The School of Mathematics

THE UNIVERSITY
of EDINBURGH

Numerical Methods for L1-Regularized Least-Square Temporal-Difference Learning

by

Alexey Na

Dissertation Presented for the Degree of

MSc in Operational Research

August 2023

Supervised by

Dr Jacek Gondzio

Abstract

Policy Evaluation is a very important techniques for measuring the performance of a policy in Reinforcement Learning. A mature and well-known method is called Least-Squared Temporal Difference (LSTD) Learning, while it is necessary to apply the L1-regularization in the cases of too massive features. There have already been many methods constructed in solving L1-Regularized Problem, varying between first and second order ones. However, their performance is totally different in different problems, especially when the problem to be solve is very ill-conditioned. In this thesis, we will firstly model and formulate regularized LSTD into some typical L1-Regularized Problem. Additionally, two methods (ADMM & IP-PMM) will be applied to the problem and the comparison of their numerical performance will be provided. We will also investigate the ill-conditioned and large-scaled problems.

Acknowledgments

I would like to express my heartfelt gratitude to Jacek Gondzio who has supported me throughout this journey of completing my dissertation. This accomplishment would not have been possible without his unwavering encouragement, guidance, and assistance.

Own Work Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

Alexey Na
Edinburgh, August 21, 2023

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Scope and Outline . . . 1
- 1.3 Preliminaries . . . 2
  - 1.3.1 General Background: Controlled Markov Chain . . . 2
  - 1.3.2 A Brief Introduction to Temporal-Difference (TD) Learning . . . 3

2 Problem Setting and Numerical Approaches **4**
- 2.1 Connection to Supervised Learning: Least-Squared Temporal-Difference Learning . . . 5
- 2.2 Two Types of Convex Optimization Reformulation of L1-Regularization LSTD . . . 7

3 Alternating Direction Method of Multipliers **9**
- 3.1 Notations in ADMM . . . 9
- 3.2 Brief Derivation of ADMM . . . 9
- 3.3 Stopping Criterion and Convergence Analysis for ADMM . . . 10
- 3.4 Convergence Analysis in Ill-Posed Problem . . . 14

4 Interior Point Proximal Method of Multipliers **21**
- 4.1 Notations in IPPMM . . . 21
- 4.2 Brief Explanation of IP-PMM . . . 21
- 4.3 Convergence Analysis in Ill-Posed Problem . . . 23

5 Numerical Experiment **30**
- 5.1 Implementation Details . . . 30
  - 5.1.1 Introduction to IGen2 . . . 30
  - 5.1.2 Introduction to x^* Generators: OsGen & OsGen3 . . . 32
  - 5.1.3 Generate of Non-Trivial Matrix $\tilde{A}$ . . . 33
- 5.2 Warm Up: A very Low-Dimension, Sparse and Well-Conditioned Case . . . 34
- 5.3 Performance in Large-Scale Problems . . . 35
- 5.4 Performance in Ill-Conditioned Problems . . . 37
  - 5.4.1 Experiment 1: Increasing Condition Number of $\tilde{C}^T\tilde{C}$ while Keeping Condition of x^* Fixed . . . 37
  - 5.4.2 Experiment 2: Increasing Condition Number of $\tilde{C}^T\tilde{C}$ with Non-Trivial x^* Constructed . . . 39
- 5.5 Performance in Dense Problems . . . 40

6 Conclusions and Future Work **45**

Appendices **48**

A Convergence Theorem of TD(0)-Learning **48**

B Another Proofs **50**

List of Tables

1 Iteration times and L2-Accuracy $\|\omega^{\infty} - \omega^*\|_2$ v.s three different values of τ 14
2 Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the dimension of variable n. 37
3 Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the condition number of matrix $\tilde{C}^T\tilde{C}$ while maintaining fixed condition of x^* using OsGen1. 39
4 Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the condition number of matrix $\tilde{C}^T\tilde{C}$ with non-trivial construction of x^* by using OsGen3. 40
5 Sparsity control analysis of matrix $\tilde{A}$ as the number of Givens rotation matrices increases. The dimensions of matrix $\tilde{A}$ remain constant at 128×256, while the density of $\tilde{A}$ is measured. The density values (numbers of non-zero entries) are provided as intervals, denoting the range between the lowest and highest values observed over 20 trials for each configuration of rotation matrices. Note that the rotation matrix $\tilde{G}, G, \tilde{G}_2$ and G_2 follows the definition of subsection 5.1.3. 42
6 Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing non-zero entries of matrix $\tilde{C}^T\tilde{C}$ with x^* generated by OsGen1. 44
7 Algorithm Selection Based on Problem Characteristics 45

List of Figures

1 The evolution of $\|\omega^k\|_1$ and $\|\tilde{C}\omega^k + \tilde{d}\|_2$ as algorithm progresses for three different values of $\tau \in \left\{0.5\frac{1}{\lambda_{\max}}, \frac{1}{\lambda_{\max}}, 1.5\frac{1}{\lambda_{\max}}\right\}$ 15
2 Performance of ADMM and IPPMM in a Simple Case 35
3 Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the dimension of variable n. (a) $n = 2^6$. (b) $n = 2^8$. (c) $n = 2^{10}$. (d) $n = 2^{12}$. (e) $n = 2^{14}$. (f) $n = 2^{16}$. 36
4 Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the condition number of matrix $\tilde{C}^T\tilde{C}$ while maintaining fixed condition of x^* using OsGen1. The y-axis is in log-scale. (a) $\kappa(\tilde{C}^T\tilde{C}) = 10^2$. (b) $\kappa(\tilde{C}^T\tilde{C}) = 10^4$. (c) $\kappa(\tilde{C}^T\tilde{C}) = 10^6$. (d) $\kappa(\tilde{C}^T\tilde{C}) = 10^8$. (e) $\kappa(\tilde{C}^T\tilde{C}) = 10^{10}$. (f) $\kappa(\tilde{C}^T\tilde{C}) = 10^{12}$. 38
5 Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the condition number of matrix $\tilde{C}^T\tilde{C}$ with non-trivial construction of x^* by using OsGen3. The y-axis is in log-scale. (a) $\kappa(\tilde{C}^T\tilde{C}) = 10^0$. (b) $\kappa(\tilde{C}^T\tilde{C}) = 10^2$. (c) $\kappa(\tilde{C}^T\tilde{C}) = 10^4$. (d) $\kappa(\tilde{C}^T\tilde{C}) = 10^6$. (e) $\kappa(\tilde{C}^T\tilde{C}) = 10^8$. (f) $\kappa(\tilde{C}^T\tilde{C}) = 10^{10}$. 41
6 Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing non-zero entries of matrix $\tilde{C}^T\tilde{C}$ with x^* generated by OsGen1. The y-axis is in log-scale. (a) $\rho(\tilde{C}) = 3.12\%$. (b) $\rho(\tilde{C}) = 11.88\%$. (c) $\rho(\tilde{C}) = 25.31\%$. (d) $\rho(\tilde{C}) = 40.67\%$. (e) $\rho(\tilde{C}) = 56.20\%$. (f) $\rho(\tilde{C}) = 68.70\%$. 43

1 Introduction

In this section, we will present the motivation, preliminaries, scope and outline of the thesis.

1.1 Motivation

In the realm of Reinforcement Learning, Temporal-Difference (TD) Learning stands out as an efficient and elegant algorithm that employs a technique called Bootstrapping [24]. However, within the context of functional approximation, a notable challenge arises – specifically, when dealing with scenarios involving an extensive set of features. This abundance of features can lead to a compromise in computational efficiency and a reduction in the accuracy of the learning process.

To address this issue, a conventional approach involves the application of l1-regularization, a technique that promotes the generation of a sparse solution by selectively emphasizing essential features. This strategy draws a connection to supervised learning, resulting in what is known as Least-Squared Temporal-Difference (LSTD) Learning. By introducing a regularization term, the formulation expands into L1-Regularized LSTD Learning. Mathematically, this can be formulated using the LASSO form [17]:

$$\omega^* = \arg\min_{\omega \in \mathcal{R}^n} \left\{ \|Y - X\omega\|_2^2 + \lambda \|\omega\|_1 \right\},$$

where

- $Y \in \mathbb{R}^T$ and $X \in \mathbb{R}^{T \times n}$ represent labels and features in the context of supervised learning;
- The objective is to determine the optimal values for ω that best fit the problem, while keeping the magnitude of ω in check using the L1-norm.

In the pursuit of solving this problem, various existing algorithms can be leveraged and compared based on their effectiveness. Nevertheless, there exist quite few of convex optimization methods for solving L1-Regularized LSTD. Additionally, It's worth noting that while ill-posed and large-scale problems are a common challenge, some algorithms may exhibit greater sensitivity and fragility when faced with ill-conditioned or large-scaled matrices compared to others [7].

In this thesis, we undertake a comparison between two methodological categories: first-order and second-order. This comparison draws a parallel to the evaluation of diverse optimization strategies: first-order methods reflect rapid yet imprecise steps, whereas second-order methods embrace a more methodical and calculated approach. Interestingly, our study reveals that when confronted with ill-posed problems, first-order methods exhibit subpar performance [8], in contrast to the surprisingly efficient performance of second-order methods, even within ill-conditioned scenarios. However, it's important to note that while second-order methods demonstrate efficacy in such challenges, they may also introduce the caveat of dramatically increased dimension in their formulations [22], potentially hampering their performance in large-scale problems. This analysis sheds light on the nuanced interplay between algorithmic choices and problem complexity within the domain of numerical methods for L1-Regularized LSTD Learning.

1.2 Scope and Outline

This subsection establishes the foundation for our study. We will outline the organizational structure of the thesis, define the scope of our investigation, and provide a guide to the key themes explored in the upcoming chapters.

- In the next subsection, we will provide preliminaries for this thesis. These are designed to equip readers without knowing the concepts of Markov Decision Processes and Reinforcement Learning.

- In Section 2, we will introduce Least-Squared Temporal-Difference learning in order to set up a connection between TD(0) learning and supervised learning. Subsequently, we will present two different convex formulations. Each of them paves the way for a corresponding numerical algorithm.

- In Section 3, we will begin by deriving a first-order method (ADMM) [27] and [19]. Nonetheless, first-order methods often exhibit poor performance in ill-conditioned problems. We will theoretically establish this concern via the analysis of diagonal matrices.

- After recognizing the limitations of first-order methods, we propose to apply an alternative numerical approach—the Interior-Point Proximal Method of Multipliers (IPPMM) [18]—for solving L1-Regularized LSTD in Section 4. This method, as a representative second-order approach through the utilization of Newton systems, demonstrates robustness in dealing with ill-conditioned problems. Moreover, we will delve into a technique called *dropping variables* in [22], which will be explored as a means to mitigate issues stemming from ill-conditioned scenarios during the iterative processes.

- While we have formulated theorems and lemmas to assess the performance of our methods in ill-conditioned problems, particularly utilizing diagonal matrices in the preceding sections, the need for practical validation remains. Hence, Section 5 is dedicated to a comprehensive series of numerical tests aimed at substantiating the validity of our assumptions and theorems in real-world scenarios.

- Lastly, Section 6 encapsulates the conclusion of this thesis along with a discussion on potential directions for future research.

1.3 Preliminaries

We start with the Controlled Markov Chain like other papers because it is the background for reinforcement learning and all problems set up are based on that.

1.3.1 General Background: Controlled Markov Chain

Denote $s \in \mathcal{S}$ as a state in some state space $\mathcal{S}$, $a \in \mathcal{A}$ as an admissible action in some admissible action space $\mathcal{A}$, and $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ as any policy function to control the Markov Chain and $(\Omega, \mathcal{F}, \mathbf{P})$ as a probability space. Then, the value function for a controlled Markov Process $(X_t^\pi)_{t\in\{0,1,2,\ldots,N\}}$ under the discount criteria is defined on the probability space such that:

$$v(s) := \mathbf{E}\left[\sum_{n=0}^{N} \gamma^n R(\alpha_n, X_n^\alpha)\Big| X_0 = s\right],$$

where

- The Markov Process X_n is controlled by a sequence of actions selected at each time n;

- $\mathbf{E}$ denotes the expectation under the probability measure $\mathbf{P}$ such that

$$\mathbf{E}[f] = \int_\Omega f \cdot d\mathbf{P}$$

- This problem could be considered as either finite-time horizon or infinite-time horizon such that

$$N \in \mathbf{N} \cup \{+\infty\}$$

- The function
$$R : \mathcal{A} \times \mathcal{S} \to \mathcal{R}$$
is called immediate reward function,

- $\gamma \in (0, 1)$ is a discount factor to control the value function convergent by Fatou Lemma, meaning that the objective function is in $L1$-Space:

$$\begin{aligned} v(s) &:= \mathbf{E}\left[\sum_{n=0}^{N} \gamma^n R(\alpha_n, X_n^{\alpha}) \Big| X_0 = s\right] \\ &\leq \|R(\cdot, X)\|_{\infty} \cdot \limsup_{N \nearrow \infty} \int_{\Omega} \sum_{n=0}^{N} \gamma^n d\mathbf{P} \\ &\leq \|R(\cdot, X)\|_{\infty} \cdot \int_{\Omega} \limsup_{N \nearrow \infty} \sum_{n=0}^{N} \gamma^n d\mathbf{P} \\ &\leq \|R(\cdot, X)\|_{\infty} \cdot \frac{1}{1-\gamma} \\ &< \infty. \end{aligned}$$

Note that the first inequality becomes an equation under the infinite-time horizon problem.

1.3.2 A Brief Introduction to Temporal-Difference (TD) Learning

This part will introduce the conventional TD-Learning algorithm. Define the Bellman Residual as the following function:

$$g(v(s)) := v_{\pi}(s) - \mathcal{T}_{\pi} v(s) = v_{\pi}(s) - \mathbf{E}[R + \gamma v_{\pi}(S')|s],$$

where $\mathcal{T}_{\pi} v : \mathcal{R} \to \mathcal{R}$ is called Bellman Operator and essentially a contraction mapping (The proof is widely accessible: For instance, it can be found in the section *Method of Successive Approximations* in [6]). Trivially, we observe that the Bellman Equation for optimal policy π^* satisfies $g(v_{\pi^*}(s)) = 0$. In a trajectory, we obtain an immediate reward after a step r, and next state hit s'. Therefore, we have a noisy observation of the Bellman Residual:

$$\tilde{g}(v_{\pi}(s)) = v_{\pi}(s) - [r + \gamma v_{\pi}(s')] = \left(v_{\pi}(s) - \mathbf{E}[R + \gamma v_{\pi}(S')|s]\right) + \left(\mathbf{E}[R + \gamma v_{\pi}(S')|s] - [r + \gamma v_{\pi}(s')]\right).$$

For solving the optimal policy where the Bellman Residual is equal to zero, we can apply Robbins–Monro algorithm by using $\tilde{g}(v_{\pi}(s))$:

$$\bar{v}_{k+1}(s) = v_k(s) - \alpha_k \tilde{g}(v_k(s)) = v_k(s) - \alpha_k [r + \gamma v_{\pi}(s')]. \tag{1.1}$$

One can show that after infinite numbers of iterations denoted by $k \in \mathbf{N}$ (hence it is necessary to visit all states infinitely many times).

$$\lim_{k \nearrow \infty} \bar{v}_k(s) \to v^*(s), \forall s \in \mathcal{S}.$$

[The convergence theorem can be found in Appendix A]

2 Problem Setting and Numerical Approaches

The development of Least-Squared Temporal-Difference (LSTD) Learning has been extensively explored in the works of [23], [3], and [16] over several years, establishing a crucial link between reinforcement learning and supervised learning. However, when considering most scenarios of TD learning involving linear functional approximation, a challenge arises due to the presence of a significantly large feature set. In such cases, the application of L1-regularization techniques becomes relevant for the purpose of selecting pertinent features and mitigating over-fitting concerns. This introduces the specific numerical problem we aim to address: L1-Regularized LSTD, as highlighted in [15], [17], and [25]. It is noteworthy that a notable portion of these approaches having been developed, for example in [10] and [12], directly employ numerical methods to tackle the fixed-point problem. An exception to this trend is found in the work of Qin [19], who devises a solution through convex optimization. Remarkably, Qin's approach pioneers the fusion of a first-order method (ADMM) with regularized LSTD, marking a distinctive departure from the conventional methodologies. On the other hand, the first-order method has a clear shortage which has been discussed in previous section. In order to improve the robustness of the regularized LSTD algorithm, we propose to apply the second-order method for it. There are also many of them having been constructed, for example [9], [8], [5] and [18]. Due to limit of the time, we will focus on an algorithm called Interior-Point Proximal Method of Multipliers (IPPMM) in this thesis, which is well developed by Gondzio, Pougkakiotis and others in [18] and [22]. Note that the later is regarding the application of IPPMM in different scenarios, and we are particularly inspired by Section 4 of it, then applying in solving L1-Regularized LSTD. Eventually, [7] provides a remarkable approach for generating instances in numerical experiments to validate our theorems proposed, examining and comparing the performance of two numerical algorithms in different challenging problems.

2.1 Connection to Supervised Learning: Least-Squared Temporal-Difference Learning

In order to choose relevant features effectively, we utilized a widely recognized method known as L1-Regularization. To implement this approach, it was necessary to reconfigure the methodology of TD(0) learning to mirror the framework of supervised learning. Our initial step involved employing linear functional approximation on the value function. This function naturally exists within the span of n basis functions denoted as: $\mathcal{H}(s) := span\{\phi_1(s), \phi_2(s), ..., \phi_n(s)\} \subsetneq \mathcal{R}^n$ for any $s \in \mathcal{S}$. This can be expressed as:

$$\tilde{V}(s) := \sum_{i=1}^{n} \omega_i \phi_i(s) = \omega^T \phi(s),$$

where $\omega \in \mathcal{R}^n$ and $\phi(s) = [\phi_1(s), \phi_2(s), ..., \phi_n(s)]^T \in \mathcal{R}^n$, and $\phi : \mathcal{S} \to \mathcal{R}^n$. Consider the Bellman Operator under an arbitrary policy $\pi \in \Pi$:

$$(\mathcal{T}_\pi \tilde{V})(s) := r(s) + \int_{\mathcal{S}\times\mathcal{A}} \tilde{V}(s') d\mathbf{P}[s'|s, \pi(s,a)],$$

which might not necessarily lie within the span of $\{\phi_1, \phi_2, ..., \phi_n\}$. This is due to the integral over the entire state space $\mathcal{S}$ and action space $\mathcal{A}$ under the measure $\mathbf{P}$. To bring it back to the spanned space, we introduce a projection operator $\tilde{\Pi}$ proposed in [10]:

$$(\tilde{\Pi}\mathcal{T}_\pi \tilde{V})(s) = \arg\min_{h\in\mathcal{H}} \|\mathcal{T}_\pi \tilde{V}(s) - h\|_D^2, \tag{2.1}$$

where $D \in \mathcal{R}^{|S|\times|S|}$ represents the initial distribution of the Markov Process $\{X_t\}_t$. In this formulation, we aim to find the optimal point within the spanned space of $\mathcal{H}$ that effectively minimizes the distance to the Bellman operator—essentially encapsulating the concept of a Projection Operator. The D-norm is equivalent to L2-norm as the number of samples goes to infinity [15]. This results in a fixed-point problem where the Projected Bellman Residual is minimized in L2-norm:

$$\tilde{V}^* = \arg\min_{\tilde{V}\in\mathcal{H}} \|\tilde{V} - \tilde{\Pi}\mathcal{T}_\pi \tilde{V}\|_2^2, \quad \Pi\mathcal{T}_\pi \tilde{V} = \arg\min_{h\in\mathcal{H}} \|\mathcal{T}_\pi \tilde{V} - h\|_2^2, \tag{2.2}$$

which follows the formulation in both [10] and [12]. In the context of TD-learning, the Bellman Operator (referred to as the probability measure $\mathbf{P}$ in the context of the Markov Chain) remains unknown. Instead, we opt to utilize the next sampled transition for addressing this aspect. Following Theorem will derive the specific form of Projection Operator:

Theorem 2.1. *Problem (2.2) can be equivalently reformulated as a fixed-point problem when a linear function approximator is applied:*

$$\omega^* = \arg\min_{\omega\in\mathcal{R}^n} \|\tilde{R} + \gamma\tilde{\Phi}'\omega^* - \tilde{\Phi}\omega\|_2,$$

where $\tilde{\Phi} = [\phi^T(s_1), \phi^T(s_2), ..., \phi^T(s_T)]^T \in \mathcal{R}^{T\times n}$, $\tilde{\Phi}' = [\phi^T(s'_1), \phi^T(s'_2), ..., \phi^T(s'_T)]^T \in \mathcal{R}^{T\times n}$, *and* $\tilde{R} = [r(1), r(2), ..., r(T)]^T \in \mathcal{R}^T$. *Here,* $\{s_1, s_2, ..., s_T\}$ *represents a sequence of state samples, and* $\{s'_1, s'_2, ..., s'_T\}$ *represents a sequence of subsequent samples within a transition. Moreover, the theoretical solution satisfies the equation*

$$\tilde{\Phi}\omega^* = \Pi\left(\tilde{R} + \gamma\tilde{\Phi}'\omega^*\right), \tag{2.3}$$

where $\Pi := \tilde{\Phi}(\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T$ *is the explicit form of* ***Projection Operator*** *for Bellman Operator and* $\tilde{\Phi}\omega$ *is called* ***Projected Bellman Operator****.*

Proof. We verify by the following three steps:

- Rewrite Problem (2.2) in terms of function approximation:

$$\theta^* := \arg\min_{\theta\in\mathcal{R}^n} \|\tilde{R} + \gamma\tilde{\Phi}'\omega_1^* - \tilde{\Phi}\theta\|_2^2,$$

  with $\omega_1^* := \arg\min_{\omega_1\in\mathcal{R}^n} \|\tilde{\Phi}\theta^* - \tilde{\Phi}\omega_1\|_2^2$. We can write Problem (2.2) in this form because $h \in \mathcal{H}$ implies the Projected Bellman Operator must stay at the span space $\mathcal{H}$ such that is a linear combination of $\phi^T(s_1), ..., \phi(s_T)^T$. Solving the first optimization problem, we get

$$(\tilde{\Phi}^T\tilde{\Phi})\theta^* = \tilde{\Phi}^T(\tilde{R} + \gamma\tilde{\Phi}'\omega_1^*) \Longrightarrow \theta^* = (\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T(\tilde{R} + \gamma\tilde{\Phi}'\omega_1^*).$$

  Subsequently, we can get the same result for ω_1^*

$$\tilde{\Phi}^T\tilde{\Phi}\omega_1^* = (\tilde{\Phi}^T\tilde{\Phi})(\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T(\tilde{R} + \gamma\tilde{\Phi}'\omega_1^*) \Longrightarrow \omega_1^* = (\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T(\tilde{R} + \gamma\tilde{\Phi}'\omega_1^*),$$

  which yields a solution that satisfies the equation (2.3).

- Consider the equivalent form in Theorem 4.1:

$$\omega_2^* = \arg\min_{\omega_2\in\mathcal{R}^n} \|\tilde{R} + \gamma\tilde{\Phi}'\omega_2^* - \tilde{\Phi}\omega_2\|_2 \Longrightarrow \omega_2^* = (\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T(\tilde{R} + \gamma\tilde{\Phi}'\omega_1^*).$$

  This is exactly the same result as that in the last bullet point.

- Thus, these two forms of formulations provide precisely the same analytical solution. Pre-multiplying $\tilde{\Phi}$, we observe that $\tilde{\Phi}\omega$ is equal to $\Pi := \tilde{\Phi}(\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T$ multiplied by Bellman Operator. Thus, we conclude that Π is the Bellman Operator we would like to derive.

□

Remark 1. *Not like TD(0)-learning, it's important to note that the sample sequence* $\{s_t\}_{t\in\{1,...,T\}}$ *does not necessarily follow a single trajectory.*

Finally, we will further formulate the equation (2.3) into the form of Linear Regression, which is a conventional model of supervised learning. In this form, we can take the L1-Regularization into account and develop numerical methods for solving it. Isolate the variable ω^* by reordering equation (2.3):

$$\underbrace{\tilde{\Pi}\tilde{R}}_{\text{Labels}} = \underbrace{(\tilde{\Phi} - \tilde{\Pi}\gamma\tilde{\Phi}')}_{\text{Features}}\omega^*.$$

2.2 Two Types of Convex Optimization Reformulation of L1-Regularization LSTD

In the preceding section, we formulated the fixed-point problem for Least-Square Temporal-Difference learning (LSTD) using a projection operator. Notably, only the Projected Bellman Residual (PBR) can be cast into a convex optimization problem, as the Bellman Residual (BR) is inherently non-learnable, as noted in [25]. Specifically, we introduced L1-Regularization to the weight vector ω, following the approach proposed in [19]:

$$\min_{\omega \in \mathcal{R}^n} \left\{ \|\omega\|_{\mathbf{L1}} : \|\tilde{\Pi}\tilde{R} + (\gamma \tilde{\Pi}\tilde{\Phi}' - \tilde{\Phi})\omega\|_{\mathbf{L2}} \leq \epsilon \right\},\ \epsilon \in \mathcal{R}^+,$$

where we can denote $\tilde{d} := \tilde{\Pi}\tilde{R},\ \tilde{C} := \gamma\tilde{\Pi}\tilde{\Phi}' - \tilde{\Phi}$ and then Problem (4.3) becomes

$$\min_{\omega \in \mathcal{R}^n} \left\{ \|\omega\|_{\mathbf{L1}} : \|\tilde{d} + \tilde{C}\omega\|_{\mathbf{L2}} \leq \epsilon \right\},\ \epsilon \in \mathcal{R}^+. \tag{2.4}$$

In this problem, we keep the PBR extremely small in the L2-norm in order to approximately attain the fixed-point ω^* in equation (2.3), while also minimizing the L1-norm of the variable ω for the purpose of selecting *contributory features*. The Problem (2.4) is in the form of **Basis Pursuit Denoising Problem**, which has an equivalent form (**LASSO**):

$$\min_{\omega \in \mathcal{R}^n} \left\{ \frac{1}{2}\|\tilde{d} + \tilde{C}\omega\|^2_{\mathbf{L2}} + \lambda\|\omega\|_{\mathbf{L1}} \right\},\ \lambda \in \mathcal{R}^+ \tag{2.5}$$

Theorem 2.2. *(ADMM Formulation) Problem (2.4) is a convex optimization problem.*

Proof. This result is quite straightforward with following two steps:

- Step 1: The objective function is L1-norm and thus convexity is straightforward. Define $f := \lambda\omega_1$ and $g := (1-\lambda)\omega_2$ where $\lambda \in [0,1]$ and $\omega_1, \omega_2 \in \mathcal{R}^n$. Clearly these two functions are both in L1-space: $f, g \subset L^1(\mathcal{R}^n)$. By Minkowski inequality, we have

$$\|\lambda\omega_1 + (1-\lambda)\omega_2\|_{\mathbf{L1}} := \|f+g\|_{\mathbf{L1}} \leq \|f\|_{\mathbf{L1}} + \|g\|_{\mathbf{L1}} = \lambda\|\omega_1\|_{\mathbf{L1}} + (1-\lambda)\|\omega_2\|_{\mathbf{L1}}$$

- Step 2: The inequality constraint is L2-norm subtracting a small constant. we can also easily verify this is a convex function. Similarly as above, define $h := \|\tilde{d} + \tilde{C}\omega\|_{\mathbf{L2}} - \epsilon$. We have for any $\lambda \in [0,1]$:

$$\begin{aligned} h(\lambda\omega_1 + (1-\lambda)\omega_2) &:= \|\tilde{d} + \tilde{C}[\lambda\omega_1 + (1-\lambda)\omega_2]\|_{\mathbf{L2}} - \epsilon \\ &\leq \|\lambda\tilde{d} + \tilde{C}\lambda\omega_1\|_{\mathbf{L2}} - \lambda\epsilon + \|(1-\lambda)\tilde{d} + \tilde{C}(1-\lambda)\omega_1\|_{\mathbf{L2}} - (1-\lambda)\epsilon \\ &:= \lambda h(\omega_1) + (1-\lambda)h(\omega_2) \end{aligned}$$

□

Theorem 2.3. *(IP-PMM Formulation) Problem (2.5) can be reformulated into a convex quadratic programming problem.*

Proof. Assign $u := \tilde{C}\omega \in \mathcal{R}^T$, $\omega := \omega^+ - \omega^-$, $M := T$ and $N := T + 2n$. Then, the problem is in the following form:

$$\begin{array}{lrcll} \min\limits_{x} & \frac{1}{2}x^T Q x + c^T x & & & \\ \text{s.t.} & Ax & = & b & \\ & x_i & \geq & 0 & \forall i \in \mathcal{I} := \{M+1, ..., N\} \end{array} \tag{2.6}$$

where we have $b = \mathbf{0}_M$, $\omega^+ = \max\{\omega, 0\}$, $\omega^- = \max\{-\omega, 0\}$, $x = \left[u^T, (\omega^+)^T, (\omega^-)^T\right]^T \in \mathcal{R}^N$, $c = \left[(\tilde{d})^T, \lambda e_\omega^T, \lambda e_\omega^T\right]^T \in \mathcal{R}^N$ and

$$Q = \begin{bmatrix} I_T & \mathbf{0}_{T\times 2n} \\ \mathbf{0}_{2n\times T} & \mathbf{0}_{2n\times 2n} \end{bmatrix} \in \mathcal{R}^{N\times N}, \quad A = \begin{bmatrix} -I_T & \tilde{C} & -\tilde{C} \end{bmatrix} \in \mathcal{R}^{M\times N}$$

Now, the objective function will be

$$\begin{aligned} \min_{u,\omega}\left\{\frac{1}{2}u^T I u + d^T u + \lambda e^T\omega^+ - \lambda e^T\omega^-\right\} &= \min_{\omega}\left\{\frac{1}{2}[(\tilde{C}\omega)^T(\tilde{C}\omega) + 2\tilde{d}^T(\tilde{C}\omega) + \tilde{d}^T\tilde{d}] - \tilde{d}^T\tilde{d} + \lambda(\omega^+ - \omega^-)\right\} \\ &= \min_{\omega}\left\{\frac{1}{2}(\tilde{d} + \tilde{C}\omega)^T(\tilde{d} + \tilde{C}\omega) + \lambda\|\omega\|_{\mathbf{L1}}\right\} \\ &= \min_{\omega}\left\{\frac{1}{2}\|\tilde{d} + \tilde{C}\omega\|_{\mathbf{L2}}^2 + \lambda\|\omega\|_{\mathbf{L1}}\right\} \end{aligned}$$

with constraints $\omega^+, \omega^- \geq 0$ and

$$\begin{aligned} Ax &= -u + \tilde{C}(\omega^+) - \tilde{C}(\omega^-) \\ &= -u + \tilde{C}\omega \\ &= 0 \end{aligned}$$

Hence, the problem (2.6) is equivalent to problem (2.5). Combined with Theorem (2.2), we end up with a convex quadratic programming problem. □

3 Alternating Direction Method of Multipliers

A simple method for solving problem (2.4) has been proposed in [27]. In many cases, especially when the problem is not too ill-posed, this first-order method has an apparent advantage that is simple to implement and also quite efficient. Note that this algorithm basically needs only three steps of update within each single iteration.

3.1 Notations in ADMM

In this subsection, we will introduce key notations used throughout this section.

- Primal variable: ω, Dual variable: y, and auxiliary variable: r;
- The subscripts in each vector denotes the j^{th} entry of that. For instance, ω_j, y_j and r_j are the j^{th} entry of variable ω^k, y^k, r^k respectively;
- The superscripts in variables ω^k, y^k, r^k denote the index of iterations;
- $\Delta\omega^k, \Delta y^k$ and Δr^k represent for the difference between the k^{th} and $k-1^{st}$ iteration w.r.t. each variable;
- ϵ denotes the tolerance for $\|r\|$, but ϵ_T denotes the tolerance for terminal criteria;
- For simplicity, we use $\lambda_{\max} := \lambda_{\max}(\tilde{C}^T\tilde{C})$ where $\lambda_{\max}(\cdot)$ is the maximal eigenvalue of a matrix;
- ∂ denotes for the sub-derivative operator on a function;
- L^k denotes the lower bound of l2-distance between ω^k and ω^*; L_b^k denotes the lower bound of l2-distance between ω_b^k and ω_b^* at k^{th} iteration where $b \in \{1, 2, ..., n\}$ is an index of entry for vector ω;
- ω_B denotes a sub-vector of ω such that $\omega_B \subset \omega$ where B is a set for indices;
- $\kappa(\cdot)$ denotes the condition number for a matrix;
- Shrink$(\cdot)$ denotes the Soft Thresholding Operator such that:

$$\text{Shrink}_a(x) := \begin{cases} x - a & \text{if} x > a \\ 0 & \text{if} |x| < a \\ x + a & \text{if} x < -a \end{cases}$$

3.2 Brief Derivation of ADMM

Since this algorithm has already been developed in [19] and [27], we only provide a very brief derivation of it. Begin with general steps for Inexact ADMM, firstly split variables

$$\min_{\omega \in \mathcal{R}^n, r \in \mathcal{R}^T} \left\{ \|\omega\|_1 : \tilde{C}\omega + \tilde{d} = r, \|r\| \leq \epsilon \right\}, \tag{3.1}$$

and write down the augmented Lagrangian sub-problem controlled by parameter μ

$$L^\mu(\omega, r, y) = \min_{\omega \in \mathcal{R}^n, r \in \mathcal{R}^T} \left\{ \|\omega\|_1 - y^T(\tilde{C}\omega + \tilde{d} - r) + \frac{\mu}{2}\|\tilde{C}\omega + \tilde{d} - r\|^2 : \|r\| \leq \epsilon \right\}, \mu > 0, \epsilon > 0.$$

Derive the following inexact alternating minimization for each sub-problem:

- For $\omega = \omega^k$, $y = y^k$ fixed, the sub-problem for variable r is given by:

$$\min_{r\in\mathcal{R}^T}\left\{\frac{\mu}{2}\left\|r-(\tilde{C}\omega+\tilde{d}-\frac{1}{\mu}y)\right\|^2 : \|r\|\le\epsilon\right\},$$

which gives the minimizer of variable r:

$$r^{k+1}=\mathcal{P}_{\mathbf{B}_\epsilon}\left(\tilde{C}\omega^k+\tilde{d}-\frac{1}{\mu}y^k\right), \mathbf{B}_\epsilon := \left\{\xi\in\mathcal{R}^T : \|\xi\|\le\epsilon\right\},$$

where $\mathcal{P}_{\mathbf{B}_\epsilon}(\cdot)$ is the orthogonal projection operator for plane $\mathbf{B}_\epsilon$ in L2-norm such that

$$\mathcal{P}_{\mathbf{B}_\epsilon}(\cdot)=\arg\min_{\beta\in\mathbf{B}_\epsilon}\{\|\cdot-\beta\|\}=\begin{cases}\tilde{C}\omega+\tilde{d}-\frac{1}{\mu}y & \text{if } \left\|\tilde{C}\omega+\tilde{d}-\frac{1}{\mu}y\right\|\le\epsilon\\ \epsilon\cdot\frac{\tilde{C}\omega+\tilde{d}-\frac{1}{\mu}y}{\left\|\tilde{C}\omega+\tilde{d}-\frac{1}{\mu}y\right\|} & \text{if } \left\|\tilde{C}\omega+\tilde{d}-\frac{1}{\mu}y\right\|>\epsilon\end{cases}$$

- For $r = r^{k+1}$, $y = y^k$ fixed, the sub-problem for variable ω is given by:

$$\min_{\omega\in\mathcal{R}^n}\left\{\|\omega\|_1+\frac{\mu}{2}\left\|\tilde{C}\omega+\tilde{d}-r-\frac{1}{\mu}y\right\|^2\right\}\Rightarrow\min_{\omega\in\mathcal{R}^n}\left\{\|\omega\|_1+\mu\left((g^k)^T(\omega-\omega^k)+\frac{1}{2\tau}\|\omega-\omega^k\|^2\right)\right\} \tag{3.2}$$

where $g=\tilde{C}^T(\tilde{C}\omega^k+\tilde{d}-r^{k+1}-\frac{1}{\mu}y^k)\in\mathcal{R}^T$ denotes for the gradient w.r.t. ω in the quadratic term at iteration k without considering the parameter μ, and τ is a step size parameter which can be found in section 2.2 of [27]. Then, we solve the RHS instead and it gives the minimizer of variable ω:

$$\omega^{k+1}=\text{Shrink}\left(\omega^k-\tau g^k,\frac{\tau}{\mu}\right):=\max\left\{|\omega^k-\tau g^k|-\frac{\tau}{\mu},0\right\}\cdot\frac{\omega^k-\tau g^k}{|\omega^k-\tau g^k|}.$$

Note that the RHS in **problem (3.2) is an approximator of the LHS** and further interpretation could be found in Theorem B.1 of Appendix B.

- For $r = r^{k+1}$, $\omega = \omega^{k+1}$ fixed, the update rule for dual variable y is straightforward:

$$y^{k+1}=y^k-\mu(\tilde{C}\omega^{k+1}+\tilde{d}-r^{k+1}),$$

such that the second dual feasibility constraint is guaranteed:

$$0=-I^T[y^k-\mu(\tilde{C}\omega^{k+1}+\tilde{d}-r^{k+1})]=-I^Ty^{k+1},$$

where the objective function does not involve variable r.

3.3 Stopping Criterion and Convergence Analysis for ADMM

Then, the criterion of convergence is arguably based on the ϵ_T-optimal primal and dual gap measured by L2-norm:

$$\max\left\{\|\omega^{k+1}-\omega^k\|_2,\|y^{k+1}-y^k\|_2\right\}\le\epsilon_T.$$

Remark 2. *Stopping criteria stated above might not include the primal variable ω in practice because*

it will take incredibly large numbers of iterations to attain an optimum, especially when the problem is ill-conditioned. That means when variables r and y achieve the optimality, variable ω may not convergence and this will prevent the iteration from terminating for a very long time. Alternatively, one can set the maximal iterations instead.

Remark 3. *In most scenarios of BPDN, $\epsilon > 0$ is a very small number, and this will lead to*

$$\|r^{k+1} - r^k\|_2 \leq \|r^{k+1}\| + \|r^k\|_2 \leq 2\epsilon$$

during the iterative process. Therefore, stopping criteria can exclude this variable for non-triviality.

Theorem 3.1. *(Convergence Theorem for ADMM) Let $\lambda_{\max}$ be the maximal eigenvalue of $\tilde{C}^T\tilde{C}$ where $\tilde{C}$ is a well-conditioned matrix such that $\kappa\left(\tilde{C}^T\tilde{C}\right) < \infty$. If $\tau > 0$ such that*

$$\tau\lambda_{\max} < 1$$

then the sequence of variables $\{(r^k, \omega^k, y^k)\}$ converges to $\{(r^*, \omega^*, y^*)\}$ for any $\mu > 0$, where (r^*, ω^*) solves the problem (3.1).

An analogous proof can be found in the paper [27].

Proof. Derive the Lagrangian dual for problem (3.1):

$$L(r, \omega, y) := \max_{y \in \mathcal{R}^T} \min_{\omega \in \mathcal{R}^n, r \in \mathcal{R}^T} \left\{ \|\omega\|_1 - y^T(\tilde{C}\omega + \tilde{d} - r) : \|r\| \leq \epsilon \right\},$$

where we have two optimization sub-problems which are minimization in ω and minimization in r:

$$L^\omega := \min_{\omega \in \mathcal{R}^n} \left\{ \|\omega\|_1 - y^T\tilde{C}\omega \right\}, \;\; L^r := \min_{r \in \mathcal{R}^T} \left\{ y^T r : \|r\| \leq \epsilon \right\},$$

where we can use Lagrangian properties to write down the optimality conditions:

$$0 \in \partial_\omega L^\omega = \partial\|\omega\|_1 - \tilde{C}^T y, \;\; 0 \in \partial_r L^r = y - \partial f(r).$$

By the first-order condition of the BPDN formulated above, we have:

- Optimality in variable ω: $\tilde{C}\omega^* + \tilde{d} - r^* = 0$
- Optimality in variable y: $\tilde{C}^T y^* \in \partial\|\omega^*\|_1$
- Optimality in variable r: $y^* \in \partial f(r^*)$.

Next, consider the update for variable r^k. Write down the first order optimality condition for updating variable r^k:

$$0 \in \partial_r L = y + \mu(r - \tilde{C}\omega - \tilde{d}) + \partial f(\|r\| - \epsilon),$$

where we convert the problem into an unconstrained one with $f(\|r\| - \epsilon)$ being a convex penalty function for minimization such that

$$f(\cdot) = \begin{cases} \infty & \text{if } \|r\| - \epsilon > 0 \\ 0 & \text{if } \|r\| - \epsilon \leq 0. \end{cases}$$

Combine with the optimality condition in variable r, we have $\mu(r^{k+1} - \tilde{C}\omega^k - \tilde{d}) + y^k \in \partial f(r^{k+1})$ and $y^* \in \partial f(r^*)$. Then, the following inequality

$$\begin{aligned} &(r^{k+1} - r^*)[y^{k+1} - y^* + \mu(r^{k+1} - \tilde{C}\omega^{k+1} - \tilde{d}) - \mu(r^{k+1} - \tilde{C}\omega^k - \tilde{d})] \\ &= (r^{k+1} - r^*)[y^{k+1} - y^* + \mu\tilde{C}(\omega_{j+1} - \omega_j)] \geq 0 \end{aligned} \tag{3.3}$$

is obtained by monotonic operator together with the update condition $y^{k+1} = y^k - \mu(\tilde{C}\omega^{k+1} + \tilde{d} - r^{k+1})$. Similarly, by applying the update condition for ω^k and optimality condition for variable y we have $\frac{1}{\mu}[\omega^k - \omega^{k+1} - \tau g^k] \in \partial\|\omega^{k+1}\|_1$ and $\tilde{C}^T y^* \in \partial\|\omega\|_1$. Then, the inequality is followed with $\tau > 0$:

$$\begin{aligned} &\tau(\omega^{k+1} - \omega^*)\left[\frac{\mu}{\tau}[\omega^k - \omega^{k+1} - \tau g^k] - \tilde{C}^T y^*\right] \\ &= \tau(\omega^{k+1} - \omega^*)\left[\frac{\mu}{\tau}(\omega^k - \omega^{k+1}) - \mu\tilde{C}^T(\tilde{C}\omega^k + \tilde{d} - r^{k+1} - \frac{1}{\mu}y^k) - \tilde{C}^T y^*\right] \\ &= \tau(\omega^{k+1} - \omega^*)\left[\frac{\mu}{\tau}(\omega^k - \omega^{k+1}) - \mu\tilde{C}^T\left(\tilde{C}\omega^k + \tilde{d} - r^{k+1} - \frac{1}{\mu}y^{k+1} - (\tilde{C}\omega^{k+1} + \tilde{d} - r^{k+1})\right) - \tilde{C}^T y^*\right] \\ &= \tau(\omega^{k+1} - \omega^*)\left[\frac{\mu}{\tau}(\omega^k - \omega^{k+1}) - \mu\tilde{C}^T\tilde{C}(\omega^k - \omega^{k+1}) + \tilde{C}^T(y^{k+1} - y^*)\right] \geq 0 \end{aligned} \tag{3.4}$$

Plugging the equation (3.3) into the inequality (3.4) and eliminating the auxiliary variable r, the following inequality will be constructed:

$$\begin{aligned} &\frac{\mu}{\tau}(\omega^{k+1} - \omega^*)(\omega^k - \omega^{k+1}) + (\omega^{k+1} - \omega^*)\left[-\mu\tilde{C}^T\tilde{C}(\omega^k - \omega^{k+1}) + \tilde{C}^T(y^{k+1} - y^*)\right] \\ &= \frac{\mu}{\tau}(\omega^{k+1} - \omega^*)(\omega^k - \omega^{k+1}) + \frac{1}{\mu}(y^{k+1} - y^k)(y^{k+1} - y^*) - (y^{k+1} - y^k)C(\omega^{k+1} - \omega^k) \geq 0 \end{aligned}$$

As the consequence, the following inequality holds by exactly the same reasoning as $(A.8) - (A.10)$ in [27]:

$$\|u^k - u^*\|_G^2 - \eta\|u^k - u^{k+1}\|_G^2 \geq \|u^{k+1} - u^*\|_G^2, \tag{3.5}$$

where $u := \begin{bmatrix} \omega \\ y \end{bmatrix}$, $u^* := \begin{bmatrix} \omega^* \\ y^* \end{bmatrix}$ and $G := \begin{bmatrix} \frac{\mu}{\tau}I & \\ & \frac{1}{\mu}I \end{bmatrix}$.

Then, we could construct the three results of $(A.11)$ in [27]:

- (a) $\|u^{k+1} - u^k\|_G \to 0$: This can be proven by contradiction: assume the sequence failed to

converge to 0 as $k \to \infty$. By induction, we have:

$$\lim_{k\nearrow\infty} \|u^{k+1} - u^*\|_G^2 \leq \lim_{k\nearrow\infty} \|u^0 - u^*\|_G^2 - \sum_{i=1}^{k} \eta \|u^i - u^{i+1}\|_G^2 \to -\infty$$

which is a contradiction of the definition of the norm. Thus, $\|u^{k+1} - u^k\|_G^2 \to 0$.

- (b) The non-increasing property and convergence of the sequence $\|u^{k+1} - u^*\|_G^2$ is followed by equation (3.5):

$$\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \geq \eta \|u^k - u^{k+1}\|_G^2 \geq 0,$$

which implies the value of $\|u^k - u^*\|_G^2$ will not increase as k increases.

- (c) $\{u^k\}$ lies in a compact region: By definition of weighted norm, we have:

$$\left\| \min\left\{\frac{\mu}{\tau}, \frac{1}{\mu}\right\} (u^0 - u^*) \right\|_2^2 \leq \|u^0 - u^*\|_G^2 = \left\| \sqrt{G}(u^0 - u^*) \right\|_2^2 \leq \lambda_{\max}(\sqrt{G}) L,$$

which implies

$$u^0 \in \mathbf{B}_{u^*} \left(\frac{\sqrt{\lambda_{\max}(\sqrt{G}) L}}{\min\left\{\frac{\mu}{\tau}, \frac{1}{\mu}\right\}} \right).$$

Using the non-increasing property stated in (b), we have obtained the maximal neighbourhood which is a close and bounded δ-Ball in $\mathcal{R}^{n+T}$ space:

$$\min\left\{\frac{\mu}{\tau}, \frac{1}{\mu}\right\} \cdot \left\| (u^k - u^*) \right\|_2^2 \leq \|u^k - u^*\|_G^2 \leq \lambda_{\max}(\sqrt{G}) L, \forall k \in N.$$

Thus, by Heine-Borel theorem, we can argue that the δ-Ball is a compact region in real space $\mathcal{R}^{n+T}$.

Eventually, rest of the proof is the same as [27].

□

Remark 4. *Note that the performance of ADMM algorithm heavily depends on hyper-parameter, especially τ. Moreover, penalty parameter μ cannot be too small or large, otherwise*

$$y^{k+1} := \lim_{\mu\searrow 0} y^k - \mu(\tilde{C}\omega^{k+1} + \tilde{d} - r^{k+1}) = y^k$$

$$r^{k+1} := \lim_{\mu\nearrow\infty} \mathcal{P}_{\mathbf{B}_\epsilon} \left(\tilde{C}\omega^k + \tilde{d} - \frac{1}{\mu} y^k \right) = \mathcal{P}_{\mathbf{B}_\epsilon} \left(\tilde{C}\omega^k + \tilde{d} \right)$$

Example 1. *In this example, we will demonstrate the sensitivity of step parameter τ. We performed this experiment using Python 3.9.12 on Anaconda Environment. The details for matrices and parameters will be listed below:*

ADMM Solver Performance		
Parameters	Iterations	Accuracy
$\tau_1 = 0.5\frac{1}{\lambda_{\max}}$	4515	0.121667
$\tau_2 = \frac{1}{\lambda_{\max}}$	3095	0.063906
$\tau_3 = 1.5\frac{1}{\lambda_{\max}}$	∞	∞

Table 1: Iteration times and L2-Accuracy $\|\omega^{\infty} - \omega^*\|_2$ v.s three different values of τ

- *Matrices:* $\tilde{C} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & -0.1 \end{bmatrix}, \tilde{d} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$
- *Starting points:* $\omega^0 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, r^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$, *and* $y^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$
- *Hyper-parameters:* $\epsilon = 0.05, \mu = 1$, *and tolerance* 10^{-3}
- *In this problem setting, we can easily find the optimal solution is* $\omega^* = \mathbf{0}$.

In Figure 1 and Table 1, we report on the behavior of ADMM with three groups of τ. *As we can see the group where* $\tau_2 = \frac{1}{\lambda_{\max}}$ *converges the fastest. The number of iterations increases by* 1500 *if we set* $\tau_1 = 0.5\frac{1}{\lambda_{\max}}$. *Even worse, when the value of that is slightly larger* (5×10^{-5}) *than the threshold in Theorem 3.1, the ADMM solver cannot converge to the optimal solution. Similar observation can be made about the accuracy: the second group is closest to the optimal solution while the first displays a two times larger l2 error. As the consequence of divergent solution, group three failed to attain the optimality.*

3.4 Convergence Analysis in Ill-Posed Problem

In this section, we will initially discuss trivial cases to investigate how the hyper-parameter τ adversely effects the convergence of primal variable ω. Before building up the proofs, there are some assumptions which will be followed in the whole sub-section:

Assumption 1. *The starting point of each variable will be zero as normally chosen:*

$$(r^0, \omega^0, y^0) = (0, 0, 0)$$

Assumption 2. *There exist noticeable distances from initial points to optimal solutions:*

$$\left\|\omega^* - \omega^0\right\|_2 \geq L, \left\|y^* - y^0\right\|_2 \geq L^y$$

such that $\min\{L, L^y\} > \epsilon_T$.

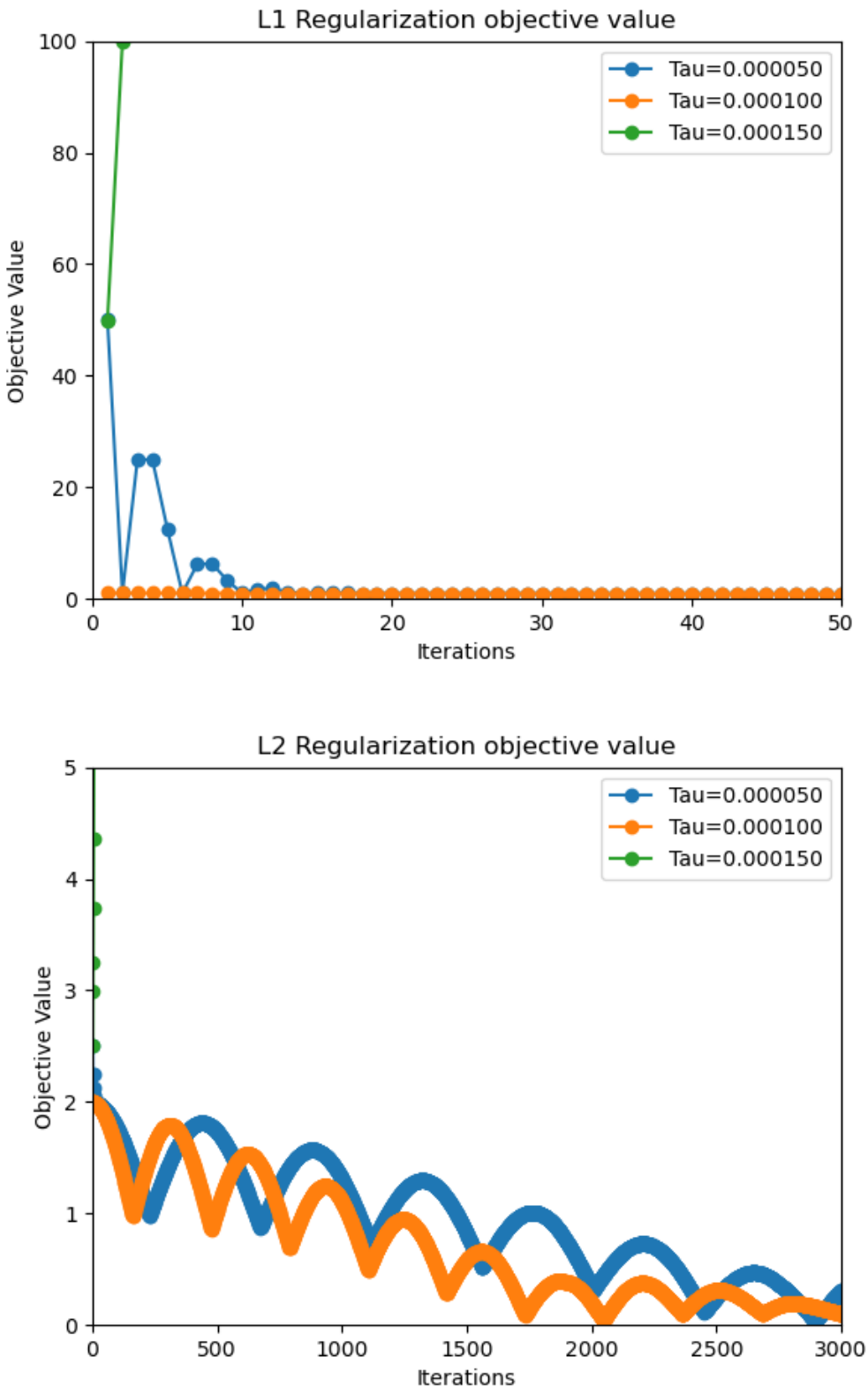


Figure 1: The evolution of $\|\omega^k\|_1$ and $\|\tilde{C}\omega^k + \tilde{d}\|_2$ as algorithm progresses for three different values of $\tau \in \left\{0.5\frac{1}{\lambda_{\max}}, \frac{1}{\lambda_{\max}}, 1.5\frac{1}{\lambda_{\max}}\right\}$.

Lemma 3.1. *(Stalling in Variable ω for Very Small τ) Consider a general feasible BPDN problem in the form of* (3.1) *such that* $0 < \lambda\left(\tilde{C}\right) < Q_C < \infty$, $\|\tilde{d}\|_\infty < D < \infty$ *and* $\epsilon > 0$ *is a reasonably small number. Then, the primal variable* ω *does not converge to the optimal solution such that for any iteration* k,

$$\|\omega^k - \omega^{k+1}\|_2 \to 0, \|\omega^k - \omega^*\|_2 > L^k$$

for some $L^k \in \mathcal{R}^+$ *when* $\tau \to 0$ *is selected. Note that* ω^* *denotes the optimal solution to the problem.*

Proof. In this Lemma, we have that the gradient g^{k-1} satisfies

$$\|g^{k-1}\|_\infty = \left\|\tilde{C}^T(\tilde{C}\omega^{k-1} + \tilde{d} - r^k - \frac{1}{\mu}y^{k-1})\right\|_\infty \leq Q_C \cdot \left(Q_C\|\omega^{k-1}\|_\infty + D + \epsilon + \frac{1}{\mu}\|y^{k-1}\|_\infty\right) < \infty$$

by Minkowski and Cauchy inequality. In addition, sequence of $\{\tau_i\}_i$ is independent with eigenvalues λ_i of matrix $\tilde{C}$. By updating rule of variable ω, we have $\omega^k = \max\left\{|\omega^{k-1} - \tau g^{k-1}| - \frac{\tau}{\mu}, 0\right\} \cdot \text{sign}(\omega^{k-1} - \tau g^{k-1})$. Then the sequence $\{\omega^k\}^k$ stops progressing at any iteration $k \in N$ when sending the sequence $\{\tau_i\}_i$ to zero. Apply Fatou Lemma:

$$\begin{aligned}
\lim_{i\nearrow\infty} \|\omega^{k-1} - \omega^k\|_2^2 &= \limsup_{i\nearrow\infty} \left\|\omega^{k-1} - \max\left\{|\omega^{k-1} - \tau g^{k-1}| - \frac{\tau}{\mu}, 0\right\} \cdot \text{sign}(\omega^{k-1} - \tau g^{k-1})\right\|_2^2 \\
&= \limsup_{i\nearrow\infty} \left\|\mathcal{T}\left\{\tau\left[g^{k-1} - \text{sign}(\omega^{k-1} - \tau g^{k-1}) \cdot \frac{1}{\mu}\right], \omega_{k-1}\right\}\right\|_2^2 \\
&\leq \left\|\limsup_{i\nearrow\infty} \mathcal{T}\left\{\tau\left[g^{k-1} - \text{sign}(\omega^{k-1} - \tau g^{k-1}) \cdot \frac{1}{\mu}\right], \omega_{k-1}\right\}\right\|_2^2 \\
&= \sum_{j=1}^{n}\left(\limsup_{i\nearrow\infty} \mathcal{T}\left\{\tau\left[g^{k-1} - \text{sign}(\omega^{k-1} - \tau g^{k-1}) \cdot \frac{1}{\mu}\right], \omega_{k-1}\right\}\right)^2 \to 0
\end{aligned} \tag{3.6}$$

where $\mathcal{T}$ denotes an operator for which the following component-wise property must hold

$$\begin{aligned}
\lim_{x\to 0} \mathcal{T}\{x, \omega^{k-1}\}_j &= \omega_j^{k-1} - \lim_{x\to 0}\left\{\text{Shrink}\left(\omega^{k-1} - x, x\right)_j\right\} \\
&= \omega_j^{k-1} - \limsup_{x\to 0} \max\left\{|\omega_j^{k-1} - x| - x, 0\right\} \cdot \text{sign}(\omega_j^{k-1} - x) \\
&= \begin{cases} \omega_j^{k-1} - \omega_j^{k-1} \to 0, & \text{if } \omega_j^{k-1} \geq 0 \\ \omega_j^{k-1} + |\omega_j^{k-1}| \to 0, & \text{if } \omega_j^{k-1} < 0 \end{cases}
\end{aligned}$$

By assumption 1 and 2, we have the following simple induction:

- At the 1^{st} iteration, the inequality $\|\omega^0 - \omega^*\|_2 \geq L > \epsilon_T$ holds. Furthermore, since the distance between two points is zero if and only if they are identical in metric (L2) space, we can argue that $\omega^1 = \omega^0$ by equation (3.6);
- For any index $k > 0$, assume that the inequality $\|\omega^k - \omega^*\|_2 \geq L > \epsilon_T$ holds. Then, we obtain

$\omega^k = \omega^{k-1}$ by the same reasoning. Thus, $\|\omega^{k+1} - \omega^*\|_2 \geq L > \epsilon_T$ will also hold next iteration which verifies the hypothesis.

Hence, we can conclude that the ADMM fails to converge to the optimal solution when τ is sufficiently small. □

The next Lemma follows closely the previous one: We demonstrate that just a single extremely large eigenvalue of the matrix $\tilde{C}$ may impact on the convergence to the optimal solution during the iteration process. Thus, we may obtain a conclusion that the greater gap between eigenvalues of matrix $\tilde{C}$, the harder problems are and many more iterations would be needed. The following assumption and definition will be applied:

Assumption 3. *The hyper-parameter τ strictly follows the Theorem (3.1) such that*

$$\tau\lambda_{\max}(\tilde{C}^T\tilde{C}) < 1$$

Definition 3.1. *(Lp-Norm of Matrix) If the p-norm ($1 \leq p < \infty$) for vectors is used for both spaces $\mathcal{R}^n$ and $\mathcal{R}^m$, then*

$$\|A\|_p := \sup_{x \neq \mathbf{0}} \frac{\|Ax\|_p}{\|x\|_p}$$

defines the corresponding operator norm and the following relations (in L2-Norm) hold:

1. $\|A\|_2 = \lambda_{\max}(A^TA) = \sigma_{\max}(A^TA)$

2. $\|Ax\|_2 \leq \sigma_{\max}(A^TA)\|x\|_2$

The proof is straightforward and can be found in Appendix B.

Lemma 3.2. *(Performance in Ill-Conditioned Diagonal Matrix) Denote set $B := \{1, 2, ..., n\}\backslash\{j\}$. Consider a sequence of diagonal matrices $\left(\tilde{C}_i\right)_{i\in N}$ such that $\tilde{C}_i^{(j,j)} = (\lambda_{\max})_i \to \infty$ is strictly increasing and $\lambda_b < \delta_\lambda < \infty, \forall b \in B$. Assume that the sub-vector $\omega_B^* \subset \omega^*$ is non-zero such that*

$$\|\omega_B^0 - \omega_B^*\|_2 \geq K > \epsilon_T.$$

Then, if $\|\omega_B^{k-1}\|_2 < \infty$ and $\|y_B^{k-1}\|_2 < \infty$ hold, then ω_β^k will stall and stop converging to the optimal as $i \to \infty$.

Proof. Without losing generality, we assume the maximal eigenvalue is the first entry of matrix $\tilde{C}_i$ because we can still use permutation matrix to change the position of the maximal eigenvalue. In this case, we assign $j = 1$ such that $B = \{1, 2, ..., n\}\backslash\{1\}$. For instance, to change the a^{th} and b^{th} entries

$(a < b)$ of matrix $\tilde{C}_i$ we have:

$$P_1\tilde{C}P_2 = \begin{bmatrix} e_1 \\ \vdots \\ e_b \\ \vdots \\ e_a \\ \vdots \end{bmatrix} \begin{bmatrix} \tilde{C}^{(1,1)} & & & & & \\ \vdots & \ddots & & & & \\ 0 & \cdots & \tilde{C}^{(a,a)} & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ 0 & \cdots & \cdots & \cdots & \tilde{C}^{(b,b)} & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} e_1 \\ \vdots \\ e_b \\ \vdots \\ e_a \\ \vdots \end{bmatrix}^T = \begin{bmatrix} \tilde{C}^{(1,1)} & & & & & \\ \vdots & \ddots & & & & \\ 0 & \cdots & \tilde{C}^{(b,b)} & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ 0 & \cdots & \cdots & \cdots & \tilde{C}^{(a,a)} & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

According to the Theorem 3.1, we have

$$\tau_i < \frac{1}{(\lambda_{max})_i}.$$

Because of the assumption that $\|\omega_B^0 - \omega_B^*\|_2 \geq K > \epsilon_T$, there exists at least one entry of ω_B^* is bounded away from zero:

$$\exists \beta \in B, s.t. \ \|\omega_\beta^0 - \omega_\beta^*\|_2 \geq K' > \epsilon_T. \tag{3.7}$$

Hence, there exists an integer $k \in N$ such that

$$\|\hat{\omega}_\beta^{k-1} - \hat{\omega}_\beta^*\|_2 \geq L_b^{k-1}$$

when the entry has not attained the optimal solution yet: $\exists L_b^{k-1} \in \mathcal{R}^+$ such that $L_b^{k-1} \geq \epsilon_T$. Trivially, we can take $k = 0$ such that $\|\omega_\beta^0 - \omega_\beta^*\|_2 \geq L_b^0 = K'$. Denote the set

$$\mathcal{A} := \left\{ \omega_b^{k-1} : |\omega_b^{k-1} - \tau_i(\tilde{C}^2) g_b^{k-1}| \geq \frac{\tau_i(\tilde{C}^2)}{\mu}, \forall b \in B \right\}.$$

Consider $\|\Delta\omega_B^k\|_2^2$ by equation (3.6) with the index $i \in N$ for sequence of $\{\tau_i\}_i$

$$\|\Delta\omega_B^k\|_2^2 := \left\|\omega_B^{k-1} - \omega_B^k\right\|_2^2 = \left\|\omega_B^{k-1} - \max\left\{|\omega_B^{k-1} - \tau_i(\tilde{C}^2) g_B^{k-1}| - \frac{\tau_i(\tilde{C}^2)}{\mu}, 0\right\} \cdot \text{sign}(\omega_B^{k-1} - \tau_i(\tilde{C}^2) g_B^{k-1})\right\|_2^2.$$

where we have the following sub-vectors and sub-matrix

$$\omega_B^{k-1} = \begin{bmatrix} \omega_2^{k-1} \\ \omega_3^{k-1} \\ \vdots \\ \omega_n^{k-1} \end{bmatrix} \in \mathcal{R}^{n-1}, \omega_B^k = \begin{bmatrix} \omega_2^k \\ \omega_3^k \\ \vdots \\ \omega_n^k \end{bmatrix} \in \mathcal{R}^{n-1}, r_B^{k-1} = \begin{bmatrix} r_2^{k-1} \\ r_3^{k-1} \\ \vdots \\ r_n^{k-1} \end{bmatrix} \in \mathcal{R}^{n-1}$$

and

$$\hat{C} = \begin{bmatrix} C^{(2,2)} & & \\ & \ddots & \\ & & C^{(n,n)} \end{bmatrix} \in \mathcal{R}^{(n-1)\times(n-1)}, \;\; y_B^{k-1} = \begin{bmatrix} y_2^{k-1} \\ y_3^{k-1} \\ \vdots \\ y_n^{k-1} \end{bmatrix} \in \mathcal{R}^{(n-1)}.$$

Note that only the first element of matrix $\tilde{C}$ is going to infinity which means the sub-matrix $\tilde{C}$ is strictly dominated by $\delta_\lambda \cdot I_{n-1}$:

$$\hat{C} = \begin{bmatrix} C^{(2,2)} & & \\ & \ddots & \\ & & C^{(n,n)} \end{bmatrix} \prec \delta_\lambda \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & & 1 \end{bmatrix}.$$

On the other hand, the multiplier $\tau_i(\tilde{C}^2) := \epsilon(\tau) \cdot \frac{1}{(\lambda_{max})_i} = \epsilon(\tau)\frac{1}{(\tilde{C}_i^{(1,1)})^2}$ is controlled by the extremely large value $(\lambda_{\max})_i$ where $\epsilon(\tau) \in (0,1)$. As the consequence, we can argue that $\|g_B^{k-1}\|_2^2$ is bounded given the primal and dual variable are both bounded in L2-norm:

$$\begin{aligned}
\|g_B^{k-1}\|_2^2 := & \left\| \begin{bmatrix} (\tilde{C}_i^{(2,2)})^2 & & & \\ & (\tilde{C}_i^{(2,2)})^2 & & \\ & & \ddots & \\ & & & (\tilde{C}_i^{(2,2)})^2 \end{bmatrix} \begin{bmatrix} \omega_2^{k-1} \\ \omega_3^{k-1} \\ \vdots \\ \omega_n^{k-1} \end{bmatrix} \right\|_2^2 \\
& + \left\| \begin{bmatrix} \tilde{C}_i^{(2,2)} & & & \\ & \tilde{C}_i^{(2,2)} & & \\ & & \ddots & \\ & & & \tilde{C}_i^{(2,2)} \end{bmatrix} \begin{bmatrix} d_2 - r_2^k - \frac{y_2^{k-1}}{\mu} \\ d_3 - r_3^k - \frac{y_3^{k-1}}{\mu} \\ \vdots \\ d_n - r_n^k - \frac{y_n^{k-1}}{\mu} \end{bmatrix} \right\|_2^2 \\
& \leq (\lambda_{\max}^{\hat{C}})_i^2 \|\omega_B\|_2^2 + (\lambda_{\max}^{\hat{C}})_i (\|d_B\|_2^2 + \|r_B\|_2^2 + \|y_B/\mu\|_2^2) \\
& \leq \delta_\lambda^2 \|\omega_B\|_2^2 + \delta_\lambda \left(D^2 + \epsilon^2 + \frac{\|y_B\|_2^2}{\mu^2} \right) < \infty
\end{aligned}$$

where $\|r\|^2 := \left\| \mathcal{P}_{\mathbf{B}_\epsilon}(\tilde{C}\omega + \tilde{d} - \frac{y}{\mu}) \right\|^2 < \epsilon^2$. For a single component, the above translates into:

- If $\omega_b^{k-1} \in \mathcal{A}$, then

$$\begin{aligned}
\|\omega_b^{k-1} - \omega_b^k\|_2^2 &\leq \left\| \tau_i g_B^{k-1} - \frac{\tau_i}{\mu} \right\|_2^2 \\
&= \tau_i^2 \left(\left\| g_B^{k-1} \right\|_2 + \left\| \frac{1}{\mu} \right\|_2 \right)^2 \\
&\leq \tau_i^2 \left(\left\| g_B^{k-1} \right\|_2^2 + \frac{2}{\mu} \left\| g_B^{k-1} \right\|_2 + \left\| \frac{1}{\mu} \right\|_2^2 \right)
\end{aligned} \tag{3.8}$$

- If $\omega_b^{k-1} \notin \mathcal{A}$, then

$$\hat{\omega}_b^k = 0$$

Next, after plugging in the upper bound on $\|g_B^{k-1}\|_2^2$ back to equation (3.8), we find for the entry with index β:

- Case 1: When the soft threshold takes value of $\hat{\omega}^{k-1} - \tau_i \hat{g}^{k-1} - \frac{\tau_i}{\mu} \cdot \text{sign}(\hat{\omega}^{k-1} - \tau_i \hat{g}^{k-1})$, the L2-distance in equation (3.8) is apparently controlled by τ_i. Hence, each step will be much smaller as the value of $(\lambda_{\max})_i$ increases and gradually converges to zero. This will eventually lead to $\|\omega_\beta^k - \omega_\beta^*\|_2 \to L_b^k$ as $i \to \infty$;
- Case 2: When the soft threshold takes value of zero, the distance between next iteration $\hat{\omega}_\beta^k$ and the optimal is still K' in condition (3.7). This is because $\omega_\beta^k = \omega_\beta^0$ and they share the same distance to the optimal solution.

For any further iteration, the process will be the same. This implies

$$\|\omega^\infty - \omega^*\|_2 \geq \|\hat{\omega}_\beta^\infty - \hat{\omega}_\beta^*\|_2 \geq \min\{K', L_b^k\} > \epsilon_T$$

as $i \to \infty$.

□

Remark 5. *In Lemma 3.2, the entry in primal variable ω corresponding to the large eigenvalue can still convergence fast while the others found very hard to progress.*

4 Interior Point Proximal Method of Multipliers

Because the first-order method has a terrible performance in ill-conditioned problems, it is necessary to implement the second-order method to handle in those cases. Aspired by Matrix Free Method for Compressed Sensing [8], we have reformulated the Basis Pursuit Denoising Problem into a convex quadratic programming problem according to Theorem (2.3). This method is set up based on that formulation. However, it has been constructed on [18] and [22] and thus very brief explanation will be provided. Note that **the notations in this section will follow closely in [18], [22] and Theorem 2.3**.

4.1 Notations in IPPMM

In this subsection, we will introduce key notations used throughout this section.

- Primal variable: $x \in \mathcal{R}^N$, Dual variable: $y \in \mathcal{R}^M$, Auxiliary variable: $z \in \mathcal{R}^N$;
- μ_k denotes the parameter in front of the barrier function;
- $c \in \mathcal{R}^N, Q \in \mathcal{R}^{N\times N}, A \in \mathcal{R}^{M\times N}, b = \mathbf{0}_M$ have been defined in Theorem 2.3;
- ξ_k represents for the proximal variable of x_k, λ_k represents for the proximal variable of y_k;
- $\mathcal{I} := \{M+1, M+2, ..., N\}$ denotes the index set for constrained variables such that $x^{\mathcal{I}} \geq 0$ in our problem;
- $\mathcal{F} := \{1, 2, ..., M\}$ denotes the index set for free variables such that $x^{\mathcal{F}} \in \mathcal{R}^{N-M}$ in our problem;
- X and Z represent diagonal matrices

$$X := \begin{bmatrix} x^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & x^N \end{bmatrix}, Z := \begin{bmatrix} z^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & z^N \end{bmatrix};$$

- $\mathcal{P}$ denotes the permutation matrix such that

$$\mathcal{P}x = [(x^{\mathcal{F}})^T, (x^{\mathcal{I}})^T]^T;$$

- $\bar{b}$ and $\bar{c}$ are defined in [18], Equation (18);
- $\kappa(\cdot)$ denotes the condition number of some matrix;
- Matrix Ξ_k is defined in [22]:

$$\Xi_k := \mathcal{P}^T \begin{bmatrix} 0_{|\mathcal{F}\times\mathcal{F}|} & 0_{|\mathcal{I}\times\mathcal{F}|} \\ 0_{|\mathcal{F}\times\mathcal{I}|} & (X_k^{\mathcal{I}})^{-1} Z_k^{\mathcal{I}} \end{bmatrix} \mathcal{P};$$

- $\lambda_{\max}(\cdot), \lambda_{\min}(\cdot)$ denote the maximal and minimal eigenvalue of some matrix;

4.2 Brief Explanation of IP-PMM

Initially, we start by Proximal Method of Multipliers [20] with the following updating rule:

- Primal variable:
$$x_{k+1} = \arg\min_x \left\{ L_{\mu_k}(x, y_k) + \frac{\mu_k}{2}\|x - \xi_k\|_2^2 \right\}$$
where $L_{\mu_k}(x, y_k)$ denotes the Augmented Lagrangian Function and $\xi_k := x_k$ is the primal variable in the k^{th} update;

- Dual variable:
$$y_{k+1} = \lambda_k - \frac{1}{\mu_k}(Ax_{k+1} - b)$$
is to promise the dual feasibility, where $\lambda_k := y_k$ is the dual variable in the k^{th} update.

Then, we can notice a sub-problem aroused in the updating rule of primal variable where the second-order method (Newton's Method) will be applied to solve. Next, start to derive the Newton system: write down the Augmented Lagrangian Function with proximal term added and logarithmic barriers included:

$$\mathcal{L}^{\mu_k}(x, \xi_k, \lambda_K) = c^T x + \frac{1}{2}x^T Q x - \lambda_k^T(Ax - b) + \frac{1}{2\mu_k}\|Ax - b\|_2^2 + \frac{\mu_k}{2}\|x_k - \xi_k\|_2^2 - \mu_k \sum_{j\in\mathcal{I}} \ln x_j$$

Next, the first-order (optimality) condition

$$\begin{bmatrix} c + Qx - A^T y - z + \mu_k(x - \xi_k) \\ Ax - b + \mu_k(y - \lambda_k) \\ X^{\mathcal{I}} z^{\mathcal{I}} - \mu_k e_{\mathcal{I}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

and Newton Equation in [22]

$$\begin{bmatrix} -(Q + \mu_k I) & A^T & I \\ A & \mu_k I & 0 \\ Z_k & 0 & X_k \end{bmatrix} \begin{bmatrix} \Delta x_k \\ \Delta y_k \\ \mathcal{P}^T \begin{bmatrix} 0_{\mathcal{F}} \\ \Delta z_k \end{bmatrix} \end{bmatrix} = \begin{bmatrix} -(c + \sigma_k \frac{\mu_k}{\mu_0}\bar{c}) - Qx_k + A^T y_k + z_k - \sigma_k \mu_k (x_k - \xi_k) \\ Ax_k - (b + \sigma_k \frac{\mu_k}{\mu_0}\bar{b}) + \sigma_k \mu_k (y_k - \lambda_k) \\ \mathcal{P}^T \begin{bmatrix} 0_{\mathcal{F}} \\ X_k Z_k e_n - \sigma_k \mu_k e_n \end{bmatrix} \end{bmatrix} \tag{4.1}$$

are derived. Because Newton Method works well only under quadratic convergence, a modified update rule (IPM) should be set up promising the convergence. In IPPMM, the updating rule of each iterative variable is in two major steps:

- Step 1: Update x, y, z such that $(x_{k+1}, y_{k+1}, z_{k+1}) = (x_k + \alpha_k \Delta x_k, y_{k+1} + \alpha_k \Delta y_k, z_{k+1} + \alpha_k \Delta z_k)$. We obtain $(\Delta x_k, \Delta y_k, \Delta z_k)$ by solving the Newton system above. Actually, we will not solve the Newton system (4.1) precisely because the Hessian matrix on LHS might be not invertible and it is also computationally inefficient. Instead, we may choose to solve that inexactly, for example by conjugate gradient method or other iterative methods for solving system $Ax = b$. However, this is out of the scope of this thesis but readers could find out more in [14] and [2].

  Step size parameter $\alpha_k \in (0, 1]$ such that

$$\alpha_k = \sup\left\{ \alpha \in (0, 1] : \mu_k(\alpha) \le (1 - 0.01\alpha)\mu_k, (x_{k+1}, y_{k+1}, z_{k+1}) \in \mathcal{N}_{\mu_k(\alpha)}(\xi_k, \lambda_k) \right\},$$

  where $\mu_k(\alpha) := \frac{(x_k + \alpha_k \Delta x_k)^T (z_k + \alpha_k \Delta z_k)}{N}, \mu_k = \frac{x_k^T z_k}{N}$ and N represents the dimension in Theorem 2.3. Note that

$$\mathcal{N}_{\mu_k}(\xi_k, \lambda_k) := \left\{ (x, y, z) \in \mathcal{R}^N \times \mathcal{R}_+^M \times \mathcal{R}_+^N : (x, y, z) \in \tilde{\mathcal{C}}_{\mu_k}(\xi_k, \lambda_k), x^i z^i \ge \gamma_\mu \mu_k, \forall i \in \mathcal{I} \right\}$$

and

$$\tilde{\mathcal{C}}_{\mu_k}(\xi_k,\lambda_k) := \left\{(x,y,z) : \left\|\left(\tilde{b}_k^{\lambda_k,\mu_k}(x,y), \tilde{c}_k^{\xi_k,\mu_k}(x,y,z)\right)\right\|_2 \le C_N, \left\|\left(\tilde{b}_k^{\lambda_k,\mu_k}(x,y), \tilde{c}_k^{\xi_k,\mu_k}(x,y,z)\right)\right\|_{\mathcal{A}} \le \gamma_{\mathcal{A}}\rho\right\},$$

where $C_N, \gamma_\mu, \gamma_{\mathcal{A}}$ and ρ are constants and we have the scaled infeasibility at iteration k:

$$\tilde{b}_k^{\lambda_k,\mu_k}(x,y) = \frac{\mu_0}{\mu_k}\left(Ax - b - \frac{\mu_k}{\mu_0}\bar{b} + \mu_k(y-\lambda_k)\right)$$
$$\tilde{c}_k^{\xi_k,\mu_k}(x,y,z) = \frac{\mu_0}{\mu_k}\left(-Qx + A^T y + z - \mu_k(x-\xi_k) - c - \frac{\mu_k}{\mu_0}\bar{c}\right)$$

and the semi-norm $\|\cdot\|_{\mathcal{A}}$ is defined as:

$$\|(b,c)\|_{\mathcal{A}} = \min_{(x,y,z)} \{\|(x,z)\|_2 : Ax = b, -Qx + A^T y + z = c\}.$$

We will then update μ by $\mu_{k+1} := \frac{(x_{k+1})^T(z_{k+1})}{N}$. Thus, α_k is the parameter who is aiming for controlling variable $(x_{k+1}, y_{k+1}, z_{k+1})$ in next iteration will always stay in the neighbourhood which will also be maximally contracted by α_k, and make the sequence $\{\mu_k\}_k$ a strict decreasing sequence as well. Because $\mu_{k+1} := \mu_k(\alpha) \le (1 - 0.01\alpha)\mu_k < \mu_k$ which will tighten the set $\tilde{\mathcal{C}}_\mu$ every update.

- Step 2: Progress in Proximal Method of Multipliers and obtain next sub-problem when the KKT is almost satisfied. Hence, in KKT we have the primal residual: $r_p = Ax_{k+1} - (b + \frac{\mu_{k+1}}{\mu_0}\bar{b})$ and dual Residual: $r_d = (c + \frac{\mu_{k+1}}{\mu_0}\bar{c}) + Qx_{k+1} - A^T y_{k+1} - z_{k+1}$ such that the variable $(\xi_0, \lambda_0) = (x_0, y_0)$ as the starting point, and

$$(\xi_{k+1}, \lambda_{k+1}) = \begin{cases} (x_{k+1}, y_{k+1}) & \text{if } \|(r_p, r_d)\|_2 \le \frac{\mu_{k+1}}{\mu_0} C_N \text{ and } \|(r_p, r_d)\|_{\mathcal{A}} \le \gamma_{\mathcal{A}}\rho \\ (\xi_k, \lambda_k) & \text{Otherwise} \end{cases}$$

will be updated by (x, y) if $\|(r_p, r_d)\|_2$ is decreased significantly and much closer to zero than before; It will be equal to itself and continue solving the same sub-problem otherwise.

The stopping criteria for IPPMM is straightforward that it promises primal and dual feasibility under the tolerance. Additionally, μ_k should also be reduced within the tolerance:

$$\|Ax - b\| < tol, \|c + Qx - A^T y - z\| < tol, \mu_k < tol.$$

Furthermore, the convergence theorems for IPPMM method have been established in Theorem 1 to 3 of [18]. This method holds a computational complexity of $\mathcal{O}(N^4)$ for convergence.

4.3 Convergence Analysis in Ill-Posed Problem

In order to well prove further why IPPMM performs better than ADMM in ill-conditioned problems, we will firstly show why the first-order method is much more poor than the second in general framework. The following Theorem compare the performance of these two methods in hard problems:

Theorem 4.1. *(Comparison between First- and Second-Order Method in Ill-Conditioned L1-Regularized Problems) Consider a LASSO formulation of L1-Regularized problem in the form of (2.5) and apply*

the pseudo-Huber function introduced in [8] such that the problem is becoming:

$$\min_{\omega\in\mathcal{R}^n}\left\{\frac{1}{2}\|\tilde{C}_i\omega+\tilde{d}\|_2^2+\lambda\psi_\mu(\omega)\right\},$$

where assume that both two methods start with the same point and $\tilde{C}_i$ is a sequence of positive diagonal matrix with the first entry $(\lambda_1)_i\to\infty$ such that the condition number

$$\kappa_i(\tilde{C}^T\tilde{C})\to\infty,$$

where $\lambda_j:=\lambda_j(\tilde{C}^T\tilde{C})$ denotes the j^{th} eigenvalue of matrix $\tilde{C}^T\tilde{C}$ and let $\sigma_j:=\sigma_j(\tilde{C}^T\tilde{C})=\sqrt{\lambda_j(\tilde{C}^T\tilde{C})}$ in this theorem. Then, the first-order method gradually stop stepping while the impact on second-order method will be minor.

Proof. Firstly, consider a typical first-order method updating rule:

$$\Delta\omega_j=\alpha\left(\lambda\nabla\psi_\mu(\omega)+\tilde{C}_i^T(\tilde{C}_i\omega+\tilde{d})\right)_j$$

Without losing the generality [similarly as Lemma (3.2)], we take the maximal eigenvalue of matrix $\tilde{C}$ on the first entry and the minimal on the last, i.e. $\lambda_1>\lambda_2>\cdots>\lambda_n$. Now, consider the choice of step size $\alpha\in\mathcal{R}^+$:

- When α is not small enough such that $\alpha\geq\epsilon_\alpha^1\frac{1}{(\lambda_1)_i}$ is bounded above and $0<\epsilon_\alpha^1<\infty$ is a finite number bounded away from zero, then the first entry of variable will be divergent as i goes:

$$\lim_{i\nearrow\infty}\alpha\left(\lambda\partial_{\omega_1}\psi_\mu(\omega)+(\lambda_1)_i\omega+(\sigma_1)_id_1\right)\to\infty$$

- However, if we control the first entry convergent by taking a small $\alpha<\epsilon_\alpha^2\frac{1}{(\lambda_1)_i}$ with $\epsilon_\alpha^2\in(0,1)$, then the last entry of variable will step significantly slowly and stop stepping as $i\to\infty$:

$$\lim_{i\nearrow\infty}\alpha\left(\lambda\partial_{\omega_1}\psi_\mu(\omega)+(\lambda_1)_i\omega+(\sigma_1)_id_1\right)\leq\lim_{i\nearrow\infty}\left(\epsilon_\alpha\frac{1}{(\lambda_1)_i}\lambda\partial_{\omega_1}\psi_\mu(\omega)+\epsilon_\alpha^2\frac{1}{\kappa_i}\omega+\epsilon_\alpha^2\frac{\sigma_n}{(\lambda_1)_i}d_1\right)\to 0$$

On the other hand. Consider the Newton–Raphson (second-order) method updating rule where we can, of course, take the inverse matrix due to a diagonal matrix itself:

$$\Delta\omega_j=(\tilde{C}_i^T\tilde{C}_i)_j^{-1}\left(\lambda\nabla\psi_\mu(\omega)+\tilde{C}_i^T(\tilde{C}_i\omega+\tilde{d})\right)_j=\frac{1}{(\lambda_j)_i}\lambda\nabla\psi_\mu(\omega)+\omega_j+\frac{1}{(\lambda_j)_i}\tilde{d}_j$$

Thus, the variable will not stop stepping although the problem is going more ill-conditioned:

- For the first entry, we have:

$$\Delta\omega_1 = \lim_{i \nearrow \infty} \frac{1}{(\lambda_1)_i}\lambda\nabla\psi_\mu(\omega) + \omega_1 + \frac{1}{(\sigma_1)_i}\tilde{d}_1 \to \omega_1$$

  However, note that the optimal ω_i^* will decrease to zero as $(\lambda_1)_i$ goes to infinity. This will therefore take only one step to optimal solution ω_1^* when λ_1 is sufficiently large.

- For the others, the updating will not be dominated by the large eigenvalue because it does not depend on the maximal eigenvalue λ_1 here.

□

By the same reasoning, we then start the analysis of updating rule for three variables in IPPMM. Before that, the following Lemma prove that the newly updated proximal variable (ξ_k, λ_k) will not go to infinity under a certain condition which is determined by PMM.

Lemma 4.1. *(Updating Behavior for Proximal Variables λ_k and ξ_k) Given that the latest updated variables (x_k, y_k) and (ξ_r, λ_r) is bounded in infinity norm, then $\|(r_k^\xi, r_k^\lambda)\|_\infty$ is also bounded at k^{th} iteration where $r_k^\xi := \xi_k - x_k$ and $r_k^\lambda := \lambda_k - y_k$ represent the proximal residuals. Note that r means the r times having updated proximal variable (ξ, λ).*

Proof. Denote $r_k \in N$ as the index where (ξ, λ) is latest updated such that

$$(x_{r_k}, y_{r_k}) \to (\xi_{r_k}, \lambda_{r_k}),$$

which is bounded in infinity norm $\|(\xi_{r_k}, \lambda_{r_k})\|_\infty < B_{r_k} < \infty$. Now, consider the k^{th} updating for variables (ξ_k, λ_k) where $k > r_k$. If primal and dual variable (x_k, y_k) is also bounded in infinity norm $\|(x_k, y_k)\|_\infty < B_k^{(x,y)} < \infty$, then there will be two cases for updating variables (ξ_k, λ_k):

- Case 1: the primal and dual residual in L2-norm is not close enough to zero such that the proximal variable will not be updated and stay in the same sub-problem where $(\xi_{k_r}, \lambda_{k_r}) \to (\xi_k, \lambda_k)$ shares the same bound in infinity norm. In this case,

$$\|(r_k^\xi, r_k^\lambda)\|_\infty \leq \|(\xi_{k_r}, \lambda_{k_r})\|_\infty + \|(x_k, y_k)\|_\infty < B_{k_r} + B_k^{(x,y)} < \infty.$$

- Case 2: the primal and dual residual in L2-norm is very close to zero implying that it attained at the optimal solution to sub-problems. In this case, we assign $(x_k, y_k) \to (\xi_k, \lambda_k)$ such that

$$\|(r_k^\xi, r_k^\lambda)\|_\infty = 0.$$

Thus, we can conclude that residuals (r_k^ξ, r_k^λ) are always bounded as long as the condition above is satisfied. □

Assumption 4. *The problem has at least bounded one feasible solution*

$$(x^*, y^*, z^*) < \infty$$

and the starting point is given as same as [18] where

$$(x_0, z_0) = \rho(e_N, e_N), \;\; y_0 = e_m$$

for some finite constant $\rho > 0$.

Assumption 5. *The vector* b *is a zero vector*

$$b = \mathbf{0}$$

because the formulation in Theorem 2.3 has exactly the same result.

Assumption 6. *The bounded matrix* Q *is diagonal and positive semi-definite*

$$0 \preceq Q := \begin{bmatrix} Q_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & Q_N \end{bmatrix} \preceq Q_{\max} \cdot I_N < \infty$$

and vector c *is bounded in infinity norm*

$$\|c\|_\infty \leq D.$$

Assumption 7. *At this point, we may assume that the matrix* Ξ_k *is well-conditioned*

$$\Xi_k^{\mathcal{I}} e_{\mathcal{I}} = X_k^{-1} Z_k \preceq c_k^{\Xi} I_{\mathcal{I}},$$

where $c_k^{\Xi} > 0$ *is a finite real number. Because* $\forall i \in \mathcal{I}, x_i z_i \geq \gamma_\mu \mu_k > 0$, *the matrix* $\Xi_k^{\mathcal{I}}$ *is positive definite. Thus, it is not necessary to take a norm when considering upper bound.*

Remark 6. *In the iterative process, matrix* $\Xi_k^{\mathcal{I}}$ *is becoming much more ill-conditioned as the variables approach to the optimal ones: In L1-regularized problem, when* $x_k \to x^*$ *and* $z_k \to z^*$ *we have*

- *Zero variables:* $x_k^j \to (x^*)^j = 0$ *such that* $\left| \frac{z_j^k}{x_j^k} \right| \to \infty$;
- *Non-zero variables:* $x_k^j \to (x^*)^j \neq 0$ *but* $x_k^j z_k^j \to \mu_k$ *which implies* $\left| \frac{z_j^k}{x_j^k} \right| \to 0$;
- *Therefore, by two bullet points above, we can conclude that*

$$\lambda_{\max}(\Xi_k^{\mathcal{I}}) \to \infty, \lambda_{\min}(\Xi_k^{\mathcal{I}}) \to 0,$$

which leads to $\kappa(\Xi_k^{\mathcal{I}}) \to \infty$.

Nevertheless, we can apply the Dropping Variables technique introduced in [22] to reduce the ill condition. Thus, we logically assume the matrix is quite well conditioned.

Because only matrix $\tilde{C}$ is generally ill-conditioned which will generate a ill-conditioned matrix A, then these assumptions will hold in the following Theorem:

Theorem 4.2. *(Newton System of IPPMM in Ill-Conditioned Quadratic Programming) Assume that there exists an entry in diagonal matrix A is sufficiently large. Then, the movement in two variables* Δx_k *and* Δy_k *will not be effected by the ill condition if the condition in Lemma 4.1 holds.*

Proof. Without losing the generality, take the large eigenvalue in the first entry of matrix A. This implies that the matrix $A^T A$ will be in the following form:

$$A^T A = \begin{bmatrix} \lambda_{\max}^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_N^2 \end{bmatrix} \succ 0.$$

Then, we consider the augmented system (2.8) in paper [22] at k^{th} iteration:

$$\begin{bmatrix} -(Q + \Xi_k + \mu_k I_N) & A^T \\ A & \mu_k I_m \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} r_k^1 \\ r_k^2 \end{bmatrix} := \begin{bmatrix} c + Qx_k - A^T y_k + \sigma_k \mu_k (x_k - \xi_k) - z_k \\ b - Ax_k - \sigma_k \mu_k (y_k - \lambda_k) \end{bmatrix}.$$

Next, write down the normal equation for Δx and Δy respectively:

- The Equation w.r.t. Δx_k:

$$\begin{aligned} & -\left((Q + \Xi_k + \mu_k I_N) + \frac{1}{\mu_k} A^T A \right) \Delta x \\ & = r_k^1 - A^T r_k^2 \frac{1}{\mu_k} \\ & = r_1^k - A^T \left(b - Ax_k - \sigma_k \mu_k (y_k - \lambda_k) \right) \cdot \frac{1}{\mu} \\ & = \frac{1}{\mu_k} A^T A x_k + A^T \left((\sigma_k - \frac{1}{\mu_k}) y_k - \sigma_k \lambda_k \right) + \sigma_k \mu_k (x_k - \xi_k) + (c + Qx_k). \end{aligned} \tag{4.2}$$

  Note that the matrix $\Theta_k := (Q + \Xi_k + \mu_k I_N)$ is a diagonal matrix such that

$$\mu_k \cdot I_N \preceq \Theta_k = \begin{bmatrix} Q_{\mathcal{F}} + \mu_k I_{|\mathcal{F} \times \mathcal{F}|} & 0_{|\mathcal{I} \times \mathcal{F}|} \\ 0_{|\mathcal{F} \times \mathcal{I}|} & \mu_k I_{|\mathcal{I} \times \mathcal{I}|} + (X_k^{\mathcal{I}})^{-1} Z_k^{\mathcal{I}} + Q_{\mathcal{I}} \end{bmatrix} \preceq \left(\mu_k + \max\left\{ c_k^{\Xi} + Q_{\mathcal{F}}, Q_{\mathcal{I}} \right\} \right) \cdot I_N$$

  Denote $y_k - \lambda_k := r_k^{\lambda}$ and $x_k - \xi_k := r_k^{\xi}$ such that

$$\|\sigma_k (y_k - \lambda_k)\|_\infty = \sigma_k \|r_k^{\lambda}\|_\infty < \infty, \quad \|\sigma_k \mu_k (x_k - \xi_k)\|_\infty = \sigma_k \mu_k \|r_k^{\xi}\|_\infty < \infty$$

by Lemma 4.1 and the two terms regarding primal and dual variables

$$\|c + Qx_k\|_\infty \leq D + Q_{\max}\|x_k\|_\infty < \infty, \ \left\|\frac{y_k}{\mu_k}\right\|_\infty < \infty.$$

Then, multiply the inversion of matrix $\left((Q + \Xi_k + \mu_k I_N) + \frac{1}{\mu_k}A^T A\right)^{-1}$ on both sides:

$$\Delta x = \left(\Theta_k + \frac{1}{\mu_k}A^T A\right)^{-1}\left[\frac{1}{\mu_k}A^T A x_k + A^T\left((\sigma_k - \frac{1}{\mu_k})y_k - \sigma_k\lambda_k\right) + \sigma_k\mu_k(x_k - \xi_k) + (c + Qx_k)\right],$$

where the matrix $\left(\Theta_k + \frac{1}{\mu_k}A^T A\right)^{-1}$ is bounded blow. Component-wisely, for the first entry of Δx which corresponds to the large eigenvalue, we have:

$$\begin{aligned}
&\Delta x_k^1 \\
&= \lim_{\lambda_{\max}\to\infty} -\left(\Theta_k + \frac{1}{\mu_k}A^T A\right)_1^{-1}\left[\frac{1}{\mu_k}A^T A x_k + A^T\left((\sigma_k - \frac{1}{\mu_k})y_k - \sigma_k\lambda_k\right) + \sigma_k\mu_k(x_k - \xi_k) + (c + Qx_k)\right]_1 \\
&= \lim_{\lambda_{\max}\to\infty}\left(-\frac{1}{\mu_k}\lambda_{\max}^2\right)^{-1}\left[\frac{1}{\mu_k}\lambda_{\max}^2 x_k^1 + \lambda_{\max}\left(\sigma_k r_k^\lambda - \frac{1}{\mu_k}y_k\right)_1 + \sigma_k\mu_k(r_k^\xi)_1 + (c + Qx_k)_1\right] \to -x_k^1.
\end{aligned}$$

For the other entries, the case will be same as Theorem (4.1) that will not be impacted by the large eigenvalue.

- The Equation w.r.t. Δy_k has been given in [22] equation (2.10):

$$\begin{aligned}
\left(A\Theta_k^{-1}A^T + \mu_k I_N\right)\Delta y_k &= r_k^2 + A\Theta_k^{-1}r_k^1 \\
&= -A\Theta_k^{-1}A^T y_k + A\Theta_k^{-1}\left[(c + Qx) + \sigma_k\mu_k r_k^\xi - z_k - \Theta_k x_k\right] - \sigma_k\mu_k r_k^\lambda.
\end{aligned} \tag{4.3}$$

Take the inversion of $A\Theta_k^{-1}A^T + \mu_k I_N$ and multiply on both sides:

$$\Delta y_k = \left(A\Theta_k^{-1}A^T + \mu_k I_N\right)^{-1}\left[r_k^2 + A\Theta_k^{-1}r_k^1\right].$$

Similarly, when considering the first entry of variable Δy_k, we have:

$$\lim_{\lambda_{\max}\to\infty}(A\Theta_k^{-1}A^T + \mu_k I_N)_1 \to (A\Theta_k^{-1}A^T)_1 = (\lambda_{\max})^2,$$

then

$$\begin{aligned}
\Delta y_k^1 &= \lim_{\lambda_{\max}\to\infty}(A\Theta_k^{-1}A^T)_1^{-1}\left(-A\Theta_k^{-1}A^T y_k + A\Theta_k^{-1}\left[(c + Qx) + \sigma_k\mu_k r_k^\xi - z_k - \Theta_k x_k\right] + -\sigma_k\mu_k r_k^\lambda\right)_1 \\
&= \lim_{\lambda_{\max}\to\infty}(\lambda_{\max})^{-2}\left(-(\lambda_{\max})^2 y_k^1 + \lambda'_{\max}\left[(c + Qx)_1 + \sigma_k\mu_k(r_k^\xi)_1 - z_k^1 - \Theta_k^1 x_k^1\right] - \sigma_k\mu_k(r_k^\lambda)_1\right) \\
&\to -y_k^1.
\end{aligned}$$

Thus, for an ill-conditioned diagonal matrix A in IPPMM, the newton system will behave similarly as Theorem (4.1).

□

The Theorem (4.1) is also applicable for the formulation in Theorem (2.3). By the problem (2.6), we have the following block matrix

$$A^T A = \begin{bmatrix} I & -\tilde{C} & \tilde{C} \\ -\tilde{C} & \tilde{C}^2 & -\tilde{C}^2 \\ \tilde{C} & -\tilde{C}^2 & \tilde{C}^2 \end{bmatrix}$$

Assume matrix $\tilde{C}$ has a sufficiently large entry (the first entry, for example), then again the large entry will only affect the first variable ω_1^+, ω_1^- and u_1 in primal variable x because of the scalar product.

5 Numerical Experiment

This chapter compares the two optimization algorithms proposed before, the Alternating Direction Method of Multipliers (ADMM) and the Interior Point Penalty and Modified Multiplier (IPPMM) method. The special focus here is on dealing with challenges commonly faced in real-world applications, like problems with dense, ill-conditioned, and large-scale data.

When working with problems that involve L1-Regularized Least Squares Temporal Difference (LSTD), situations might arise where dense matrices come into play. These matrices have a lot of non-zero elements and this depends on the chosen method for approximating functions. Ill-conditioned matrices add another layer of difficulty due to their sensitivity to even tiny changes, causing numerical instability. Large-scale matrices make things even more complex, leading to increased computational demands. ADMM and IPPMM, which are known for their effectiveness, are compared to see how well they handle these challenges.

The earlier theoretical discussions in Chapters 3 and 4 are complemented by a focus on practical usability. This chapter breaks down how these algorithms deal with the specific characteristics of each type of matrix, giving practical insights into their performance.

5.1 Implementation Details

In this section, we will introduce how we implemented the numerical experiments. All the experiments in Section 5.2 to 5.5 were performed on Alienware m15 R6 running Windows 11 with a 11^{th} Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz processors with 16GB memory by employing MATLAB 2023A. To best control the scale, condition and sparsity of each problem, we take use of generators proposed in paper [7]. Specifically, Instance Generator 2 (IGen2) will be applied throughout all experiments in this chapter. Pseudo-code for the implementation can be found below Algorithm 1. Next, we will introduce the theory behind IGen2 and all of the code for Algorithm 1 is available in this GitHub link https://github.com/Fusion2010/My-Incredible-Dissertation, originated from https://www.maths.ed.ac.uk/ERGO/trillion/. Note that we will use m and n represent the number of rows and columns in matrix $\tilde{A}$ in problem (5.1) throughout this chapter and in order to avoid abuse of notations, we replace A and b in [7] with $\tilde{A}$ and $\tilde{b}$ respectively.

Remark 7. *Following will explain the terminology in Algorithm 1:*

- ***McPA*** *is the matrix generator introduced in page 615 of [7];*
- ***OsGen1*** *and* ***OsGen3*** *refer to the solution generator introduced in section 5.1.2;*
- $rand(a, b, c)$ *means that it generates a random vector with length of* c *within the interval* $[a, b]$*;*
- ***QPconvertor*** *is to convert ADMM formulation into IPPMM in Formulation (2.6).*

5.1.1 Introduction to IGen2

This introduction will follow closely with Section 3.2 in paper [7]. Overall, the general idea behind IGen2 is that customize matrix A and optimal solution x^* such that users can control the attributes of them, and then the generator will match the vector b to make x^* is optimal to the problem. Now, assume that the l1-regularized problem is constructed in the form of

$$\min_{x \in \mathcal{R}^n} \left\{ \frac{1}{2} \|\tilde{A}x - \tilde{b}\|^2 + \epsilon \|x\|_1 \right\} \tag{5.1}$$

Algorithm 1 Modified IGen2 Instances Generator for ADMM and IPPMM

Require: $P > 0$, $\epsilon > 0$, $\theta > 0$, $\gamma > 0$, $I_x \in \{0, 1\}$ ▷ P is the power involved in generating n, ϵ is the same nation in Problem (5.1), θ is rotation angle in generating matrix $\tilde{A}$ in equation (5.3), γ is defined in section 5.1.2, and I_x is the indicator for using OsGen1 or OsGen3 in section 5.1.2

Ensure: $n > m$ and $m \geq k$

$n \leftarrow 2^P$

$m \leftarrow \frac{1}{2}n$

$k \leftarrow m$ ▷ Number of nonzero entries in x^*

$\Sigma_B \leftarrow rand(\lambda^B_{\max}, \lambda^B_{\min}, m)$

$\Sigma_N \leftarrow rand(\lambda^N_{\max}, \lambda^N_{\min}, m)$

$B \leftarrow \mathbf{MvPA}(\Sigma_B, \theta)$ ▷ Generate matrix B

$N \leftarrow \mathbf{MvPA}(\Sigma_N, \theta)$ ▷ Generate matrix N

if $I_x = 1$ **then**

  $x^* \leftarrow \mathbf{OsGen3}(\gamma, k)$ ▷ Generate the optimal solution x^*

else

  $x^* \leftarrow \mathbf{OsGen1}(\gamma, k)$ ▷ Generate the optimal solution x^*

end if

for $i \in \{1, 2, ..., m\}$ **do**

  if $x^*_i > 0$ **then**

    $g_i \leftarrow 1$

  else if $x^*_i < 0$ **then**

    $g_i \leftarrow -1$

  else

    $g_i \leftarrow rand(-1, 1)$ ▷ Randomly choose a value for g_i within $[-1, 1]$

  end if

end for

$e \leftarrow \tau B^{-1} g$

for $i \in \{1, 2, ..., m\}$ **do**

  $\tilde{N}_i \leftarrow \frac{\tau N_i}{\|N^T e\|_\infty} \cdot rand(-1, 1)$ ▷ We replace $|N_i^T e|$ with $\|N^T e\|_\infty$ to promise matrix $\tilde{A}$ well-conditioned

end for

$\tilde{A} \leftarrow [B|\tilde{N}]$, $\tilde{b} \leftarrow Ax^* + e$ and x^*

$\tilde{C} \leftarrow \tilde{A}$ and $\tilde{d} \leftarrow -\tilde{b}$ ▷ Convert into ADMM formulation in 2.4

$[A, Q, b, c] \leftarrow \mathbf{QPconvertor}(m, n, \tilde{C}, \tilde{d}, \epsilon)$ ▷ Convert into IPPMM formulation in 2.6

return x^*, $\tilde{C}$, $\tilde{d}$, Q, A, c and b

with $\tilde{A} \in \mathcal{R}^{m\times n}$ and $\epsilon > 0$, where we have $m < n$ such that the matrix A has more number of columns than rows. Then, we consider to split matrix $\tilde{A}$ into $B \in \mathcal{R}^{m\times m}$ and $N \in \mathcal{R}^{m\times(n-m)}$. If we obtain an optimal solution x^* and the matrix B and N (will be discussed in detail later), then vector $\tilde{b} \in \mathcal{R}^m$ can be built up by first-order condition of the problem stated above. Note that the optimal solution x^* must satisfy the following conditions:

- Without losing the generality, denote the set $I_B := \{1, 2, ..., m\}$ and $I_N := \{m+1, m+2, ..., n\}$ as the index set for columns in matrix B and N respectively. For each $i \in I_N$, we have $x_i^* = 0$;
- Furthermore, there exists a subset $I_S \subseteq I_B$ such that $x_s^* \neq 0, \forall s \in I_S$.

Then, we consider the first-order condition:

$$0 \in \tilde{A}^T(\tilde{A}x^* - \tilde{b}) + \epsilon\partial\|x^*\|_1 \Rightarrow \tilde{A}^T(\tilde{A}x^* - \tilde{b}) \in -\epsilon\partial\|x^*\|_1,$$

where ∂ denotes the sub-derivative. Hence, if we take $e := \tau B^{-T}\partial\|x^*\|_1$, $\tilde{N}_i := \tau\frac{\xi}{|N_i^T e|}N_i$ where ξ is a uniform random variable in $[-1, 1]$ and assign $\tilde{b} = \tilde{A}x^* + e$, then

$$\begin{aligned} B^T(Bx_B^* - Ax^* - e) &= B^T(Bx_B^* - Bx_B^* - e) \\ &= -B^T e \\ &= -\epsilon\partial\|x_B^*\|_1 \end{aligned}$$

and

$$\begin{aligned} \tilde{N}^T(\tilde{N}x_N^* - b) &= -\tilde{N}^T e \\ &= -\epsilon\frac{\xi}{|N_i^T e|}N_i e_i \\ &\in -\epsilon[-1, 1]^{n-m} := -\epsilon\partial\|x_N^*\|_1 \end{aligned}$$

will hold, where x_B and x_N represent the collections of entries in x corresponding to index sets I_B and I_N respectively. As a consequence, it satisfies the optimality condition for each entry of variable x^*, meaning that the vector b we obtained made the x^* optimal. After obtaining A, b and x^* by IGen2, we assign:

$$\tilde{A} \to \tilde{C}, -\tilde{b} \to \tilde{d} \text{ and } x^* \to \omega^* \tag{5.2}$$

to be consistent with our problem formulated in equation (2.4) or (2.5). One could also refer to page 611 in [7] for Pseudo-code.

5.1.2 Introduction to x^* Generators: OsGen & OsGen3

There are two technical issues have not been addressed yet: generate matrix A and optimal solution as inputs to IGen2. Thus, we will discuss two generators for x^* in this sub-section. Get started on OsGen which just simply use the random numbers in uniform distribution, but we made a tiny modification in our cases. Rather than randomly selecting s numbers of all indices where x^* will take non-zero value in those indices, we assign the first s entries of x^* to be non-zero and uniformly choose value within the interval $[-\gamma, \gamma], \gamma > 0$ for them. This is for the purpose of being consistent with IGen2 in terms of programming. OsGen has an apparent advantage: the selection process for x^* is totally independent with matrix $\tilde{A}$. In this case, x^* can be chosen as a very well-conditioned solution no matter what condition matrix $\tilde{A}$ is.

On the other hand, value of the optimal solution(s) x^* will actually depend on the eigenvalues of matrix $\tilde{A}$ in practice. For example, if there exists an entry $\tilde{A}_{(i,j)}$ of $\tilde{A}$ is very large, then the corresponding x_j might be forced to very small value for a minimization problem. The OsGen3 Generator

follows this idea:

$$x^* := \arg\min_{x \in \mathcal{R}^n} \|G^T x - \gamma(\Sigma^T \Sigma)^{-1} \mathbf{1}_n\|^2, \text{ Subject to: } \|x\|_0 \leq s,$$

where $\mathbf{1}_n$ denotes the unit vector with dimension n, G denotes the right singular vectors, Σ denotes singular values of $\tilde{A}^T \tilde{A}$ and $\|\cdot\|_0$ is the L0-norm referring to the number of non-zero entries. In such case, x^* is approximately equal to Gv where $v = \gamma(\Sigma^T \Sigma^T)^{-1} 1_n$:

$$G^T x - \gamma(\Sigma^T \Sigma)^{-1} \mathbf{1}_n := G^T G \gamma(\Sigma^T \Sigma^T)^{-1} 1_n - \gamma(\Sigma^T \Sigma)^{-1} \mathbf{1}_n = 0.$$

After sorting non-zero entries $x_i^*, i \in 1, 2, ..., s$, we obtain the optimal solutions by assigning rest of components equal to zero. For more details, one could refer to Section 6 in [7].

5.1.3 Generate of Non-Trivial Matrix $\tilde{A}$

In the IGen2, matrix $\tilde{A}$ is assigned by the block matrix $[B|N]$. However, we provide only the singular values of matrix B and N as inputs to IGen2. The theoretical analysis for handling challenging situations involving ill-conditioned problems has been constructed by picking $\tilde{C}$ (which is actually $\tilde{A}$ here but different notation: please see updating rule 5.2) being a diagonal matrix. To generalize our results, we have to consider the cases that are more complex and practical (where $\tilde{A}$ is not simply a diagonal matrix). In the method outlined in Section 4 of [7], it used singular value decomposition (SVD) along with rotation matrices to craft matrix $\tilde{A}$. Now, we state the following theorem for SVD:

Theorem 5.1. *(Singular Value Decomposition) Assume $A \in \mathcal{R}^{m \times n}$ with unitary matrices $V \in \mathcal{R}^{n \times n}$, $U \in \mathcal{R}^{m \times m}$ and corresponding singular matrix $\Sigma \in \mathcal{R}^{m \times n}$. Then*

$$U^T A V = \Sigma$$

which is Singular Value Decomposition (SVD) and such decomposition exists.

Proof can be found in Appendix B, Theorem B.2.

By multiplying U and V^T on both sides, we can express an unknown matrix $\bar{A}$ by orthogonal matrices U and V with singular value matrix Σ:

$$(UU^T)\bar{A}(VV^T) = \bar{A} = U\Sigma V^T.$$

Take a Givens rotation matrix:

$$G(i, j, \theta) := \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & \cos(\theta) & \cdots & -\sin(\theta) & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & \sin(\theta) & \cdots & \cos(\theta) & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}. \tag{5.3}$$

This matrix possesses orthogonality properties that can be easily observed: On the diagonal of the matrix, the elements follow the pattern $\cos^2(\theta) + \sin^2(\theta) = 1$, which reflects the well-known trigonometric identity. For the off-diagonal elements at positions i and j, the entries $\cos(\theta)\sin(\theta) - \cos(\theta)\sin(\theta) = 0$,

resulting in a cancellation of terms. Understanding this concept, we can utilize a random permutation matrix P to achieve the following:

$$(P\tilde{G}P)\Sigma G^T = \tilde{A},$$

where $\tilde{G} \in \mathcal{R}^{m\times m}, G \in \mathcal{R}^{n\times n}$ follows the same definition in page 614 of [7]. In practice, we apply this decomposition to both the matrices B and $\tilde{N}$. The matrix P introduces a random shuffling of the rows and columns, effectively reordering the elements within the matrices. By using this method, we make the most of random permutation and rotation matrices to purposefully change and improve the features of matrices in a practical and organized way.

5.2 Warm Up: A very Low-Dimension, Sparse and Well-Conditioned Case

In this small-scale example, we are able to display the complete matrices, vectors, and optimal solutions. Thus, only the hyper-parameter settings will be listed below:

- We set $m = 4$ and $n = 8$ for dimension of matrix $\tilde{C}$;
- In ADMM (this will follow notations in Section 3), we take $\tau = 0.95/\lambda_{\max}(\tilde{C}^T\tilde{C})$, starting points $(r, \omega, y) = (0_m, 0_n, 0_m)$, $\mu = 1$, $\epsilon = 0.5$, $\epsilon_T = 1e-6$;
- In IPPMM (this will follow notations in Section 4), we take $tol = 1e-6$ and $\lambda = 0.5$.

Here is the specific case we've solved in this test:

$$\min_{\omega\in\mathcal{R}^n} \left\{ \|\omega\|_{\mathbf{L1}} : \|\tilde{d} + \tilde{C}\omega\|_{\mathbf{L2}} \le \epsilon \right\},\ \epsilon \in \mathcal{R}^+$$

Matrix $\tilde{C}$:

$$\tilde{C} = \begin{bmatrix} 5.116 & 2.397 & 0.0 & 0.0 & 0.806 & -0.200 & -0.068 & 0.069 \\ -2.397 & 1.483 & 0.0 & 0.0 & 0.262 & 0.614 & -0.022 & -0.211 \\ 0.0 & 0.0 & 2.922 & 2.844 & 0.642 & -0.059 & 0.515 & -0.193 \\ 0.0 & 0.0 & -2.844 & 4.906 & 0.208 & 0.182 & 0.167 & 0.593 \end{bmatrix}$$

Vector $\tilde{d}$:

$$\tilde{d} = \begin{bmatrix} -3.0138 \\ -3.9623 \\ -3.1756 \\ -0.8297 \end{bmatrix}$$

The optimal solution and objective value:

$$\omega^* = \begin{bmatrix} -0.330340 \\ 1.947874 \\ 0.551980 \\ 0.488718 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, f^* := \frac{1}{2}\|\tilde{C}\omega^* + \tilde{d}\|^2 + \epsilon\|\omega^*\|_1 = 1.714666.$$

Below are plots depicting the performance of both the ADMM and IPPMM algorithms:

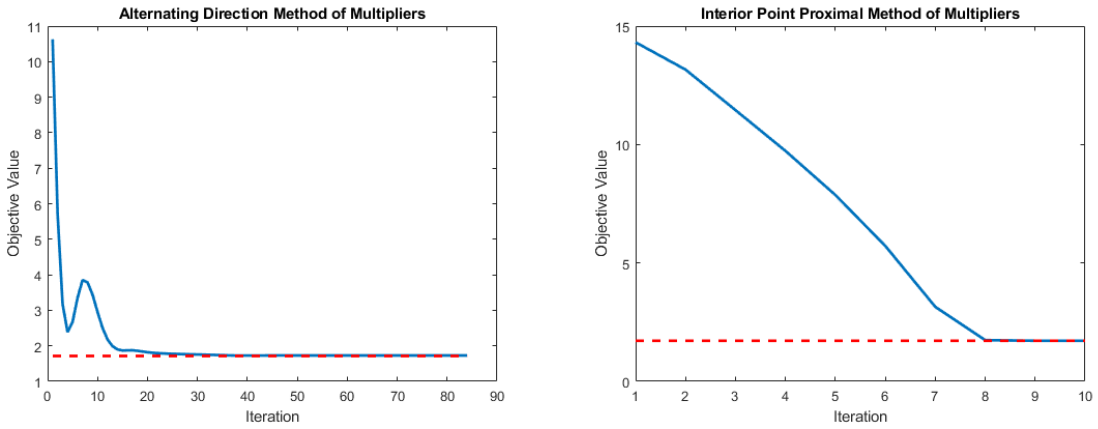


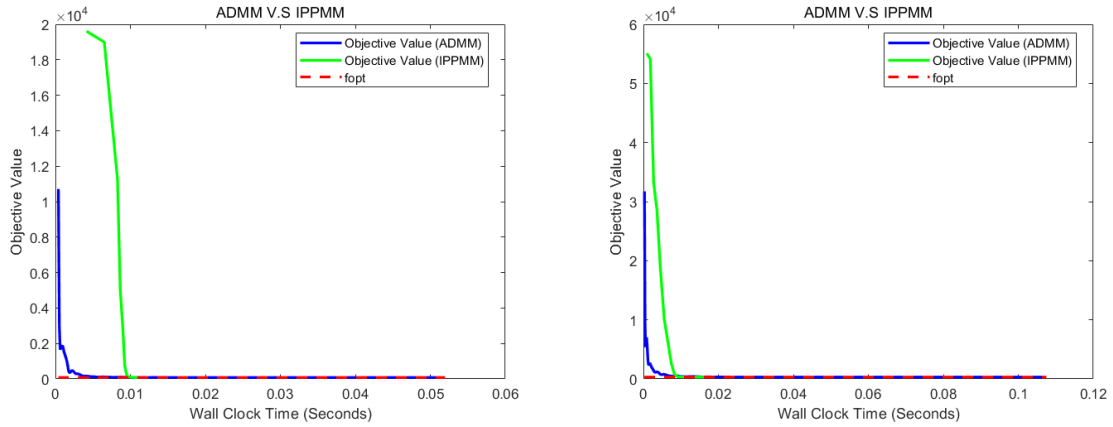
Figure 2: Performance of ADMM and IPPMM in a Simple Case

Both plots present a perfect convergence of the algorithms.

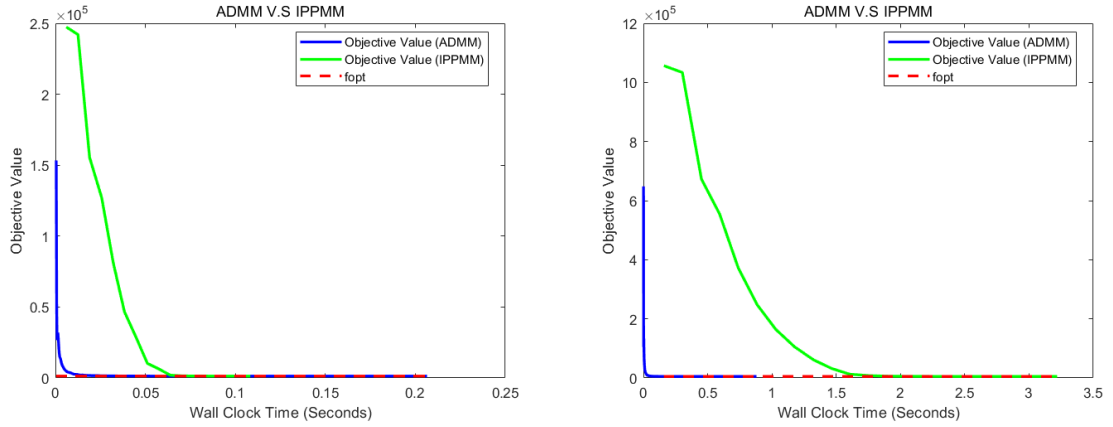
5.3 Performance in Large-Scale Problems

In this sub-section, we will demonstrate the performance of ADMM and IPPMM on problem (2.4) and (2.5) as the dimension of the primal variable, denoted as n gradually grows. Notations and parameter settings can be found below:

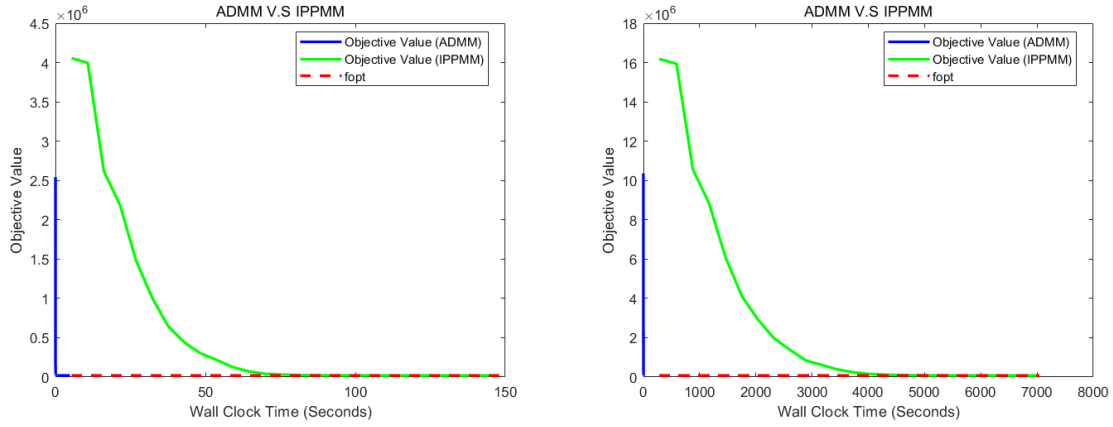
- Generator: IGen2 and OsGen1;
- Denote n the size of primal variable ω and $m = n/2$ the size of dual and auxiliary variable y, r;
- Denote $k = m$ the size of non-zero primal variable;
- $\epsilon = \lambda = 0.5$ are the parameters in front of the L1-norm in problems;
- $fvalOpt$ is the optimal objective value $\frac{1}{2}\|\tilde{C}\omega + \tilde{d}\|^2 + \epsilon\|\omega\|_1$. In our case: $\epsilon = \lambda$;
- $\tau = 0.5/\lambda(\max)(\tilde{C}^T\tilde{C})$ is the step parameter in ADMM;
- stopping criteria are both equal to 1×10^{-6};
- $\mu = 1$ is the parameter in ADMM;
- Matrix $\tilde{C}$ is constructed very well conditioned for controlling variant purpose such that $\lambda(\max)(\tilde{C}) = 2^2, \lambda_{\min}(\tilde{C}) = 1$;
- $\theta = \pi/6$ denotes the rotation angle in IGen2;
- The matrix $\tilde{C}$ is constructed as a relatively sparse matrix using the formula $\tilde{C} = (P\tilde{G}P)\Sigma G^T$;
- The optimal value x^* takes values within the interval $[-1, 1]$ by OsGen1.



(a) (b)



(c) (d)



(e) (f)

Figure 3: Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the dimension of variable n. (a) $n = 2^6$. (b) $n = 2^8$. (c) $n = 2^{10}$. (d) $n = 2^{12}$. (e) $n = 2^{14}$. (f) $n = 2^{16}$.

	Dimension		ADMM		IPPMM		
Index	n	m	CPU	Iter	CUP	Iter	$fvalOpt$
1	2^6	2^5	0.0520	513	0.0120	14	80.155008
2	2^8	2^7	0.1075	882	0.0187	19	302.063218
3	2^{10}	2^9	**0.2065**	1069	**0.1083**	17	1302.042474
4	2^{12}	2^{11}	0.8798	1706	3.2187	22	5119.691571
5	2^{14}	2^{13}	4.8190	2737	147.9718	25	20451.655826
6	2^{16}	2^{15}	**23.1480**	3357	$\mathbf{7.0401 \times 10^3}$	25	82389.046904

Table 2: Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the dimension of variable n.

In Figure 3 and Table 2, we demonstrate that IPPMM outperforms ADMM when the dimension of the matrix $\tilde{C}$ is relatively small under well-conditioned cases. However, as we increase the value of n to 2^{10}, ADMM gains an advantage. This is because IPPMM expands the dimension by a factor of five in its formulation (i.e., $\tilde{C} \in \mathcal{R}^{m\times n}$ in ADMM, but $Q \in \mathcal{R}^{(2n+m)\times(2n+m)} = \mathcal{R}^{5m\times 5m}$ in IPPMM, considering the condition where $n = 2m$). As an illustration, in the final test, we choose $n = 2^{16}$; however, the matrix Q exhibits dimensions of $(2^{17} + 2^{15}) \times (2^{17} + 2^{15})$ which is considerably large. Notably, in this experiment, ADMM spent only about 23 seconds to converge and terminate the iterative process, whereas IPPMM took an incredibly long time to attain the optimal solution (nearly 2 hours). In summary, when dealing with a problem that is terribly large in scale but relatively well-conditioned, ADMM might be a more suitable choice compared to IPPMM.

5.4 Performance in Ill-Conditioned Problems

In this sub-section, we will demonstrate the performance of ADMM and IPPMM for increasing condition number of matrix $\tilde{C}^T\tilde{C}$ or x^*.

5.4.1 Experiment 1: Increasing Condition Number of $\tilde{C}^T\tilde{C}$ while Keeping Condition of x^* Fixed

In the first experiment, we use IGen2 and OsGen1 generator and will only increase the condition number of $\tilde{C}^T\tilde{C}$. Notations and parameter settings can be found below:

- Generator: IGen2 and OsGen1;
- Fix $n = 2^{12}$ the size of primal variable ω and $m = n/2$ the size of dual and auxiliary variable y, r;
- Denote $k = m$ the size of non-zero primal variable;
- $\epsilon = \lambda = 0.5$ are the parameters in front of the L1-norm in problems;
- $fvalOpt$ is the optimal objective value $\frac{1}{2}\|\tilde{C}\omega + \tilde{d}\|^2 + \epsilon\|\omega\|_1$. In our case: $\epsilon = \lambda$;
- $\tau = 0.5/\lambda(\max)(\tilde{C}^T\tilde{C})$ is the step parameter in ADMM;
- stopping criteria are both equal to 1×10^{-6};
- $\mu = 1$ is the parameter in ADMM;
- $\kappa(\tilde{C}^T\tilde{C})$ denotes the condition number of matrix $\tilde{C}^T\tilde{C}$ is gradually growing up in the first experiment;
- $\theta = \pi/10$ denotes the rotation angle in IGen2;

- The matrix $\tilde{C}$ is constructed as a relatively sparse matrix using the formula $\tilde{C} = (P\tilde{G}P)\Sigma G^T$;

- The optimal value x^* takes values within the interval $[-100, 100]$ by OsGen1.

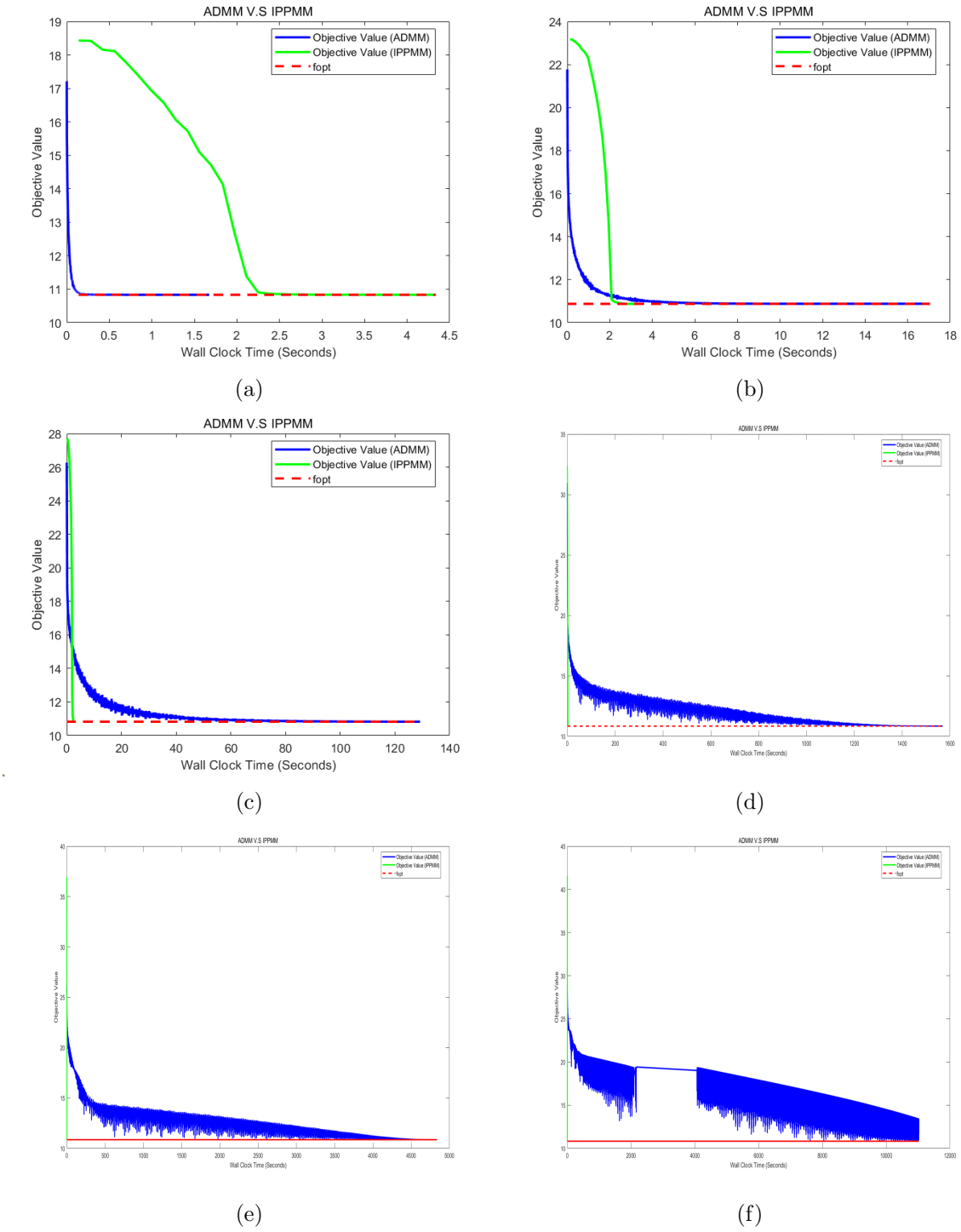


Figure 4: Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the condition number of matrix $\tilde{C}^T\tilde{C}$ while maintaining fixed condition of x^* using OsGen1. The y-axis is in log-scale. (a) $\kappa(\tilde{C}^T\tilde{C}) = 10^2$. (b) $\kappa(\tilde{C}^T\tilde{C}) = 10^4$. (c) $\kappa(\tilde{C}^T\tilde{C}) = 10^6$. (d) $\kappa(\tilde{C}^T\tilde{C}) = 10^8$. (e) $\kappa(\tilde{C}^T\tilde{C}) = 10^{10}$. (f) $\kappa(\tilde{C}^T\tilde{C}) = 10^{12}$.

	Condition		ADMM		IPPMM		
Index	$\kappa(\tilde{C}^T\tilde{C})$	γ	CPU	Iter	CUP	Iter	$fvalOpt$
1	10^2	100	1.6739	3493	4.3375	31	50727.091640
2	10^4	100	**17.0514**	3.4282×10^4	**3.3555**	24	53000.986304
3	10^6	100	129.1832	2.5122×10^5	3.0238	21	50003.0382
4	10^8	100	1.5694×10^3	2.5500×10^6	3.2724	20	51006.169379
5	10^{10}	100	$\mathbf{4.8377 \times 10^3}$	8.3710×10^6	**3.4080**	19	51756.294135
6	10^{12}	100	$\mathbf{1.1026 \times 10^4}$	1.3028×10^7	**4.0012**	17	50431.607427

Table 3: Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the condition number of matrix $\tilde{C}^T\tilde{C}$ while maintaining fixed condition of x^* using OsGen1.

In section 5.1.3, we introduced matrix $\tilde{A}$, which is generated through the operation $(PGP)\Sigma G^T$, with Σ representing the matrix containing singular values, and G being a given rotation matrix. Subsequently, we set $\tilde{C}$ to be equal to $\tilde{A}$. To regulate the condition $\kappa(\tilde{C}^T\tilde{C})$, we vary the value of p within the range $[0, 10^p]$, where p takes on values from the set $1, 2, ..., 6$. This choice causes each value in matrix Σ to be selected uniformly randomly from this interval. Consequently, as we increase p, the condition number $\kappa(\tilde{C}^T\tilde{C})$ grows from 10^2 to 10^{12} with each step. Additionally, we remain the condition of optimal solution x^* in each experiment the same, where each entry of x^* is uniform random variable in $[-\gamma, \gamma], \gamma = 100$ by applying OsGen1 generator. Finally, we select an appropriate dimension of $\tilde{C}$, i.e. $n = 2^{12}$, in order to ensure that the inclusion of a wide variety of values, spanning from small to large numbers within the range of $[0, 10^q]$. This choice is made to effectively establish the condition κ.

Overall, ADMM demonstrates higher speed when addressing cases with $\kappa(\tilde{C}^T\tilde{C}) = 10^2$, while the second-order method IPPMM exhibits notable advantages when dealing with situations where $\kappa(\tilde{C}^T\tilde{C}) \geq 10^4$. It's worth highlighting that ADMM's performance significantly deteriorates in the last two scenarios. In these cases, the algorithm takes approximately 1.33 and 3 hours to reach termination in the 5^{th} and 6^{th} cases, respectively. Moreover, in the case denoted as (f) in Figure 4, it becomes evident that ADMM struggles to converge effectively to the optimal solution; instead, it exhibits considerable fluctuations eventually. The reason for this behavior has been discussed in Lemma 3.2: In the last test, the step parameter in ADMM is forced to be incredibly small: $a \times 10^{-13}, a \in (0, 10)$. This is a crucial factor to the observed outcome. Hence, as the step parameter gradually decays, the attainment of the optimal solution x^* through ADMM becomes significantly more challenging. This will be exemplified in the next experiment.

5.4.2 Experiment 2: Increasing Condition Number of $\tilde{C}^T\tilde{C}$ with Non-Trivial x^* Constructed

In the second experiment, we will also gradually increase the condition number of $\tilde{C}^T\tilde{C}$. However, OsGen3 will be applied for constructing the non-trivial solution (see section 5.1.2). Notations and parameter settings can be found below:

- Generator: IGen2 and OsGen3;
- Fix $n = 2^6$ the size of primal variable ω and $m = n/2$ the size of dual and auxiliary variable y, r;
- Denote $k = m$ the size of non-zero primal variable;
- $\epsilon = \lambda = 0.5$ are the parameters in front of the L1-norm in problems;
- $\Delta \log f^*$ represents the absolute logarithmic difference in the optimal objective value $\frac{1}{2}\|\tilde{C}\omega + \tilde{d}\|^2 + \epsilon\|\omega\|_1$ attained by ADMM and IPPMM;

- $\tau = 0.5/\lambda(\max)(\tilde{C}^T\tilde{C})$ is the step parameter in ADMM;
- stopping criteria are both equal to 1×10^{-9};
- $\mu = 1$ is the parameter in ADMM;
- $\kappa(\tilde{C}^T\tilde{C})$ denotes the condition number of matrix $\tilde{C}^T\tilde{C}$ is gradually growing up in the experiment;
- $\theta = \pi/10$ denotes the rotation angle in IGen2;
- The matrix $\tilde{C}$ is constructed as a relatively sparse matrix using the formula $\tilde{C} = (P\tilde{G}P)\Sigma G^T$;
- The optimal value x^* generated by OsGen3.

	Dimension		ADMM		IPPMM		
Index	$\kappa(\tilde{C}^T\tilde{C})$	γ	CPU	Iter	CUP	Iter	$\Delta \log f^*$
1	10^0	100	46.6551	420381	0.0444	123	0.2973
2	10^2	100	6.3562	58382	0.0083	74	0.2911
3	10^4	100	0.7372	7025	0.0093	24	0.2448
4	10^6	100	0.4676	4590	0.0066	18	1.0968
5	10^8	100	0.5069	4862	0.0062	18	**4.2181**
6	10^{10}	100	1.2492	11536	0.0076	17	**10.8989**

Table 4: Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the condition number of matrix $\tilde{C}^T\tilde{C}$ with non-trivial construction of x^* by using OsGen3.

A notable rise in the absolute logarithmic difference of the objective value is evident in Table 4. This outcome conveys that ADMM exhibits a notably inferior performance. The underlying rationale for this phenomenon is theoretically substantiated by Lemma 3.2. The ensuing analysis of this experiment unfolds as follows:

- It's important to acknowledge that as the condition number $\kappa(\tilde{C}^T\tilde{C})$ escalates, the value of x^* goes towards zero due to the influence of OsGen3 generator: $x^* = Gv = G\gamma(\Sigma^T\Sigma^T)^{-1}\mathbf{1}_n$ (One can also find this by observing the optimal objective values in Figure 5 considerably decrease);
- In the context of the ADMM algorithm, the assignment of vector of ones for the starting point is noteworthy as it implies that, from the second group onward, the gap between them progressively widens;
- Finally, as the step parameter diminishes significantly, a noticeable plateauing effect becomes apparent. This results in a condition where the algorithm experiences difficulty in making substantial progress toward converging to the optimal solution. This has been presented by the *leveling off* in Figure 5.

Besides, note that in Figure 5, we denote **fopt** the optimal value obtained in IPPMM instead of IGen2 and OsGen3 generator.

5.5 Performance in Dense Problems

This sub-section naturally follows on from sub-section 5.1.3, where we've already established the method for generating matrix $\tilde{A}$. In order to control over the sparsity of $\tilde{A}$, we can achieve this by incorporating the count of Givens rotation matrix multiplications. This approach has been developed in section 4.2 of [7] and also is exemplified through two instances, showcasing that augmenting the count of Givens rotation matrices involved in the multiplication yields a denser matrix $\tilde{A}$. The ensuing

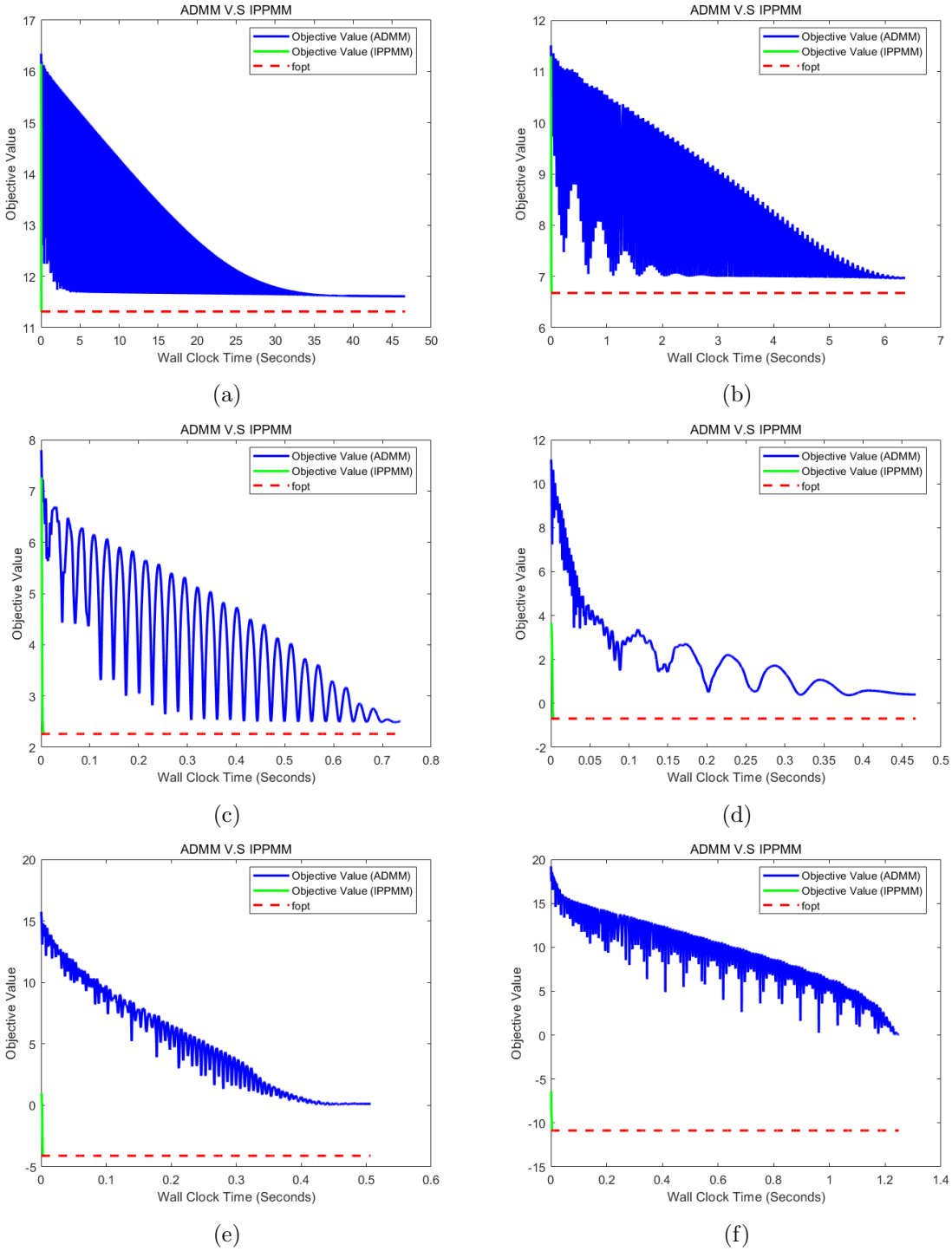


Figure 5: Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing the condition number of matrix $\tilde{C}^T\tilde{C}$ with non-trivial construction of x^* by using OsGen3. The y-axis is in log-scale. (a) $\kappa(\tilde{C}^T\tilde{C}) = 10^0$. (b) $\kappa(\tilde{C}^T\tilde{C}) = 10^2$. (c) $\kappa(\tilde{C}^T\tilde{C}) = 10^4$. (d) $\kappa(\tilde{C}^T\tilde{C}) = 10^6$. (e) $\kappa(\tilde{C}^T\tilde{C}) = 10^8$. (f) $\kappa(\tilde{C}^T\tilde{C}) = 10^{10}$.

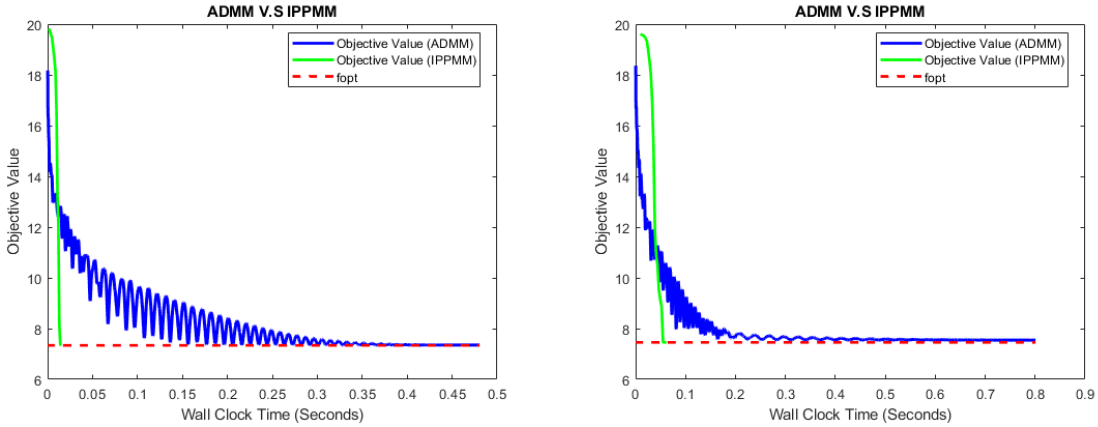
table will demonstrate how the number of non-zero entries evolves with a rise in the engagement of rotation matrices:

Sparsity Control of $\tilde{A}$ by IGen2		
Numbers of Givens Rotation Matrices	Dimension of Matrix $\tilde{A}$	Density of Matrix $\tilde{A}$
$A = (P\tilde{G}P)\Sigma G^T$	128×256	$[1016, 1024]$
$A = (P\tilde{G}\tilde{G}_2P)\Sigma(G_2G)^T$	128×256	$[3852, 3926]$
$A = (P\tilde{G}\tilde{G}_2\tilde{G}P)\Sigma(GG_2G)^T$	128×256	$[8056, 8228]$
$A = (P\tilde{G}_2\tilde{G}\tilde{G}_2\tilde{G}P)\Sigma(G_2GG_2G)^T$	128×256	$[12912, 13314]$
$A = (P\tilde{G}_2\tilde{G}\tilde{G}_2\tilde{G}\tilde{G}_2P)\Sigma(G_2GG_2GG_2)^T$	128×256	$[17900, 18476]$
$A = (P\tilde{G}_2\tilde{G}\tilde{G}_2\tilde{G}\tilde{G}_2\tilde{G}P)\Sigma(G_2GG_2GG_2G)^T$	128×256	$[21776, 23218]$

Table 5: Sparsity control analysis of matrix $\tilde{A}$ as the number of Givens rotation matrices increases. The dimensions of matrix $\tilde{A}$ remain constant at 128×256, while the density of $\tilde{A}$ is measured. The density values (numbers of non-zero entries) are provided as intervals, denoting the range between the lowest and highest values observed over 20 trials for each configuration of rotation matrices. Note that the rotation matrix $\tilde{G}, G, \tilde{G}_2$ and G_2 follows the definition of subsection 5.1.3.

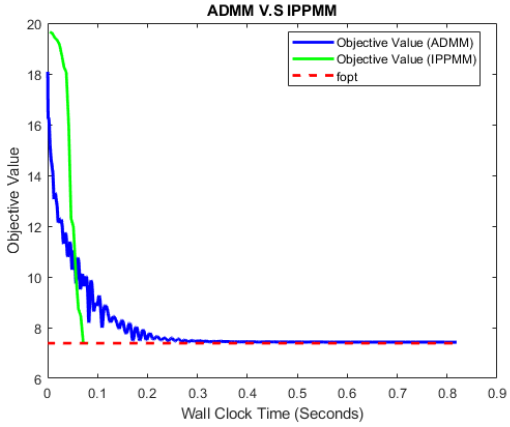
In the forthcoming experiment, we will systematically augment the density of matrices produced by the IGen2 with the technique introduced above. Subsequently, we will evaluate the performance of both ADMM and IPPMM algorithms in Figure 6 and table 6 when confronted with the progressively *densified* matrices in problem-solving scenarios. Notations and parameter settings can be found below:

- Generator: IGen2 and OsGen1;
- Fix $n = 2^8$ the size of primal variable ω and $m = n/2$ the size of dual and auxiliary variable y, r;
- Denote $k = m$ the size of non-zero primal variable;
- $\epsilon = \lambda = 0.5$ are the parameters in front of the L1-norm in problems;
- $\log f^*$ represents the logarithmic optimal objective value $\frac{1}{2}\|\tilde{C}\omega + \tilde{d}\|^2 + \epsilon\|\omega\|_1$;
- $\tau = 0.5/\lambda(\max)(\tilde{C}^T\tilde{C})$ is the step parameter in ADMM;
- stopping criteria are both equal to 1×10^{-3};
- $\mu = 1$ is the parameter in ADMM;
- $\kappa(\tilde{C}^T\tilde{C}) = 10^4$ denotes the condition number of matrix $\tilde{C}^T\tilde{C}$;
- $\theta = \pi/10$ denotes the rotation angle in IGen2;
- The sparsity of matrix $\tilde{C}$ is measured by $\rho(\tilde{C}) := \frac{\#\text{ Non-zero Entries}}{\#\text{ Total Entries}} \in [0, 1]$;
- The optimal value x^* generated within the interval $[-100, 100]$ by OsGen1.

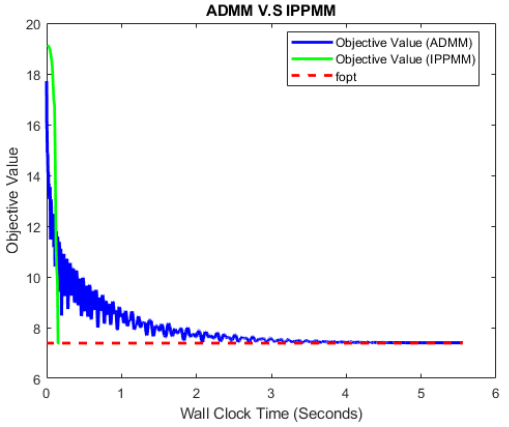


(a)

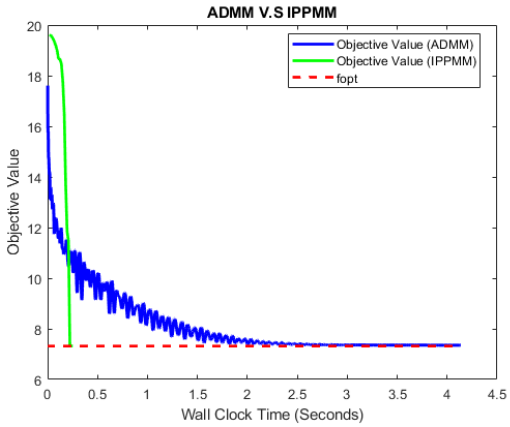
(b)



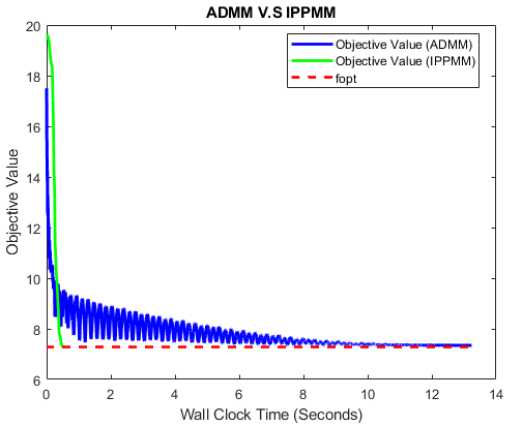
(c)



(d)



(e)



(f)

Figure 6: Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing non-zero entries of matrix $\tilde{C}^T\tilde{C}$ with x^* generated by OsGen1. The y-axis is in log-scale. (a) $\rho(\tilde{C}) = 3.12\%$. (b) $\rho(\tilde{C}) = 11.88\%$. (c) $\rho(\tilde{C}) = 25.31\%$. (d) $\rho(\tilde{C}) = 40.67\%$. (e) $\rho(\tilde{C}) = 56.20\%$. (f) $\rho(\tilde{C}) = 68.70\%$.

	Density		ADMM		IPPMM		
Index	$\rho(\tilde{C})$	#zero entries	CPU	Iter	CUP	Iter	$\log f^*$
1	3.12%	1024	0.4805	3597	0.0151	13	7.3397
2	11.88%	3892	0.8008	3536	0.0618	18	7.4569
3	25.31%	8292	0.8189	2254	0.0814	17	7.3849
4	40.67%	13328	5.5553	11142	0.1716	17	7.3875
5	56.20%	18416	4.1360	6313	0.2518	17	7.3197
6	68.70%	22512	13.2388	16208	0.5677	32	7.2850

Table 6: Performance of ADMM and IPPMM on Problem (2.4) and (2.5) for increasing non-zero entries of matrix $\tilde{C}^T\tilde{C}$ with x^* generated by OsGen1.

As depicted in Figure 6 and outlined in Table 6, a noticeable pattern emerges regarding the termination times of different algorithms based on the sparsity of the matrix. Specifically, it becomes evident that as the parameter $\rho(\tilde{C})$ increases, the time required for termination by each algorithm also increases.

Intriguingly, the ADMM algorithm demonstrates a substantial escalation in iteration time as $\rho(\tilde{C})$ rises. This suggests a heightened sensitivity of the ADMM algorithm to variations in matrix sparsity. On the other hand, the IPPMM algorithm appears to be considerably less affected by changes in sparsity. Even when the number of non-zero entries in the matrix surpasses half, the CPU time for the IPPMM algorithm remains consistently below 1 second.

6 Conclusions and Future Work

This thesis compares, analyzes, and illustrates two numerical approaches for the L1-Regularized Least-Square Temporal-Difference Learning algorithm, which is also a typical L1-regularized inverse problem. Specifically, we have established theoretical analyses for the performance of ADMM and IPPMM in ill-conditioned problems. In Section 3, we first established that the algorithm will cease to make progress as the step parameter τ in ADMM approaches infinity. This implies that ADMM fails to converge when the condition of matrix $\tilde{C}$ is significantly large, possibly due to the existence of extremely large entries. To address this issue, we introduce a second-order method known as IPPMM in Section 4. Prior to formally presenting IPPMM's effectiveness in handling ill-conditioning caused by a diagonal matrix, we provide a comprehensive comparison between first- and second-order methods for solving an ill-conditioned LASSO problem. However, since the complexity of these two algorithms has already been discussed, this thesis will not focus on this aspect. Instead, we have demonstrated the two algorithms above through numerical instances in various challenging scenarios.

Particularly, the theoretical results obtained in sections 3 and 4 have been verified in numerical experiments with non-trivial instances (no longer by diagonal matrices) constructed, where ADMM exhibited poor performance and struggled to converge under ill-conditioned problems, while IPPMM was not significantly affected. Furthermore, regarding the complexity of the two algorithms, we illustrated it through experiments on very large-scale problems and concluded that IPPMM performed worse mainly due to dimension expansion. Besides, the sparsity of matrices could also impact the termination time. In short, the time required would increase as the matrix density goes up. Nevertheless, we still find that ADMM is more sensitive to changes in sparsity, compared with IPPMM. To conclude above, we arrive at the following table:

Problem Characteristics	Algorithm Choice	
	ADMM	IPPMM
Ill-Conditioned		✓
Large-Scale and Ill-Conditioned		✓
Large-Scale but Well-Conditioned	✓	
Dense		✓

Table 7: Algorithm Selection Based on Problem Characteristics

Lastly, There are two potential avenues for future work: the development of more robust numerical algorithms and the formulation for regularized LSTD. Firstly, both of these algorithms have their respective disadvantages as we have seen in the numerical test, which suggests the opportunity to create more resilient alternatives. Secondly, there is the potential for further exploration in refining the formulation of regularized LSTD.

References

[1] M. D. Asic and V. V. Kovacevic-Vujcic. Ill-conditionedness and interior-point methods. *Publications of the Faculty of Electrical Engineering. Series Mathematics*, 11:53–58, 2000.

[2] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, pages 1–137, 2005.

[3] J. A. BOYAN. Technical update: Least-squares temporal difference learning. *Machine Learning*, 49:233–246, 2002.

[4] P. Christopher and P. Ronald. Greedy algorithms for sparse reinforcement learning. *International Conference on Machine Learning*, 2012.

[5] I. Dassios, K. Fountoulakis, and J. Gondzio. A preconditioner for a primal-dual newton conjugate gradient method for compressed sensing problems. *SIAM Journal on Scientific Computing*, 37(6):A2783–A2812, 2015.

[6] C. DERMAN. *Finite State Markovian Decision Processes*. ACADEMIC PRESS New York and London, 1971.

[7] K. Fountoulakis and J. Gondzio. Performance of first- and second-order methods for l1-regularized least squares problems. *Computational Optimization and Applications*, 65:605–635, 2016.

[8] K. Fountoulakis and J. Gondzio. A second-order method for strongly convex l1-regularization problems. *Mathematical Programming*, 156:189–219, 2016.

[9] K. Fountoulakis, J. Gondzio, and P. Zhlobich. Matrix-free interior point method for compressed sensing problems. *Mathematical Programming Computation*, 6:1–31, 2014.

[10] M. Geist and B. Scherrer. l1-penalized projected bellman residual. *European Workshop on Reinforcement Learning*, 2011.

[11] J. George James Minty. On the monotonicity of the grandient of a convex function. 14(1), 1964.

[12] M. W. Hoffman, A. Lazaric, M. Ghavamzadeh, and R. Munos. Regularized least squares temporal difference learning with nested l2 and l1 penalization. *European Workshop on Reinforcement Learning*, 2011.

[13] T. Jaakkola, M. I. Jordan, and S. P. Singh. On the convergence of stochastic iterative dynamic programming algorithms. Technical report, Massachusetts Institute of Technology Artificial Intelligence Laboratory and Center for Biological and Computational Learning, Department of Brain and Cognitive Sciences, 1993.

[14] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1995.

[15] J. Z. Kolter and A. Y. Ng. Regularization and feature selection in least-squares temporal difference learning. pages 102–114, 2011.

[16] M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research 4*, pages 1107–1149, 2003.

[17] M. Loth, M. Davy, and P. Philippe. Sparse temporal difference learning using lasso. 2007.

[18] S. Pougkakiotis and J. Gondzio. An interior point-proximal method of multipliers for convex quadratic programming. *Computational Optimization and Applications*, 78:307–351, 2020.

[19] Z. Qin, W. Li, and F. Janoos. Sparse reinforcement learning via convex optimization. *Proceedings of Machine Learning Research*, 32(2):424–432, 2014.

[20] R. T. Rockafellar. Augmented lagrangians and applications of the proximal point algorithm in convex programming. *MATHEMATICS OF OPERATIONS RESEARCH*, 1(2), 1976.

[21] R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on control and optimization*, 14(5), 1976.

[22] V. D. Simone, D. di Serafino, J. Gondzio, S. Pougkakiotis, and M. Viola. Sparse approximations with interior point methods. *Society for Industrial and Applied Mathematics*, 64(4):954–988, 2022.

[23] J. B. Steven and G. B. Andrew. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22:33–57, 1996.

[24] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2018.

[25] R. S. Sutton, H. R. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvvariand, and E. Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. volume 382 of *ACM International Conference Proceeding Series*. 26th Annual International Conference on Machine Learning, 2009.

[26] F. A. Yaghmaie and F. Gustafsson. Using reinforcement learning for model-free linear quadratic control with process and measurement noises. IEEE 58th Conference on Decision and Control (CDC), 2019.

[27] J. Yang and Y. Zhang. Alternating direction algorithms for l1-problems in compressive sensing. *SIAM Journal on Scientific Computing*, 33(1):250–278, 2021.

[28] H. Yu. Convergence of least squares temporal difference methods under general conditions. Technical report, Dept. Computer Science, University of Helsinki, 2010.

[29] S. Zhao. Mathematical foundation of reinforcement learning. GitHub Link: https://github.com/MathFoundationRL/Book-Mathmatical-Foundation-of-Reinforcement-Learning.

Appendices

A Convergence Theorem of TD(0)-Learning

Theorem A.1. *(Convergence of TD Learning) By applying equation (1.1), v_k will converge to v^* as $k \to \infty$ almost surely in probability if the learning rate sequence is $\sum a_i = \infty$ and $\sum a_i^2 < \infty$. One typical example for such a sequence is to take $a_k = \frac{1}{k}$.*

Proof. This can be proven by applying a variant of Dvoretzky Theorem (in [13], Theorem 1 and 2) with the following two properties having to be verified. Firstly, rewrite the equation (1.1) as following:

$$v_{k+1}(s) = (1 - \alpha_k)v_k(s) + \alpha_k[r_{k+1} + \gamma v_k(s')]$$

Subtracting the optimal value function v_π^* such that the error denotes:

$$\Delta_{k+1}(s) = (1 - \alpha_k)\Delta_k(s) + \alpha_k[r_{k+1} + \gamma v_k(s') - v_\pi(s)] \tag{A.1}$$

- Property 1: $\|\mathbf{E}[e_k(s)|\mathcal{H}_k]\|_\infty \leq \gamma\|\Delta_k(s)\|_\infty$ where $e_k(s) = r_{k+1} + \gamma v_k(s') - v_\pi(s)$ and $\mathcal{H}_k := \{\Delta_k, \Delta_{k-1}, ..., e_{k-1}, ..., \alpha_{k-1}, ...\}$. We have:

$$\begin{aligned}
\mathbf{E}[e_k(s)|\mathcal{H}_k] &\leq \|r_{k+1} + \gamma v_k(s') - v_\pi(s)\|_\infty \\
&= \|r_\pi + \gamma P v_k(s') - r_\pi - \gamma P v_\pi(s)\|_\infty \\
&\leq \gamma\|v_k - v_\pi\|_\infty \\
&= \gamma\|\Delta_k(s)\|_\infty
\end{aligned} \tag{A.2}$$

- Property 2: $Var[e_k(s)|\mathcal{H}_k] \leq C(1 + \|\Delta_k(s)\|_\infty)^2$.

$$\begin{aligned}
Var[e_k(s)|\mathcal{H}_k] &= \mathbf{E}[e_k^2(s)|\mathcal{H}_k] - \mathbf{E}^2[e_k(s)|\mathcal{H}_k] \\
&\leq \|r_{k+1} + \gamma v_k(s') - v_\pi(s)\|_\infty^2 - \|r_{k+1} + \gamma v_k(s') - v_\pi(s)\|_\infty \\
&\leq \gamma\|v_j(s') - v_\pi(s)\|_\infty^2 - \gamma\|v_j(s') - v_\pi(s)\|_\infty \\
&\leq \gamma(\|\Delta_k(s)\|_\infty^2 - \|\Delta_k(s)\|_\infty) \\
&\leq \gamma(\|\Delta_k(s)\|_\infty + 1)^2 - 3\gamma(\|\Delta_k(s)\|_\infty + 1) \\
&\leq C(\gamma) \cdot (\|\Delta_k(s)\|_\infty + 1)^2
\end{aligned} \tag{A.3}$$

Thus, we conclude that a non-increasing sequence $\{\|\Delta_k\|_\infty\}_k$ such that

$$\|\Delta_{k+1}\|_\infty - \|\Delta_k\|_\infty = \alpha_k(\mathbf{E}\|e_k\|_\infty - \|\Delta_k\|_\infty) \leq \alpha_k(\gamma - 1)\|\Delta_k\|_\infty \leq 0$$

Taking the condition $\sum a_i = \infty$ and $\sum a_i^2 < \infty$ into account, the desired result will follow by sending $k \to \infty$. $\square$

B Another Proofs

Definition B.1. *(Lp-Norm of Matrix) If the p-norm ($1 \leq p < \infty$) for vectors is used for both spaces $\mathcal{R}^n$ and $\mathcal{R}^m$, then the corresponding operator norm is*

$$\|A\|_p := \sup_{x \neq \mathbf{0}} \frac{\|Ax\|_p}{\|x\|_p}$$

and following relations (in L2-Norm) hold:

1. $\|A\|_2 = \lambda_{\max}(A^T A) = \sigma_{\max}(A^T A)$
2. $\|Ax\|_2 \leq \sigma_{\max}(A^T A)\|x\|_2$

Proof. The results are followed by definition:

1. Expanding the L2-norm of matrix A:

$$\|A\|_2 = \max_{x \neq \mathbf{0}} \frac{\|Ax\|_{\mathbf{L}_2}}{\|u\|_{\mathbf{L}_2}} = \max_{x \neq \mathbf{0}} \frac{\sqrt{x^T A^T A x}}{\sqrt{x^T x}} = \sqrt{\lambda_{\max}\left(A^T A\right)}$$

2. By Cauchy inequality:

$$\|Ax\|_2 \leq \|A\|_2 \|x\|_2 = \sigma_{\max}(A^T A)\|x\|_2$$

□

Theorem B.1. *The problem*

$$\min_{\omega \in \mathcal{R}^n} \left\{ \|\omega\|_1 + \frac{\mu}{2\tau}\|\omega - \omega^k\|^2 + \mu g^{k-1}(\omega - \omega^k) \right\}$$

is a quadratic approximator to the problem formulated in the second bullet point of section 3.2:

$$\min_{\omega \in \mathcal{R}^n} \left\{ \|\omega\|_1 + \frac{\mu}{2} \left\| \tilde{C}\omega + \tilde{d} - r - \frac{1}{\mu} y \right\|^2 \right\}.$$

Proof. Apply the Quadratic Approximation to the squared l2-norm in

$$\min_{\omega \in \mathcal{R}^n} \left\{ \|\omega\|_1 + \frac{\mu}{2} \left\| \tilde{C}\omega + \tilde{d} - r - \frac{1}{\mu} y \right\|^2 \right\}.$$

We get

$$\omega^* = \arg\min_{\omega \in \mathcal{R}^n} \left\{ \|\omega\|_1 + \frac{\mu}{2\tau} \left\| \omega - (\omega^k - \tau g^{k-1}) \right\|^2 \right\} = \arg\min_{\omega \in \mathcal{R}^n} \left\{ \|\omega\|_1 + \frac{\mu}{2\tau}\|\omega - \omega^k\|^2 + \mu g^{k-1}(\omega - \omega^k) \right\},$$

where we use a step parameter τ to replace the second-order gradient to $\left\|\omega - (\omega^k - \tau g^{k-1})\right\|^2$. In Proximal Gradient Descent algorithm, it constructs a similar problem by using the same technique. □

Theorem B.2. *(Singular Value Decomposition) Assume $A \in \mathcal{R}^{m\times n}$ with unitary matrices $V \in \mathcal{R}^{n\times n}$, $U \in \mathcal{R}^{m\times m}$ and corresponding singular matrix $\Sigma \in \mathcal{R}^{m\times n}$. Then*

$$U^T AV = \Sigma$$

which is Singular Value Decomposition (SVD) and such decomposition exists.

Proof. Define the transpose matrix of $\bar{A} := A^T \in \mathcal{R}^{n\times m}$ such that $m \ll n$. Then, we have $\bar{A}^T\bar{A} \in \mathcal{R}^{m\times m}$ where $r \leq \min\{m, n\}$ is the rank of matrix $\bar{A}^T\bar{A}$. By Spectral Decomposition,

$$(\bar{A}^T\bar{A})\bar{V} = \bar{V}\Lambda, \textit{where } \Lambda := \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} \textit{ and } \bar{V} := \begin{bmatrix} \bar{v}_1 & \bar{v}_2 & \cdots & \bar{v}_n \end{bmatrix}.$$

If $r < m$ strictly holds, then $\Lambda_{i,i} > 0, \forall i \in \{1, 2, ..., r\}$ but $\Lambda_{i,i} = 0, \forall i \in \{r+1, r+2, ..., n\}$. By defining $\bar{u}_i = \frac{\bar{A}\bar{V}_i}{\sigma_i}$ and $\sigma_i = \sqrt{\lambda_i}$, we have:

$$\bar{A}^T\bar{u}_i = \bar{V}_i\sigma_i, \textit{ and } \bar{A}\bar{A}^T\bar{u}_i = \sigma_i\bar{A}\bar{V}_i = \lambda_i\bar{u}_i.$$

Hence, each $\bar{u}_i$ is eigenvector of $\bar{A}\bar{A}^T \in \mathcal{R}^{n\times n}$ with orthonormality

$$\bar{u}_i^T\bar{u}_j = \frac{\bar{v}_i^T\bar{A}^T\bar{A}\bar{v}_j}{\sigma_i\sigma_j} = \bar{v}_i^T\bar{v}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases}$$

Now,

$$[\bar{U}^T\bar{A}\bar{V}]_{i,j} = \bar{u}_i^T A v_j = \sigma_j\bar{u}_i^T\bar{u}_j = \begin{cases} \sigma_i & \text{if } i = j \leq r \\ 0 & \text{Otherwise} \end{cases} = \left[\begin{array}{cccc|c} \sigma_1 & & & & \\ & \sigma_2 & & & \\ & & \sigma_3 & & \\ & & & \ddots & \\ & & & \sigma_m & \end{array}\right]^T \begin{array}{c} \\ \\ \\ \\ \\ \end{array} \begin{array}{c} \\ \\ \\ \\ \mathbf{0_{m,n-m}} \\ \\ \end{array} := \Sigma^T.$$

The desired result will follow by taking transpose

$$[\bar{U}^T\bar{A}\bar{V}]^T = \bar{V}^T\bar{A}^T\bar{U} := U^T AV = \Sigma.$$

□