatrix} 0 & \sqrt{0.5} \end{pmatrix}.$$

Therefore, the above function is a strongly convex function. In this example we assume 2-blocks approximation is performed, with $n_1 = n_2 = 1$. Notice that the classical Newton's method solves the above QP in one iteration. The coarse correction step, on the other hand, fails to do so; in fact, the diagonal of the Hessian has only 1's, which implies that in this particular example, the coarse correction step is equivalent to gradient descent.

5.7.3 Weakly connected Hessian

We now introduce a specific class of problems in which the coarse correction step could be as good as Newton's method in the limit.

Definition 5.42 *Consider a twice continuously differentiable strongly convex function f_h which satisfies Assumption 5.1. f_h is said to have (δ, q)-weakly connected Hessians if*

$$\nabla^2 f_h(\mathbf{x}_h) = \mathbf{Q}_h(\mathbf{x}_h) + \delta\hat{\mathbf{Q}}_h(\mathbf{x}_h), \tag{5.35}$$

where $\mathbf{Q}_h(\mathbf{x}_h) = diag(\mathbf{Q}_{h,1}(\mathbf{x}_h), \mathbf{Q}_{h,2}(\mathbf{x}_h), \ldots, \mathbf{Q}_{h,q}(\mathbf{x}_h))$ is a block diagonal matrix with q blocks, with $\mathbf{Q}_{h,i}(\mathbf{x}_h) \in \mathbb{R}^{n_i \times n_i}$ and $\sum_{j=1}^{q} n_j = N$ for $i = 1, 2, \ldots, q$. All $\mathbf{Q}_{h,i}(\mathbf{x}_h)$'s are positive definite, and there exists positive constants $\mu_{h,q}, \mu_{h,\hat{q}}, L_{h,q}, L_{h,\hat{q}}$ such that

$$\begin{aligned} \mu_{h,q}\mathbf{I} &\preccurlyeq \mathbf{Q}_h(\mathbf{x}_h) \preccurlyeq L_{h,q}\mathbf{I} \\ \mu_{h,\hat{q}}\mathbf{I} &\preccurlyeq \hat{\mathbf{Q}}_h(\mathbf{x}_h) \preccurlyeq L_{h,\hat{q}}\mathbf{I} \end{aligned}$$

Definition 5.42 defines the specific structure we consider in this section. The defined (δ, q)-weakly connected Hessian provides a connection between the block diagonal matrix and general positive definite matrix. Suppose when $\delta = 0$, then the (δ, q)-weakly connected Hessian is exactly a block diagonal matrix. Similarly, when $\delta = \mathcal{O}(1)$, then the (δ, q)-weakly connected Hessian is a general positive definite matrix.

Notice that when $\delta = 0$, the coarse correction step (5.34) is exactly same as Newton's method. In what follows, we will consider f_h which has (δ, q)-weakly connected Hessians and show how the performance of coarse correction step (5.34) converges to quadratic convergence when $\delta \to 0$.

5.7.4 Analysis

In order to analyze the convergence of the coarse correction step (5.34), we relate it to the classical Newton's step and derive the difference using the following propositions.

Proposition 5.43 ([TS86]) *For matrices* $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$*, suppose* $\mathbf{A}$*,* $\mathbf{C}$*, and* $\mathbf{A}+\mathbf{BCD}$ *are non-singular, then*

$$(\mathbf{A}+\mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1}+\mathbf{D}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{D}\mathbf{A}^{-1}$$

Proof See [TS86]. ∎

Proposition 5.44 *Suppose the Hessian* $\nabla^2 f_{h,k}$ *is* (δ, q)*-weakly connected as defined in Definition 5.42, then*

$$\left(\frac{1}{L_{h,\hat{q}}} + \frac{\delta}{L_{h,q}}\right)\mathbf{I} \preccurlyeq \hat{\mathbf{Q}}^{-1} + \delta\mathbf{Q}^{-1} \preccurlyeq \left(\frac{1}{\mu_{h,\hat{q}}} + \frac{\delta}{\mu_{h,q}}\right)\mathbf{I},$$

and so

$$\left(\frac{1}{\mu_{h,\hat{q}}} + \frac{\delta}{\mu_{h,q}}\right)^{-1}\mathbf{I} \preccurlyeq (\hat{\mathbf{Q}}^{-1} + \delta\mathbf{Q}^{-1})^{-1} \preccurlyeq \left(\frac{1}{L_{h,\hat{q}}} + \frac{\delta}{L_{h,q}}\right)^{-1}\mathbf{I},$$

where

$$\left(\frac{1}{L_{h,\hat{q}}} + \frac{\delta}{L_{h,q}}\right)^{-1} = \frac{L_{h,\hat{q}}L_{h,q}}{L_{h,q}+\delta L_{h,\hat{q}}} \quad \textit{and} \quad \left(\frac{1}{\mu_{h,\hat{q}}} + \frac{\delta}{\mu_{h,q}}\right)^{-1} = \frac{\mu_{h,\hat{q}}\mu_{h,q}}{\mu_{h,q}+\delta\mu_{h,\hat{q}}}.$$

The constants $\mu_{h,q}, \mu_{h,\hat{q}}, L_{h,q}, L_{h,\hat{q}}$ *are defined in Definition 5.42.*

Proof This can be obtained via direct computation. ∎

Proposition 5.45 *Suppose the Hessian* $\nabla^2 f_{h,k}$ *is* (δ, q)*-weakly connected as defined in Definition 5.42, then*

$$\hat{\mathbf{Q}}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1} = \mathbf{I} - \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1},$$

and for any $\mathbf{r}_h \in \mathbb{R}^N$

$$\frac{1}{L_{h,\hat{q}}}\frac{\mu_{h,\hat{q}}\mu_{h,q}}{\mu_{h,q}+\delta\mu_{h,\hat{q}}}\|\mathbf{r}_h\| \leq \|\hat{\mathbf{Q}}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{r}_h\|.$$

Proof

$$\begin{aligned}\mathbf{I} &= (\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}, \\ &= \hat{\mathbf{Q}}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1} + \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1},\end{aligned}$$

and thus,

$$\mathbf{I} - \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1} = \hat{\mathbf{Q}}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}.$$

For the second part,

$$\begin{aligned}
\|\hat{\mathbf{Q}}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{r}_h\|^2 &= \mathbf{r}_h^T(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\hat{\mathbf{Q}}_{h,k}^{-1}\hat{\mathbf{Q}}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{r}_h, \\
&\geq \frac{1}{L_{h,\hat{q}}^2}\mathbf{r}_h^T(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{r}_h, \\
&\geq \frac{1}{L_{h,\hat{q}}^2}\left(\frac{\mu_{h,\hat{q}}\mu_{h,q}}{\mu_{h,q} + \delta\mu_{h,\hat{q}}}\right)^2 \|\mathbf{r}_h\|^2.
\end{aligned}$$

So we obtain the desired result. ∎

We derive the difference between the classical Newton's step and the coarse correction step in the following lemma.

Lemma 5.46 *Suppose the function $f_h(\mathbf{x}_h)$ has (δ, q)-weakly connected Hessians as defined in Definition 5.42. Let*

$$\begin{aligned}
\mathbf{d}_{h,k}^N &= -[\nabla^2 f_{h,k}]^{-1}\nabla f_{h,k}, \\
\mathbf{d}_{h,k}^B &= -[\mathbf{Q}_{h,k}]^{-1}\nabla f_{h,k}.
\end{aligned} \tag{5.36}$$

Then

$$\mathbf{d}_{h,k}^N = \mathbf{d}_{h,k}^B - \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{d}_{h,k}^B.$$

Proof The Newton's step is

$$\mathbf{d}_{h,k}^{\mathrm{N}} = -[\nabla^2 f_{h,k}]^{-1}\nabla f_{h,k} = -[\mathbf{Q}_{h,k} + \delta\hat{\mathbf{Q}}_{h,k}]^{-1}\nabla f_{h,k}.$$

Using Proposition 5.43, we have

$$\begin{aligned}
[\mathbf{Q}_{h,k} + \delta\hat{\mathbf{Q}}_{h,k}]^{-1} &= [\mathbf{Q}_{h,k} + \mathbf{I}(\delta\hat{\mathbf{Q}}_{h,k})\mathbf{I}]^{-1}, \\
&= \mathbf{Q}_{h,k}^{-1} - \mathbf{Q}_{h,k}^{-1}(\delta^{-1}\hat{\mathbf{Q}}_{h,k}^{-1} + \mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{Q}_{h,k}^{-1}, \\
&= \mathbf{Q}_{h,k}^{-1} - \mathbf{Q}_{h,k}^{-1}(\delta\mathbf{I})(\delta\mathbf{I})^{-1}(\delta^{-1}\hat{\mathbf{Q}}_{h,k}^{-1} + \mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{Q}_{h,k}^{-1}, \\
&= \mathbf{Q}_{h,k}^{-1} - \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{Q}_{h,k}^{-1}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\mathbf{d}_{h,k}^{\mathrm{N}} &= -\left(\mathbf{Q}_{h,k}^{-1} - \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{Q}_{h,k}^{-1}\right)\nabla f_{h,k}, \\
&= \mathbf{d}_{h,k}^{\mathrm{B}} - \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{d}_{h,k}^{\mathrm{B}},
\end{aligned}$$

as required. ∎

5.7.5 Convergence

Using Proposition 5.45 and Lemma 5.46, we derive the composite convergence rate.

Theorem 5.47 *Suppose the function $f_h(\mathbf{x}_h)$ has (δ, q)-weakly connected Hessians defined in Definition 5.42. Suppose $\mathbf{d}_{h,k}^{B}$ in (5.36) is taken and $\alpha_{h,k} = 1$, then*

$$\begin{aligned}
\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star}\| \leq\ & \delta\frac{\mu_{h,q} + \delta\mu_{h,\hat{q}}}{\mu_{h,\hat{q}}\mu_{h,q}^2}\frac{L_{h,\hat{q}}^2 L_{h,q}}{L_{h,q} + \delta L_{h,\hat{q}}}\|\mathbf{x}_{h,k+1} - \mathbf{x}_{h,k}\| \\
&+ L_{h,\hat{q}}\frac{\mu_{h,q} + \delta\mu_{h,\hat{q}}}{\mu_{h,\hat{q}}\mu_{h,q}}\frac{M_h}{2\mu_h}\|\mathbf{x}_{h,k} - \mathbf{x}_{h,\star}\|^2.
\end{aligned}$$

Proof Using Lemma 5.46, we obtain

$$\begin{aligned}
\mathbf{x}_{h,k+1} - \mathbf{x}_{h,\star} &= \mathbf{x}_{h,k} - \mathbf{x}_{h,\star} + \mathbf{d}_{h,k}^{\mathrm{B}}, \\
&= \mathbf{x}_{h,k} - \mathbf{x}_{h,\star} + \mathbf{d}_{h,k}^{\mathrm{B}} + \mathbf{d}_{h,k}^{\mathrm{N}} - \mathbf{d}_{h,k}^{\mathrm{N}}, \\
&= \left(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star} + \mathbf{d}_{h,k}^{\mathrm{N}}\right) + \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}\mathbf{d}_{h,k}^{\mathrm{B}}, \\
&= \left(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star} + \mathbf{d}_{h,k}^{\mathrm{N}}\right) + \delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1} + \delta\mathbf{Q}_{h,k}^{-1})^{-1}(\mathbf{x}_{h,k+1} - \mathbf{x}_{h,k}).
\end{aligned}$$

Using the fact that

$$\begin{aligned}&\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1}+\delta\mathbf{Q}_{h,k}^{-1})^{-1}(\mathbf{x}_{h,k+1}-\mathbf{x}_{h,k})\\&\qquad=\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1}+\delta\mathbf{Q}_{h,k}^{-1})^{-1}(\mathbf{x}_{h,k+1}-\mathbf{x}_{h,\star})-\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1}+\delta\mathbf{Q}_{h,k}^{-1})^{-1}(\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}),\end{aligned}$$

we have

$$\begin{aligned}&(\mathbf{I}-\delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1}+\delta\mathbf{Q}_{h,k}^{-1})^{-1})(\mathbf{x}_{h,k+1}-\mathbf{x}_{h,\star})\\&\qquad=\left(\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}+\mathbf{d}_{h,k}^{\mathrm{N}}\right)-\delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1}+\delta\mathbf{Q}_{h,k}^{-1})^{-1}(\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}).\quad(5.37)\end{aligned}$$

Using Proposition 5.45, we have

$$\begin{aligned}\frac{1}{L_{h,\hat{q}}}\frac{\mu_{h,\hat{q}}\mu_{h,q}}{\mu_{h,q}+\delta\mu_{h,\hat{q}}}\|\mathbf{x}_{h,k+1}-\mathbf{x}_{h,\star}\| &\leq \|(\mathbf{I}-\delta\mathbf{Q}_{h,k}^{-1}(\hat{\mathbf{Q}}_{h,k}^{-1}+\delta\mathbf{Q}_{h,k}^{-1})^{-1})(\mathbf{x}_{h,k+1}-\mathbf{x}_{h,\star})\|,\\&\leq \|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}+\mathbf{d}_{h,k}^{\mathrm{N}}\|\\&\quad+\delta\|\mathbf{Q}_{h,k}^{-1}\|\|(\hat{\mathbf{Q}}_{h,k}^{-1}+\delta\mathbf{Q}_{h,k}^{-1})^{-1}\|\|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|,\\&\leq \frac{M_h}{2\mu_h}\|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|^2\\&\quad+\frac{\delta}{\mu_{h,q}}\frac{L_{h,\hat{q}}L_{h,q}}{L_{h,q}+\delta L_{h,\hat{q}}}\|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|.\end{aligned}$$

Therefore,

$$\begin{aligned}\|\mathbf{x}_{h,k+1}-\mathbf{x}_{h,\star}\| &\leq L_{h,\hat{q}}\frac{\mu_{h,q}+\delta\mu_{h,\hat{q}}}{\mu_{h,\hat{q}}\mu_{h,q}}\frac{M_h}{2\mu_h}\|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|^2\\&\quad+L_{h,\hat{q}}\frac{\mu_{h,q}+\delta\mu_{h,\hat{q}}}{\mu_{h,\hat{q}}\mu_{h,q}}\frac{\delta}{\mu_{h,q}}\frac{L_{h,\hat{q}}L_{h,q}}{L_{h,q}+\delta L_{h,\hat{q}}}\|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|,\end{aligned}$$

as required. ∎

Theorem 5.47 shows that the coefficient of $\|\mathbf{x}_{h,k}-\mathbf{x}_{h,\star}\|$ is in $\mathcal{O}(\delta)$. As expected, as $\delta\to 0$, the composite rate in Theorem 5.47 will recover the quadratic convergence, and the linear component of composite rate decays at least linearly with δ.

5.8 Numerical Experiments

In this section, we will first verify our convergence results with three numerical examples. Each example will correspond to each of the settings in Section 5.5-5.7. The first example corresponds to Section 5.5, and it is an one-dimensional Poisson's equation, which is a standard example in numerical analysis and multigrid algorithms. In the second example, considered in Section 5.6, we use regularized logistic problem to be the illustrative example. In the third example, we consider a synthetic example to study the case in Section 5.7. We investigate the convergence by varying the parameter δ.

In the second part of this section, we will compare GAMA with other algorithms. We emphasize that the goal of this chapter is to gain understanding in the Galerkin-based multilevel algorithm, which apparently is closely related to many existing algorithms: ranging from conventional multigrid algorithms to machine learning-driven algorithms. The use of this section is to show the potential of Galerkin model, and we are not trying to claim that GAMA outperforms the state-of-the-art algorithms, including variants of GAMA.

5.8.1 Poisson's Equation

We consider an one-dimensional Poisson's equation

$$-\frac{\mathrm{d}^2}{\mathrm{d}q^2}u = w(q) \quad \text{in} \quad [0,1], \quad u(0) = u(1) = 0,$$

where $w(q)$ is chosen as

$$w(q) = \sin(4\pi q) + 8\sin(32\pi q) + 16\sin(64\pi q).$$

We discretize the above problem and denote $\mathbf{x}, \mathbf{b} \in \mathbb{R}^{N-1}$, where $(\mathbf{x})_i = u(i/N)$ and $(\mathbf{b})_i = w(i/N)$, for $i = 1, 2, \ldots, N-1$. By using finite differences, we approximate the above equation with

$$\min_{\mathbf{x}\in\mathbb{R}^{N-1}} \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}, \tag{5.38}$$

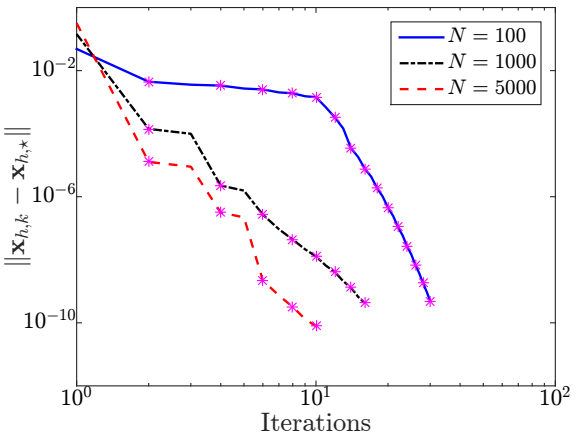


Figure 5.4: Convergence of solving Poisson's equation with different N's

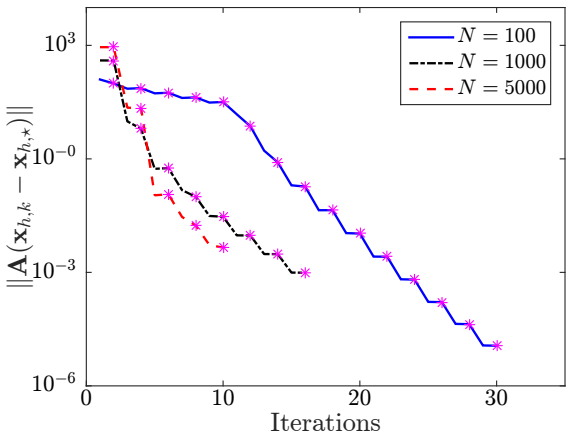


Figure 5.5: The smoothing effect with different N's

where $\mathbf{A}$ is defined as in Lemma 5.32, which is a discretized Laplacian operator.

Figure 5.4 shows the convergence results of solving (5.38) with different N's. In this example we use the prolongation and restriction operators that are defined in (5.25) and (5.26). Since there is only one pair of $\mathbf{P}$ and $\mathbf{R}$, we follow the traditional multigrid approach in which we combine the coarse correction step with the fine correction step. Gradient descent is used to compute the fine correction step. The pink stars in Figure 5.4 and Figure 5.5 indicate where coarse correction steps were used.

As expected from Corollary 5.33, the performance of convergence is inversely proportional to the discretization level N. More interestingly, one can see the complementary of fine correction step and coarse correction step. From Figure 5.4, fine correction steps are often deployed after coarse correction steps. Each pair of fine and coarse correction steps provides significant improvement in convergence. Figure 5.5 shows the smoothing effect of the fine correction step by looking at the quantity $\|\mathbf{A}(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$, where $\mathbf{A}$ is the discretized Laplacian operator, as defined in Lemma 5.32. As opposed to coarse correction steps, fine correction steps are effective in reducing $\|\mathbf{A}(\mathbf{x}_{h,k} - \mathbf{x}_{h,\star})\|$. Once the error is smoothed, coarse correction steps provide large reduction in error, as shown in Figure 5.4.

5.8.2 Regularized Logistic Regression

We study the Galerkin model that is generated via naïve Nyström method and consider an example in ℓ_1 regularized logistic regression,

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{m}\sum_{i=1}^{m} \log(1+\exp(-b_i\mathbf{x}^T\mathbf{a}_i)) + \omega_1\|\mathbf{x}\|_1,$$

where $\omega_1 \in \mathbb{R}^+$, and $\{(\mathbf{a}_i, b_i)\}_{i=1}^m$ is a training set with m instances. For $i = 1, 2, \ldots, m$, $\mathbf{a}_i \in \mathbb{R}^N$ is an input and $b_i \in \mathbb{R}$ is the corresponding output.

Notice that the above formulation involves the non-differentiable function $\|\mathbf{x}\|_1$, and so the above problem is beyond the scope of the setting in this chapter. To overcome this issue, we replace the $\|\mathbf{x}\|_1$ with its approximation, the pseudo-Huber function [FG16], and solve the following formulation.

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{m}\sum_{i=1}^{m} \log(1+\exp(-b_i\mathbf{x}^T\mathbf{a}_i)) + \omega_1\sum_{i=1}^{N}\left((\mu_r^2+\mathbf{x}_i^2)^{1/2}-\mu_r\right), \tag{5.39}$$

where $\mu_r \in \mathbb{R}^+$ is a parameter, and it provides good approximation of the ℓ_1 norm when μ_r is small.

The dataset *gisette* is used for $\{(\mathbf{a}_i, b_i)\}_{i=1}^m$. *Gisette* is a handwritten digits dataset from the NIPS 2003 feature selection challenge. In this example $N = 5000$, $m = 6000$, and we choose parameter ω_1 from [LSS14, YHL12] and $\mu_r = 0.001$. One can find and download *gisette* at *https://www.csie.ntu.edu.tw/˜cjlin/libsvmtools/datasets.*

Notice that when $\mathbf{P} \in \mathbb{R}^{N\times n}$ and $\mathbf{R} \in \mathbb{R}^{n\times N}$ are generated using naïve Nyström method according to Definition 5.34, n is a user-defined parameter, and the probabilistic approach mentioned in Section 5.2 is used to generate multiple $\mathbf{P}$'s and $\mathbf{R}$'s. That is, a pair of $\mathbf{P}$ and $\mathbf{R}$ is sampled uniformly over $\binom{N}{n}$ possible coarse models. This setting satisfies the condition stated in Proposition 5.7, and so no fine correction step is needed.

Figure 5.6 shows the convergence results. As expected from Corollary 5.38 and 5.39, the performance of convergence is proportional to n.

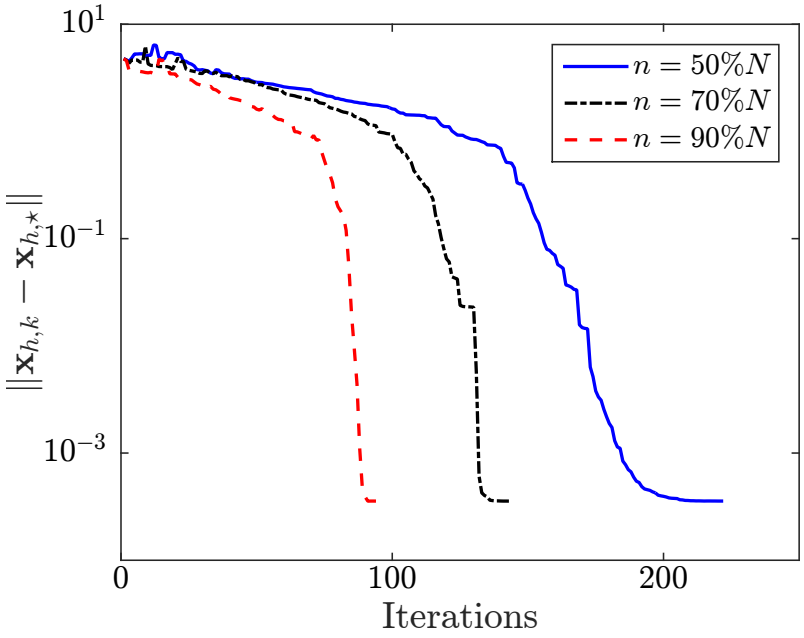


Figure 5.6: The ℓ_1 regularized logistic regression example.

5.8.3 A Synthetic Example for Block Diagonal Approximation

To study the case of block diagonal approximation in Section 5.7, we construct an artificial example with weakly connected Hessian. In particular, we solve

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\mathbf{x}^T \left(\mathbf{Q}_h + \delta\hat{\mathbf{Q}}_h\right)\mathbf{x} + \mathbf{b}^T\mathbf{x}, \tag{5.40}$$

where $\delta \in \mathbb{R}^+$, $\mathbf{Q}_h = \text{diag}(\mathbf{Q}_{h,1}, \mathbf{Q}_{h,2}, \ldots, \mathbf{Q}_{h,p})$ is a block diagonal matrix with p blocks, with $\mathbf{Q}_{h,i}(\mathbf{x}_h) \in \mathbb{R}^{n_i\times n_i}$ and $\sum_{i=1}^p n_i = n$ for $i = 1, 2, \ldots, p$. In this example, we have $N = 1000$, $p = 10$, $n_1 = n_2 = \cdots = n_{10} = 100$. We construct $\hat{\mathbf{Q}}_h$ via

$$\hat{\mathbf{Q}}_h = \sum_{j=1}^{N} v_j \mathbf{u}_j \mathbf{u}_j^T,$$

where $v_j \in \mathbb{R}^+$ is sampled uniformly from $[v_\delta, 1 + v_\delta]$, and $\mathbf{u}_j \in \mathbb{R}^N$ is a random orthonormal vectors, for $j = 1, 2, \ldots, N$. Each $\mathbf{Q}_{h,i}$ is also constructed similar to $\hat{\mathbf{Q}}_h$ but in the smaller dimension $\mathbb{R}^{n_i\times n_i}$, for $i = 1, 2, \ldots, p$. $v_\delta = 0.0001$ in this example.

We consider the optimization problem in (5.40) with different δ's. As expected from Theo-

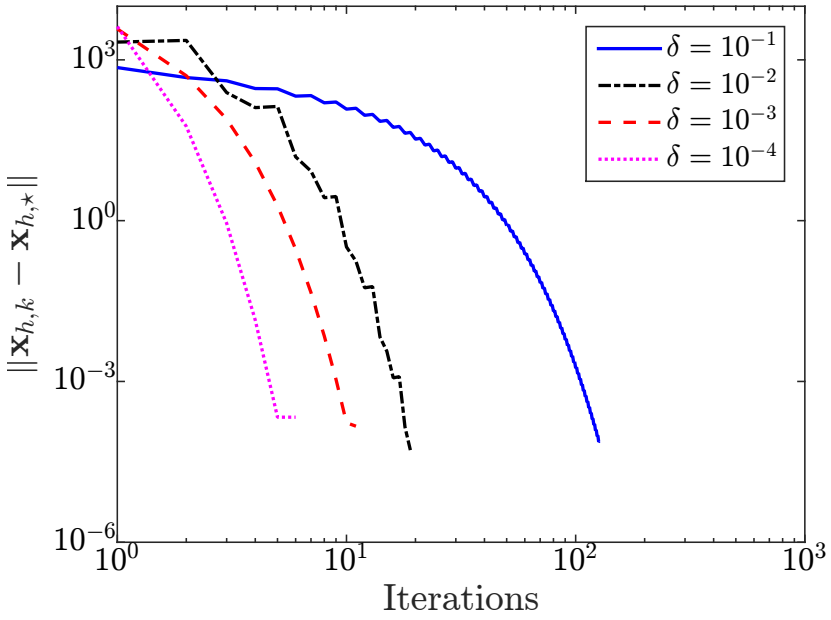


Figure 5.7: Block diagonal approximation.

rem 5.47, Figure 5.7 shows that the performance of convergence is inversely proportional to δ.

5.8.4 Numerical Performance: PDE Test Cases

We now compare the numerical performance of GAMA with the conventional unconstrained optimization algorithms as well as conventional line search multilevel/multigrid algorithm in [WG09]. We focus on PDE-based optimization problems in this section.

We test algorithms on five examples from [WG09, GMS$^+$10], and all of them are discretized 2-dimensional variational problems over the unit square $\mathcal{S}_2 \triangleq [0,1] \times [0,1]$. The decision variable, $u(x,y)$, obeys the boundary condition, $u = 0$ on $\partial\mathcal{S}_2$, for all problems. The five problems are listed in the following.

1. Problem DSSC:

$$\min_{u \in \mathcal{S}_2} \int_{\mathcal{S}_2} \frac{1}{2} \|\nabla u(x,y)\|^2 - \lambda \exp(u(x,y)), \quad \text{where} \quad \lambda = 6.$$

2. Problem WEN:

$$\min_{u\in\mathcal{S}_2}\int_{\mathcal{S}_2}\frac{1}{2}\|\nabla u(x,y)\|^2+\lambda\exp\left[u(x,y)\right](u(x,y)-1)-\gamma(x,y)u(x,y),$$

where $\lambda=6$ and

$$\gamma(x)=\left[\left(9\pi^2+\lambda\exp\left[\left(x^2-x^3\right)\sin(3\pi y)\right]\right)\left(x^2-x^3\right)+6x-2\right]\sin\left(3\pi y\right).$$

3. Problem BRATU:

$$\min_{u\in\mathcal{S}_2}\int_{\mathcal{S}_2}\|\Delta u(x,y)-\lambda\exp(u(x,y))\|^2,\quad\text{where}\quad\lambda=6.8.$$

4. Problem POSSION2D:

$$\min_{\mathbf{x}}\frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x}-\mathbf{b}^T\mathbf{x},$$

where $\mathbf{A}$ and $\mathbf{b}$ are the discretizations of the Laplacian and the function $\gamma(x,y)=2(y(1-y)+x(1-x))$, respectively.

5. Problem IGNISC:

$$\min_{u\in\mathcal{S}_2}\int_{\mathcal{S}_2}(u(x,y)-z)^2+\frac{\beta}{2}\int_{\mathcal{S}_2}(\exp[u(x,y)]-\exp[z])^2 + \frac{\zeta}{2}\int_{\mathcal{S}_2}\|\Delta u(x,y)-\delta\exp(u(x,y))\|^2,$$

where $\delta=\beta=6.8$, $\zeta=10^{-5}$, and $z=1/\pi^2$.

Table 5.1 shows the numerical performance of different algorithms, i.e., the CPU time (Time) needed to achieve a small $\|\nabla f_{h,k}\|$ (Accuracy). We denote the COnventional Multilevel Algorithm as *COMA*. For both GAMA and COMA, we denote "-NT" and "-qNT" when Newton's method and L-BFGS are used for the fine correction steps, respectively. For all five examples, we choose the fine models to be the discretization with mesh size $\Delta x\times\Delta y$, where $\Delta x=\Delta x=1/2^{10}$, and the standard five-point finite differences are used. We point out that all the algorithmic

	DSSC		WEN		BRATU	
	Time	Accuracy	Time	Accuracy	Time	Accuracy
L-BFGS	5524.9	9.5164e-06	1048.7	9.9788e-06	44355	12.449
Newton	59.01	1.3351e-07	47.6	4.3493e-08	565.79	2.0853e-06
GAMA-NT	21.8	1.153e-13	21.26	4.6412e-12	180.19	1.9028e-06
COMA-NT	20.13	1.1531e-13	20.19	4.6412e-12	161.52	1.9027e-06
GAMA-qNT	13	7.2882e-06	5.1	7.9565e-06	840.47	0.0021644
COMA-qNT	12.52	9.4619e-06	6.43	7.3332e-06	860.4	37.708

	POSSION2D		IGNISC	
	Time	Accuracy	Time	Accuracy
L-BFGS	1105.7	7.815e-06	50274	0.00039108
Newton	15.99	7.2561e-15	124.93	2.1409e-06
GAMA-NT	20.93	0	77.58	2.0008e-11
COMA-NT	20.92	0	77.69	2.0008e-11
GAMA-qNT	1.28	8.1249e-06	62.81	9.0643e-06
COMA-qNT	1.46	8.1304e-06	43.13	9.1113e-06

Table 5.1: PDE-based text examples

settings are the same as in [WG09], including line search strategy, stopping criteria, and choice of parameters.

For both GAMA and COMA, we follow the same strategy as in [WG09], and the standard full multilevel scheme is deployed. Suppose level j is denoted as the discretization with mesh size $\Delta x \times \Delta y$, where $\Delta x = \Delta y = 1/2^j$. For $j = 3, 4, \ldots, 9$, we compute the solution $\mathbf{x}_{j,\star}$ in level j, and use $\mathbf{P}_j^{j+1}\mathbf{x}_{j,\star}$ as the initial guess for level $j+1$. $\mathbf{P}_j^{j+1}$ is denoted as the prolongation operator from level j to level $j+1$.

From Table 5.1, we can see that the multilevel algorithms clearly outperform the conventional algorithms. The performance of GAMA is comparable with COMA and is more robust due to the use of second order information. In the problem BRATU, first order algorithms (i.e. L-BFGS, GAMA-qNT, and COMA-qNT) are not efficient, but GAMA-qNT is able to achieve much better accuracy. Therefore, GAMA is empirically competitive against the conventional multilevel algorithm, and yet more robust with a more understandable rate of convergence.

	f_i's	N	m	ω_1	ω_2
YearPredictionMSDt	Quadratic	90	51630	10^{-6}	10^{-6}
log1pE2006test	Quadratic	4272226	3308	10^{-6}	10^{-6}
w8at	Logistic	300	14951	0	$1/m$
Gisette	Logistic	5000	6000	$1/(0.25m)$	0
epsilon_normalizedt	Logistic	2000	100000	0	$1/m$

Table 5.2: Details of ERM Test Examples

5.8.5 Numerical Performance: Machine Learning Test Cases

We now study the performance of GAMA that is generated by Nyström method. Suppose we have the training set $\{(\mathbf{a}_i, b_i)\}_{i=1}^m$, we use GAMA to solve the empirical risk minimization (ERM) problem

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{a}_i^T\mathbf{x}) + \omega_1\|\mathbf{x}\|_1 + \frac{\omega_2}{2}\|\mathbf{x}\|_2^2,$$

where $\omega_1, \omega_2 \in \mathbb{R}$ and $\mathbf{a}_i \in \mathbb{R}^N$, for $i = 1, 2, \ldots, m$. Special cases of f_i include

1. Quadratic loss function: $f_i(x) = \frac{1}{2}(x - b_i)^2$.
2. Logistic loss function: $f_i(x) = \log(1 + \exp(-xb_i))$.

In the case where $w_2 = 0$ and f_i's are logistic loss functions, we yield the ℓ_1 regularized logistic regression considered in Section 5.8.2. Similar to Section 5.8.2, we replace the $\|\mathbf{x}\|_1$ with the pseudo-Huber function.

The numerical test is conducted over five examples. All dataset/training set can be download at *https://www.csie.ntu.edu.tw/˜cjlin/libsvmtools/datasets/*. Table 5.2 provides details of the test examples. We point out that for logistics regression, we select the choice of ω_1 and ω_2 based on [LSS14, GGR16]. For linear regression, we simply select $\omega_1 = \omega_2 = 10^{-6}$, which is a commonly used value.

Figure 5.8-5.12 show the numerical performance of GAMA, compared to Newton's method and L-BFGS. Over these five examples, GAMA only performs coarse correction steps, and n is chosen to be $10\%N$, $20\%N$, and $30\%N$. An exception can be found in log1pE2006test because these choices of n's are too large to be traceable. The performance of Newton's method is

also missing for log1pE2006test because computing its search direction is intractable due to the size of N. From Figure 5.8 and 5.10, we can see that when N is small, Newton's method outperforms the other methods. This is not surprising since the per-iteration cost is cheap for small N and yet Newton's method enjoys the quadratic convergence. When N is sufficiently large, as showed in Figure 5.9 and 5.11, GAMA is competitive compared to both Newton's method and L-BFGS.

In Figure 5.12, we can see that the performance of GAMA is better than Newton's method and similar to L-BFGS. From Table 5.2, $N = 2000$ and it is a reasonably good size for Newton's method. The poor performance of Newton's method is due to the large m, which is 100000. For large m, the evaluation of Hessians becomes the computational bottleneck. To further illustrate this, in Figure 5.13, we perform Newton's method and GAMA with sub-sampling. For subsample Newton's method, at each iteration, we evaluate the Hessian based on $\sqrt{m}$ data points in the training set. Data points are sampled uniformly without replacement. For GAMA, we deploy the idea of SVRG, sample $\sqrt{m}$ data points at each coarse correction step, and create a coarse model with

$$f_H(\mathbf{x}) = \frac{1}{\sqrt{m}} \sum_{i \in \mathcal{S}_H} f_i(\mathbf{a}_i^T \mathbf{x}) + \omega_1 \sum_{i=1}^{N} \left((\mu_r^2 + \mathbf{x}_i^2)^{1/2} - \mu_r \right) + \frac{\omega_2}{2} \|\mathbf{x}\|_2^2,$$

where $\mu_r = 0.001$ and $\mathcal{S}_H$ is the set of the samples. We call the coarse model with above f_H the intermediate coarse model. When solving the intermediate coarse model, we apply the Galerkin-model that is generated by Nyström method, and apply five coarse correction steps. The incumbent solution of the intermediate coarse model is then prolonged to the fine model and results in a coarse correction step on the fine model. We clarify that this algorithmic procedure follows the idea of SVRG, as introduced in Section 5.2.5. As shown in Figure 5.13, great improvements are achieved for both (subsample) Newton's method and GAMA. The computational bottleneck of evaluating Hessians is removed by subsampling data points. Since solving a system of 2000 linear equations can be managed easily, Newton's method outperforms all the other method in this case. Notice that since the Hessians are not evaluated exactly in this case, Newton's method and GAMA no longer enjoy quadratic rate and composite rate,

respectively. The theoretical performance of these methods are beyond the scope of this chapter.

5.9 Comments and Perspectives

We showed the connections between the general multilevel framework and the conventional optimization methods. The case of using Galerkin model (GAMA) is further studied, and the local composite rate of convergence is derived. When the coefficient of the linear component in composite rate is sufficiently small, then GAMA is superior to Newton's method in complexity. This linear component is then studied in three different cases, and we showed how the structure in each case would improve the rate of convergence.

This work advances research in multilevel optimization algorithms in several non-exploited directions. Firstly, the connections between multilevel framework and standard optimization methods would motivate systematic designs in optimization algorithms, and the multilevel framework could be used beyond the traditional linesearch multilevel method in [WG09].

Secondly, we take the first step towards showing how the structure of problems could improve the convergence. We expect that similar manner of thinking could be applied beyond GAMA, and we believe this line of research could motivate more developments in multilevel algorithms when one tries to tackle problems with specific structure.

We believe the results presented in this chapter can be generalized and refined. For example, the local composite rate of convergence when solving PDE-based optimization can be extended to cases beyond one-dimensional problems or uniform discretization. These extensions would require more careful and tedious algebra, but the general approach presented in Section 5.5 can be applied. On the other hand, one can extend results in Section 5.6 by considering different versions of Nyström method, or even different methods in low rank approximation in general. These generalizations could be done under the general approach of this chapter.

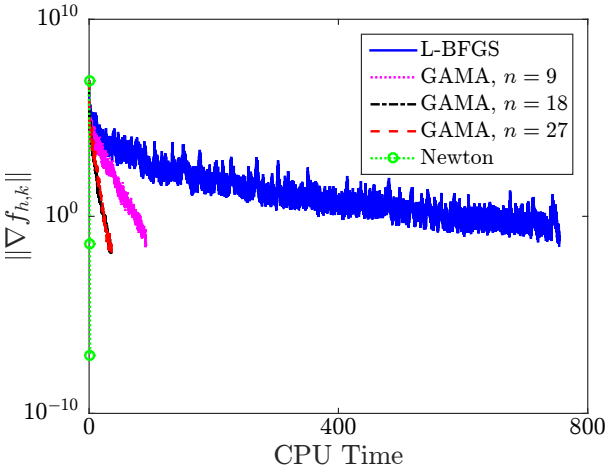


Figure 5.8: YearPredictionMSDt

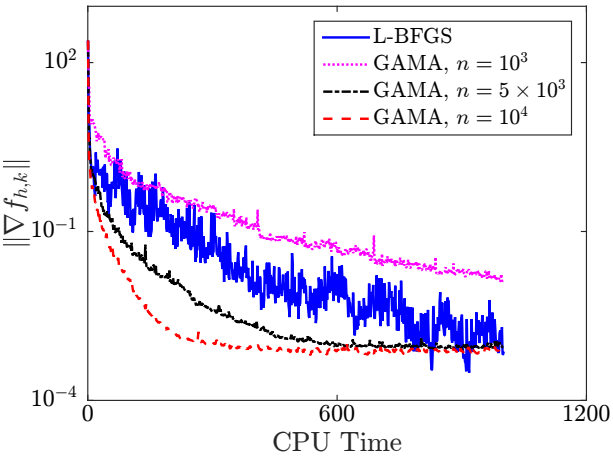


Figure 5.9: log1pE2006test

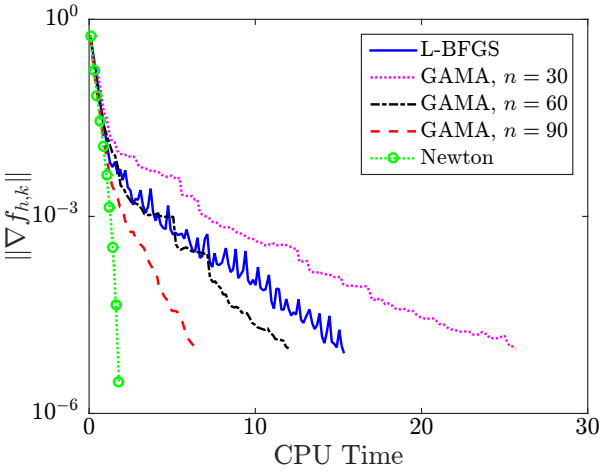


Figure 5.10: w8at

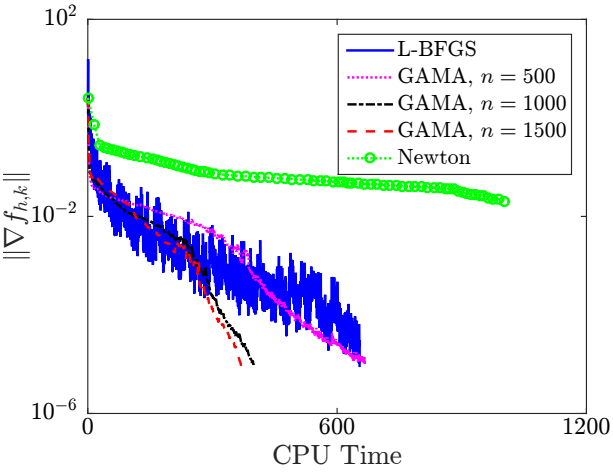


Figure 5.11: Gisette

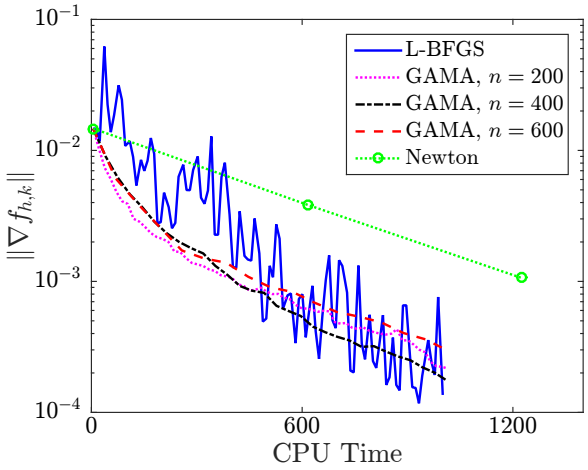


Figure 5.12: epsilon_normalizedt

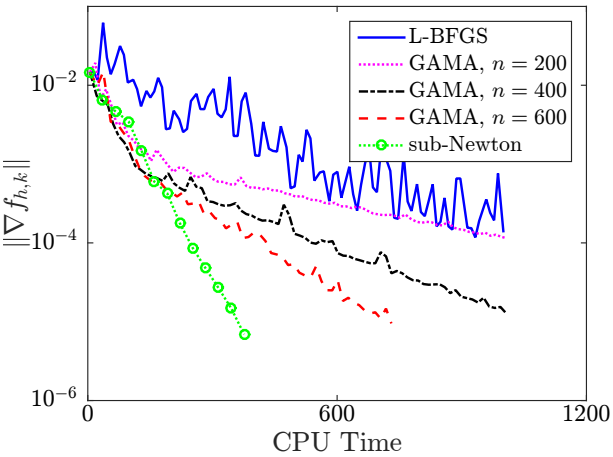


Figure 5.13: epsilon_normalizedt (subsample)

Chapter 6

Discussion

We can only see a short distance ahead, but we can see plenty there that needs to be done.

Alan Turing

Solving optimization problems efficiently could be a difficult or maybe even an impossible task. One obvious bottleneck could be caused by the size of the problem, and many different approaches have been proposed to improve the computational performance. In this thesis, we concentrate on making use of the inherent structure of problems to improve the computation performance. By taking a close look at the structure presented, we are able to design algorithms to tackle problems with special structures.

6.1 Summary

In Chapter 3, we considered MDPs with multiscale structure, which we called MMDPs. We showed that the multiscale stucture causes the ill-conditioning when solving MMDPs. Using the existing aggregation techniques, we designed a multilevel algorithm in which part of the computations is replaced by using approximations of the exact MMDPs. We show that the

proposed algorithm is able to circumvent the ill-conditioning of this class of problems, and thus has lower complexity.

In Chapter 4, the class of ERM problems was considered. First order methods are considered to be the standard approach when solving ERMs. Although the existing literature considers the complexity of first order method as "dimension-free", we showed that the complexity of solving ERMs in fact depends on the statistics of the training data. This was achieved by applying random matrix theory to derive bounds for Lipschitz constant of the empirical risk. The results were also used to design a new stepsize strategy for first order algorithms.

Chapter 5 concerns the unconstrained twice continuously differentiable convex programs. The connections between standard optimization methods and the general multilevel framework were discussed. By considering the Galerkin model, we conducted case studies on infinite-dimensional optimization, low rank models, and models with weakly connected Hessians, and we showed how the structure of each case could affect the convergence.

6.2 Future Work

There are many exciting unexplored research directions related to this thesis. The general multilevel framework, in our opinion, could be applied beyond the traditional setting of geometric structure. In Chapter 5, we showed that the general multilevel framework could be reduced to some state-of-the-art algorithms for machine learning applications, including block-coordinate gradient descent and SVRG. One interesting extension could be applying state-of-the-art sketching methods to approximate the training set with a smaller matrix and create a coarse model with a smaller dataset. A multilevel algorithm is then created using the general framework, and operations such as function evaluations and gradient evaluations would be much cheaper when the coarse model is used.

Similar ideas could be applied to block-coordinate gradient descent methods. The core idea of these methods is to partition the updates of the decision variables in blocks. However, this partitioning process ignores the correlation among variables and thus these algorithms require

a larger number of iterations. Using the same idea of sketching methods, we can "sketch" the variables that are not selected in the block for each update, add them to the updating block, and so the correlations among variables are preserved.

We believe the general multilevel framework can also be used to design distributed and parallel algorithms. Using a similar idea as SVRG, one can subsample the data into many sub-datasets, and each sub-data would correspond to one coarse model. All coarse models could be solved in a distributed or parallel manner. Since all solutions of the coarse models would provide descent directions, the sum of them will guarantee a descent direction. One would expect that the direction generated by this distributed algorithm will be more robust, compared to using only one coarse model.

In conclusion, we believe the general approach taken in this thesis can be used to tackle structured problems in many applications.

Bibliography

[BCW14] A. Belloni, V. Chernozhukov, and L. Wang. Pivotal estimation via square-root Lasso in nonparametric regression. *The Annals of Statistics*, 42(2):757–788, 2014.

[Ber95] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.

[Ber07] D. P. Bertsekas. *Dynamic Programming and Optimal Control: 2*. Athena Scientific, 2007.

[BF95] J. V. Burke and M. C. Ferris. A Gauss-Newton method for convex composite optimization. *Mathematical Programming*, 71(2, Ser. A):179–194, 1995.

[BHM00] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM, 2000.

[Bil12] P. Billingsley. *Probability and measure*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., Hoboken, NJ, 2012. Anniversary edition [of MR1324786], With a foreword by Steve Lalley and a brief biography of Billingsley by Steve Koppes.

[BL11] J. R. Birge and F. Louveaux. *Introduction to stochastic programming*. Springer Series in Operations Research and Financial Engineering. Springer, New York, second edition, 2011.

[BM08] M. Bard and C. March. Multiscale singular perturbations and homogenization of optimal control problems. *Series on Advances in Mathematics for Applied Sciences*, 76:1–27, 2008.

[Bot98] L. Bottou. On-line learning in neural networks. chapter On-line Learning and Stochastic Approximations, pages 9–42. Cambridge University Press, New York, NY, USA, 1998.

[Bot12] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Statistical learning and data science*, Computer Science and Data Analysis Series, pages 17–25. CRC Press, Boca Raton, FL, 2012.

[BT09] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[BT13] A. Beck and L. Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013.

[BTMN01] A. Ben-Tal, T. Margalit, and A. Nemirovski. The ordered subsets mirror descent optimization method with applications to tomography. *SIAM Journal on Optimization*, 12(1):79–108 (electronic), 2001.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[Cal10] G. C. Calafiore. Random convex programs. *SIAM Journal on Optimization*, 20:3427–3464, 2010.

[CC05] G. Calafiore and M. C. Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1, Ser. A):25–46, 2005.

[Chr09] P. D. Christofides. *Control and Optimization of Multiscale Process Systems*. Birkhuser, 2009.

[CL06] F. Chung and L. Lu. *Complex graphs and networks*, volume 107 of *CBMS Regional Conference Series in Mathematics*. Published for the Conference Board of the Mathematical Sciences, Washington, DC; by the American Mathematical Society, Providence, RI, 2006.

[CR09] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

[CR16] D. Csiba and P. Richtárik. Importance sampling for minibatches. *CoRR*, abs/1602.02283, 2016.

[CRT06] E. J. Candès, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

[CS94] T. F. Chan and B. F. Smith. Domain decomposition and multigrid algorithms for elliptic problems on unstructured meshes. In *Domain decomposition methods in scientific and engineering computing (University Park, PA, 1993)*, volume 180 of *Contemporary Mathematics*, pages 175–189. American Mathematical Society, Providence, RI, 1994.

[CT91] C.-S. Chow and J. N. Tsitsiklis. An optimal one-way multigrid algorithm for discrete-time stochastic control. *IEEE Transactions on Automatic Control*, 36(8):897–914, 1991.

[DBLJ14] A. Defazio, F. Bach, and S. Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1646–1654. Curran Associates, Inc., 2014.

[DES82] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact newton methods. *SIAM Journal on Numerical Analysis*, 19:400–408, 1982.

[dFVR04] D. P. de Farias and B. Van Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29:462–478, 2004.

[DM77] J. E. Dennis, Jr. and J. J. Moré. Quasi-Newton methods, motivation and theory. *SIAM Review*, 19(1):46–89, 1977.

[DM05] P. Drineas and M. W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

[Don06] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[EM15] M. A. Erdogdu and A. Montanari. Convergence rates of sub-sampled newton methods. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3052–3060, 2015.

[EY36] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[FG16] K. Fountoulakis and J. Gondzio. A second-order method for strongly convex ℓ_1-regularization problems. *Mathematical Programming*, 156(1-2, Ser. A):189–219, 2016.

[FT15] K. Fountoulakis and R. Tappenden. A Flexible Coordinate Descent Method for Big Data Applications. *ArXiv e-prints*, July 2015.

[GGR16] R. M. Gower, D. Goldfarb, and P. Richtárik. Stochastic Block BFGS: Squeezing More Curvature out of Data. *ArXiv e-prints*, March 2016.

[Git11] A. Gittens. The spectral norm error of the naïve Nystrom extension. *ArXiv e-prints*, October 2011.

[Git13] A. Gittens. *Topics in Randomized Numerical Linear Algebra*. ProQuest LLC, Ann Arbor, MI, 2013. Thesis (Ph.D.)–California Institute of Technology.

[GK13] C. C. Gonzaga and E. W. Karas. Fine tuning Nesterov's steepest descent algorithm for differentiable convex programming. *Mathematical Programming*, 138(1-2, Ser. A):141–166, 2013.

[GMS+10] S. Gratton, M. Mouffe, A. Sartenaer, P. L. Toint, and D. Tomanos. Numerical experience with a recursive trust-region method for multilevel nonlinear bound-constrained optimization. *Optimization Methods and Software*, 25(3):359–386, 2010.

[GS97] K. Golabi and R. Shepard. Pontis: A system for maintenance optimization and improvement of us bridge networks. *Interfaces*, 27(1):71–88, 1997.

[GST08] S. Gratton, A. Sartenaer, and P. L. Tonint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, 19:414–444, 2008.

[Hac03] W. Hackbusch. *Multi-Grid Methods and Applications*. Springer, 2003.

[HL15] Y. Huang and H. Liu. A Barzilai-Borwein type method for minimizing composite functions. *Numerical Algorithms*, 69(4):819–838, 2015.

[HMT11] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

[HP14] C. P. Ho and P. Parpas. Singularly perturbed Markov decision processes: a multiresolution algorithm. *SIAM Journal on Control and Optimization*, 52(6):3854–3886, 2014.

[HPZ15] V. Hovhannisyan, P. Parpas, and S. Zafeiriou. MAGMA: Multi-level accelerated gradient mirror descent algorithm for large-scale convex composite minimization. *ArXiv e-prints*, September 2015.

[HYB17] J. Han, Y. Yang, and H. Bi. A new multigrid finite element method for the transmission eigenvalue problems. *Applied Mathematics and Computation*, 292:96–106, 2017.

[JZ13] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and

K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 315–323. Curran Associates, Inc., 2013.

[Kan52] L. V. Kantorovich. *Functional analysis and applied mathematics.* NBS Rep. 1509. U. S. Department of Commerce, National Bureau of Standards, Los Angeles, Calif., 1952. Translated by C. D. Benster.

[KKO87] P. Kokotovic, H. K. Khali, and J. O'reilly. *Singular perturbation methods in control: analysis and design*, volume 25. SIAM, 1987.

[KM15] V. Koltchinskii and S. Mendelson. Bounding the smallest singular value of a random matrix without concentration. *International Mathematics Research Notices. IMRN*, (23):12991–13008, 2015.

[KM16] M. Kočvara and S. Mohammed. A first-order multigrid method for bound-constrained convex optimization. *Optimization Methods and Software*, 31(3):622–644, 2016.

[KRC+15] K. Karanasos, S. Rao, C. Curino, C. Douglas, K. Chaliparambil, G. M. Fumarola, S. Heddaya, R. Ramakrishnan, and S. Sakalanaga. Mercury: Hybrid centralized and distributed scheduling in large shared clusters. In *Proceedings of the 2015 USENIX Conference on Usenix Annual Technical Conference*, USENIX ATC '15, pages 485–497, Berkeley, CA, USA, 2015. USENIX Association.

[KSHdM02] A. J. Kleywegt, A. Shapiro, and T. Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2001/02.

[LN05] R. M. Lewis and S. G. Nash. Model problems for the multigrid optimization of systems governed by differential equations. *SIAM Journal on Scientific Computing*, 26(6):1811–1837 (electronic), 2005.

[LN13] R. M. Lewis and S. G. Nash. Using inexact gradients in a multilevel optimization algorithm. *Computational Optimization and Applications*, 56(1):39–61, 2013.

[LPRR16] D. V. N. Luong, P. Parpas, D. Rueckert, and B. Rustem. A Weighted Mirror Descent Algorithm for Nonsmooth Convex Optimization Problem. *Journal of Optimization Theory and Applications*, 170(3):900–915, 2016.

[LSS14] J. D. Lee, Y. Sun, and M. A. Saunders. Proximal Newton-type methods for minimizing composite functions. *SIAM Journal on Optimization*, 24(3):1420–1443, 2014.

[Mey08] S. P. Meyn. *Control techniques for complex networks*. Cambridge University Press, 2008.

[Mir60] L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *The Quarterly Journal of Mathematics*, 11:50–59, 1960.

[MKC13] L. Mai, E. Kalyvianaki, and P. Costa. Exploiting time-malleability in cloud-based batch processing systems. In *Workshop on Large-Scale Distributed Systems and Middleware (LADIS'13)*. ACM, November 2013.

[MNJ16] P. Moritz, R. Nishihara, and M. I. Jordan. A linearly-convergent stochastic L-BFGS algorithm. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain, May 9-11, 2016*, pages 249–258, 2016.

[Nas00] S. G. Nash. A multigrid approach to discretized optimization problems. *Optimization Methods and Software*, 14(1-2):99–116, 2000. International Conference on Nonlinear Programming and Variational Inequalities (Hong Kong, 1998).

[Nas14] S. G. Nash. Properties of a class of multilevel optimization algorithms for equality-constrained problems. *Optimization Methods and Software*, 29(1):137–159, 2014.

[Nes04] Y. Nesterov. *Introductory lectures on convex optimization: a basic course*. Kluwer Academic Publishers, Boston, MA, 2004.

[Nes13] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1, Ser. B):125–161, 2013.

[Nes15] Y. Nesterov. Universal gradient methods for convex optimization problems. *Mathematical Programming*, 152(1-2, Ser. A):381–404, 2015.

[NW06] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer Series in Operations Research and Financial Engineering. Springer, New York, second edition, 2006.

[Nys30] E. J. Nyström. Über Die Praktische Auflösung von Integralgleichungen mit Anwendungen auf Randwertaufgaben. *Acta Mathematica*, 54(1):185–204, 1930.

[PGD+15] R. Patel, T. A. Goldstein, E. L. Dyer, A. Mirhoseini, and R. G. Baraniuk. oASIS: Adaptive Column Sampling for Kernel Matrix Approximation. *ArXiv e-prints*, May 2015.

[PJ92] B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

[PLRR] P. Parpas, D. V. N. Luong, D. Rueckert, and B. Rustem. A multilevel proximal algorithm for large scale composite convex optimization, Working paper.

[Pol87] B. T. Polyak. *Introduction to optimization*. Translations Series in Mathematics and Engineering. Optimization Software, Inc., Publications Division, New York, 1987. Translated from the Russian, With a foreword by Dimitri P. Bertsekas.

[Pow11] W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley, 2011.

[Pri07] J.L. Prigent. *Portfolio Optimization and Performance Analysis*. Chapman and Hall/CRC Financial Mathematics Series. CRC Press, 2007.

[PTVF96] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes: the art of scientific computing. Code CD-ROM v 2.06 with Windows, DOS, or Macintosh single-screen license*. Cambridge University Press, Cambridge, 1996.

[PW14] P. Parpas and M. Webster. A stochastic multiscale model for electricity generation capacity expansion. *European Journal of Operational Research*, 232(2):359 – 374, 2014.

[QR14] Z. Qu and P. Richtarik. Coordinate descent with arbitrary sampling ii: expected separable overapproximation. *arXiv:1412.8063*, 2014.

[QRTF15] Z. Qu, P. Richtárik, M. Takác, and O Fercoq. SDNA: Stochastic Dual Newton Ascent for Empirical Risk Minimization. *CoRR*, abs/1502.02268, 2015.

[QSG13] Z. Qin, K. Scheinberg, and D. Goldfarb. Efficient block-coordinate descent algorithms for the group Lasso. *Mathematical Programming Computation*, 5(2):143–169, 2013.

[Ros06] S. M. Ross. *Introduction to Probability Models*. Academic Press Inc, 2006.

[RR95] G. Rothwell and J. Rust. A dynamic programming model of U.S. nuclear power plant, Operations Discussion Paper 410, 1995.

[RT16] P. Richtárik and M. Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1-2, Ser. A):433–484, 2016.

[RV10] M. Rudelson and R. Vershynin. Non-asymptotic theory of random matrices: extreme singular values. In *Proceedings of the International Congress of Mathematicians. Volume III*, pages 1576–1602. Hindustan Book Agency, New Delhi, 2010.

[SA61] H. A. Simon and A. Ando. Aggregation of variables in dynamic systems. *Econometrica*, 29(2):pp. 111–138, 1961.

[SNW12] S. Sra, S. Nowozin, and S.J. Wright. *Optimization for Machine Learning*. Neural information processing series. MIT Press, 2012.

[SS00] A. J. Smola and B. Schökopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 911–918, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[SSBD14] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press, 2014.

[SSJL99] J. Subramanian, S. Stidham Jr., and C. J. Lautenbacher. Airline yield management with overbooking, cancellations, and no-shows. *Transportation Science*, 33(2):147–167, February 1999.

[ST13] A. Saha and A. Tewari. On the nonasymptotic convergence of cyclic coordinate descent methods. *SIAM Journal on Optimization*, 23(1):576–601, 2013.

[Str07] G. Strang. *Computational science and engineering*. Wellesley-Cambridge Press, Wellesley, MA, 2007.

[Stü01] K. Stüben. A review of algebraic multigrid. *Journal of Computational and Applied Mathematics*, 128(1-2):281–309, 2001. Numerical analysis 2000, Vol. VII, Partial differential equations.

[SWH12] C. Schutte, S. Winkelmann, and C. Hartmann. Optimial control of molecular dynamics using markov state models. *Mathematical Programming*, 134(1, Ser. B):259–282, 2012.

[SZ94] S. P. Sethi and Q. Zhang. *Hierarchical decision making in stochastic manufacturing systems*. Birkhauser Verlag, 1994.

[SZ13] T. Sun and C.-H. Zhang. Sparse matrix inversion with scaled lasso. *Journal of Machine Learning Research*, 14:3385–3418, 2013.

[TBI97] L. N. Trefethen and D. Bau III. *Numerical Linear Algebra*. SIAM, 1997.

[TH12] T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

[TOS01] U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, Inc., San Diego, CA, 2001. With contributions by A. Brandt, P. Oswald and K. Stüben.

[Tro11] J. A. Tropp. Improved analysis of the subsampled randomized Hadamard transform. *Advances in Adaptive Data Analysis. Theory and Applications*, 3(1-2):115–126, 2011.

[Tro12] J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

[Tro15] J. A. Tropp. An Introduction to Matrix Concentration Inequalities. *ArXiv e-prints*, January 2015.

[TS86] D.J. Tylavsky and G.R.L. Sohie. Generalization of the matrix inversion lemma. *Proceedings of the IEEE*, 74(7):1050–1052, July 1986.

[Wes92] P. Wesseling. *An introduction to multigrid methods*. Pure and Applied Mathematics (New York). John Wiley & Sons, Ltd., Chichester, 1992.

[WG09] Z. Wen and D. Goldfarb. A line search multigrid method for large-scale nonlinear optimization. *SIAM Journal on Optimization*, 20(3):1478–1503, 2009.

[WS01] C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 682–688. MIT Press, 2001.

[Wu96] Z. Wu. The effective energy transformation scheme as a special continuation approach to global optimization with application to molecular conformation. *SIAM Journal on Optimization*, 6(3):748–768, 1996.

[XN09] D. Xie and Q. Ni. An incomplete Hessian Newton minimization method and its application in a chemical database problem. *Computational Optimization and Applications*, 44(3):467–485, 2009.

[YCYA12] A. Yalaoui, H. Chehade, F. Yalaoui, and L. Amodeo. *Optimization of Logistics*. ISTE. Wiley, 2012.

[YFI89] E. Yamakawa, M. Fukushima, and T. Ibaraki. An efficient trust region algorithm for minimizing nondifferentiable composite functions. *SIAM Journal on Scientific and Statistical Computing*, 10(3):562–580, 1989.

[YHL12] G.-X. Yuan, C.-H. Ho, and C.-J. Lin. An improved glmnet for l1-regularized logistic regression. *Journal of Machine Learning Research*, 13(1):1999–2030, June 2012.

[YZ13] G. Yin and Q. Zhang. *Continuous-time Markov chains and applications*. Springer New York, 3rd edition, 2013.