이학석사학위논문

A Comparative Study of the Proximal Distance Algorithm and the Primal Dual Hybrid Gradient method for Statistical Problems

통계적 문제에 대한 근위 거리 알고리즘과 원시-쌍대 복합 경사법의 비교 연구

2020 년 2 월

서울대학교 대학원

통계학과

한 승 지

이학석사학위논문

A Comparative Study of the Proximal Distance Algorithm and the Primal Dual Hybrid Gradient method for Statistical Problems

통계적 문제에 대한 근위 거리 알고리즘과 원시-쌍대 복합 경사법의 비교 연구

2020 년 2 월

서울대학교 대학원

통계학과

한 승 지

A Comparative Study of the Proximal Distance Algorithm and the Primal Dual Hybrid Gradient method for Statistical Problems

통계적 문제에 대한 근위 거리 알고리즘과 원시-쌍대 복합 경사법의 비교 연구

지도교수 원 중 호

이 논문을 이학석사 학위논문으로 제출함

2019 년 12 월

서울대학교 대학원

통계학과

한 승 지

한승지의 이학석사 학위논문을 인준함

2019 년 12 월

Abstract

In this thesis, we compare the proximal distance algorithm and the primal dual hybrid gradient methods for solving convex constrained problems with linear operators in constraints. Unlike other algorithms containing inner minimization subproblem, both algorithms can be applied with simple update rules. Algorithms are implemented and compared with existing software through six examples: linear programming, constrained least squares, estimating the closest kinship matrix, projection onto a second-order cone, low rank matrix completion, and logistic regression with partially ordered constraints. The solution of the proximal distance algorithm would lose optimality and stability when parameters are chosen inappropriately, while the output of the primal dual hybrid gradient methods is robust to the selection of parameters. Numerical results show that the primal dual hybrid gradient methods usually outperform and the proximal distance algorithm is still competitive with existent programs.

Keywords: Convex optimization, Proximal distance algorithm, Primal dual hybrid gradient, Statistical computing

Student Number: 2018-27696

Contents

List of Tables

Chapter 1

Introduction

In this thesis, we focus on algorithms for convex optimization problems that arise in statistics. Especially, we consider problems in the following form:

$$\min_{x \in \mathbb{R}^n} f(x) \text{ subject to } Dx \in C, \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be convex, closed and proper, C is a convex set, and $D \in \mathbb{R}^{m \times n}$ is a linear operator. We further assume that the projection operator $P_C(x) = \{y \in C : \|x - y\| = \inf_{y \in C} \|x - y\|\}$ can be easily computed.

It is difficult to solve problem (1.1) directly due to existence of D in the constraint. If $D = I$ and f is differentiable, a well-known method for solving problem (1.1) is the proximal gradient algorithm [13]. Note that problem (1.1) with $D = I$ can be viewed as an unconstrained but non-smooth problem:

$$\min_{x \in \mathbb{R}^n} \; f(x) + \delta_C(x),$$

where $\delta_C(x)$ is an indicator function which is 0 if x is on C and ∞ otherwise. Then, the proximal gradient algorithm reduces to the projected gradient method [18]:

$$\begin{aligned} x_{k+1} &= \operatorname*{argmin}_{x\in\mathbb{R}^n} \left\{ f(x_k) + \nabla f(x_k)^T(x - x_k) + \frac{1}{2\rho_k}\|x - x_k\|_2^2 + \delta_C(x) \right\} \\ &= \operatorname{prox}_{\rho_k\delta_C}(x_k - \rho_k\nabla f(x_k)) = P_C\left(x_k - \rho_k\nabla f(x_k)\right), \end{aligned}$$

where $\operatorname{prox}_\phi(x) = \operatorname{argmin}_{y\in\mathbb{R}^n}(\phi(y) + \frac{1}{2}\|y - x\|_2^2)$ is the proximity operator for a convex function ϕ.

The alternating direction method of multipliers (ADMM) [6] is one of the most popular solution method when linear operator D is preesent. ADMM introduces a new variable y and a constraint $y = Dx$, thus (1.1) becomes

$$\min_{(x,y)\in\mathbb{R}^n\times\mathbb{R}^m} f(x) + \delta_C(y) \text{ subject to } Dx - y = 0. \tag{1.2}$$

Forming the augmented Lagrangian of (1.2) and updating variables in alternating directions give the ADMM iteration

$$\begin{aligned} x_{k+1} &= \operatorname*{argmin}_{x\in\mathbb{R}^n} \left(f(x) + \frac{\rho}{2}\|Dx - y_k + u_k\|_2^2 \right) \qquad (1.3) \\ y_{k+1} &= \operatorname*{argmin}_{y\in\mathbb{R}^m} \left(\delta_C(y) + \frac{\rho}{2}\|Dx_{k+1} - y + u_k\|_2^2 \right) \\ &= \operatorname{prox}_{\rho^{-1}\delta_C}(Dx_{k+1} + u_k) = P_C(Dx_{k+1} + u_k) \\ u_{k+1} &= u_k + (Dx_{k+1} - y_{k+1}). \end{aligned}$$

While ADMM is a versatile tool for optimization problems, the update (1.3) could be a burden when the x-update are not representable in closed form.

The goal of this thesis is to review two classes of algorithm for solving (1.1) in the presence of the linear operator D. One is the proximal distance algorithm proposed by Keys, Zhou, and Lange [22, 26] and the other is the primal dual hybrid gradient method, which is first suggested by Zhu and Chan [36] and later studied in many literature [11, 14, 16, 27, 35]. Additionally, we implemented them to several examples and compared the performance with other existent solvers. Since both algorithms only involve a simple proximity operator of f and the projection map onto C, they take two advantages over ADMM: First, they avoid inner minimization subproblems. Second, they can exploit broad research about proximity operators and projection maps. In the rest of this thesis, we assume that f satisfies one of the following conditions:

(C1) The proximity operator of f can be computed easily,

(C2) f is differentiable and its gradient is L_f-Lipschitz.

If the proximity operator of the objective f is not able to be expressed in closed form, that is f does not satisfy (C1), both algorithms can be derived in other simple formulations under (C2).

The outline of this thesis is as follows. In Chapter 2, we describe the proximal distance algorithm and apply it for solving problem (1.1). Convergence rate of the proximal distance algorithm and its acceleration are also reviewed. In Chapter 3, we consider three instances of the primal dual hybrid gradient method in different problem setting, and study their convergence rates and regions. Chapter 4 covers six examples in the form of (1.1). We provide and implement solution methods for of each example using the two algorithm classes and compare the performance with some other benchmarks. We summarize and discuss our results in Chapter 5.

Chapter 2

Proximal Distance Algorithm

Proximal distance algorithm is an algorithm combining Courant's penalty method [3, 15] and the majoraization-minimization (MM) principle [21, 25], which deals with the generic problem:

$$\min_{x \in \mathbb{R}^n} f(x) \text{ subject to } x \in C,$$

where C is a closed set and not necessarily convex.

The classical penalty method attacks (1.1) by minimizing the objective with a proper penalty function oriented from the constraint, thus it minimizes unconstrained loss $f(x) + \rho q(x)$ where $q(x)$ is a nonnegative function that is 0 on C and $\rho > 0$. As ρ increases, the solution goes close to the optimal solution of (1.1).

At each iteration in minimizing $f(x)$, the MM principle constructs a surrogate function $g(x|x_k)$ majorizing $f(x)$ at anchor x_k rather than optimizing $f(x)$ directly. The minimizer of $g(x|x_k)$ is selected to the next iteration x_{k+1}. The surrogate $g(x|x_k)$ satisfies two conditions: the tangency condition

$g(x_k|x_k) = f(x_k)$ and the domination condition $g(x|x_k) \geq f(x)$ for all feasible x. For each iteration, the update satisfies the descent property

$$f(x_{k+1}) \leq g(x_{k+1}|x_k) \leq g(x_k|x_k) = f(x_k),$$

guaranteeing convergence. If the problem is the maximization of the objective, one builds a surrogate $g(x|x_k)$ that minorizes $f(x)$ and gets x_{k+1} maximizing the $g(x|x_k)$.

For detailed derivation of the proximal distance algorithm, suppose that the constraint set C can be expressed as an intersection of closed sets $\cap_{i=1}^{N} C_i$. To apply penalty method, we consider a convex combination of squared distance functions

$$q(x) = \frac{1}{2}\sum_{i=1}^{N} \alpha_i \mathrm{dist}(x, C_i)^2,$$

where $\mathrm{dist}(x, C) = \inf\limits_{y \in C} \|x - y\|^2$, $\sum_{i=1}^{N} \alpha_i = 1$, and $0 \leq \alpha_i \leq 1$ for $i = 1, \cdots, N$. For given x_k, it is easy to majorize distance functions

$$\frac{1}{2}\sum_{i=1}^{N} \alpha_i \mathrm{dist}(x, C_i)^2 \leq \frac{1}{2}\sum_{i=1}^{N} \alpha_i \|x - P_{C_i}(x_k)\|^2,$$

which gives the surrogate function of $f(x)$ as

$$\begin{aligned} g(x|x_k) &= f(x) + \frac{\rho}{2}\sum_{i=1}^{N} \alpha_i \|x - P_{C_i}(x_k)\|^2 \\ &= f(x) + \frac{\rho}{2}\left\| x - \sum_{i=1}^{N} \alpha_i P_{C_i}(x_k) \right\|^2 + c_k \end{aligned}$$

for a constant c_k irrelevant to x. Therefore, the proximal distance algorithm iteratively updates

$$\begin{aligned} x_{k+1} &= \operatorname*{argmin}_{x \in \mathbb{R}^n} g(x|x_k) \\ &= \mathrm{prox}_{\rho^{-1} f}\left(\sum_{i=1}^{N} \alpha_i P_{C_i}(x_k) \right), \end{aligned}$$

under (C1).

Even if we do not assume that the objective function f is neither convex nor smooth and C is convex, broad theoretical results on proximal maps and projection operators support this algorithm. Some numerical results are adduced that the proximal distance algorithm works well on non-convex objectives and constraints [22].

It is not only the penalties but also the objective that can have the surrogate function, if needed. Some functions like logistic loss function $f(x) = \log(1 + \exp(-x))$ do not have closed form representations of the proximity operators. Although they are not 'proximable', we can derive the proximal distance update under (C2). Considering a quadratic majorization

$$f(x) \leq f(x_k) + \nabla f(x_k)^T(x - x_k) + \frac{L_f}{2}\|x - x_k\|^2 \tag{2.1}$$

for $f(x)$ and combining both majorization of the objective and penalties, the surrogate becomes

$$g(x|x_k) = f(x_k) + \nabla f(x_k)^T(x - x_k) + \frac{L_f}{2}\|x - x_k\|^2 + \frac{\rho}{2}\left\|x - \sum_{i=1}^{m} \alpha_i P_{C_i}(x_k)\right\|^2.$$

Therefore, the updated proximal distance iterate is induced from minimizing $g(x|x_k)$:

$$x_{k+1} = \frac{1}{L_f + \rho}\left(-\nabla f(x_k) + L_f x_k + \rho \sum_{i=1}^{m} \alpha_i P_{C_i}(x_k)\right).$$

To solve problem (1.1) by the proximal distance algorithm, we convert it to an unconstrained penalized problem

$$f(x) + \frac{\rho}{2}\mathrm{dist}(Dx, C)^2 \tag{2.2}$$

and its surrogate function becomes

$$g(x|x_k) = f(x) + \frac{\rho}{2}\|Dx - P_C(Dx_k)\|^2. \tag{2.3}$$

If one can attain the minimizer of the surrogate function (2.3) easily, it would be a proximal distance update. Unfortunately, minimizing (2.3) often involves an inner minimization subproblem much like ADMM. There are at least two bypasses. One is exploiting a quadratic majoriziation (2.1) of $f(x)$, if possible. Then, the surrogate becomes

$$g(x|x_k) = f(x_k) + \nabla f(x_k)^T(x - x_k) + \frac{L_f}{2}\|x - x_k\|^2 + \frac{\rho}{2}\|Dx - P_C(Dx_k)\|^2,$$

which results in the update

$$x_{k+1} = (\rho D^T D + L_f I)^{-1}(L_f x_k + \rho D^T P_C(Dx_k) - \nabla f(x_k)).$$

Another is the strategy of variable splitting. We introduce $y = Dx$ into problem (2.2) and solve

$$\min_{x\in\mathbb{R}^n} f(x) + \frac{\rho}{2}\text{dist}(y, C)^2 \text{ subject to } y = Dx.$$

The constrained problem can be solved approximately by minimizing

$$f(x) + \frac{\rho}{2}\text{dist}(y, C)^2 + \frac{\rho}{2}\text{dist}(z, M)^2,$$

where $M = \{z = (x, y) : y = Dx\}$. Let u_k and v_k be subvectors of $P_M(z)$ corresponding to x_k and y_k respectively, then the proximal distance update is

$$\begin{aligned}
x_{k+1} &= \underset{x\in\mathbb{R}^n}{\text{argmin}}\ f(x) + \frac{\rho}{2}\|x - u_k\|^2 \\
&= \text{prox}_{\rho^{-1}f}(u_k) \\
y_{k+1} &= \underset{y\in\mathbb{R}^m}{\text{argmin}}\ \frac{\rho}{2}\|y - P_C(y_k)\|^2 + \frac{\rho}{2}\|y - v_k\|^2 \\
&= \frac{1}{2}(P_C(y_k) + v_k).
\end{aligned}$$

There are two ways to get the projection $(u, v) = P_M(z)$. Assuming that $D \in \mathbb{R}^{m\times n}$, one solves

$$p(u) = \frac{1}{2}\|u - x\|^2 + \frac{1}{2}\|Du - y\|^2$$

by using the definition of projection operator. Then the solution becomes

$$u = (D^T D + I_n)^{-1}(x + D^T y) \text{ and } v = Du.$$

Otherwise, one can consider the Lagrangian

$$\mathcal{L}(u, v, \lambda) = \frac{1}{2}\|u - x\|^2 + \frac{1}{2}\|v - y\|^2 + \lambda^T(Du - v),$$

which leads the stationary equation

$$\begin{aligned} 0 &= u - x + D^T\lambda \\ 0 &= v - y - \lambda. \end{aligned}$$

Then the projection operator $P_M(z)$ can be represented as

$$u = x - D^T(DD^T + I_m)^{-1}(Dx - y) \text{ and } v = y + (DD^T + I_m)^{-1}(Dx - y).$$

Both result involves matrix inversion, thus one should choose proper solution depending on the size of D.

Keys, Zhou, and Lange [22] provide some convergence analysis of the proximal distance algorithm in presence of convexity. Let $h_\rho(x) = f(x) + \frac{\rho}{2}\text{dist}(x, C)^2$ and $x^\star$ be the optimal solution of $h_\rho(x)$. For fixed $\rho > 0$, the proximal distance iterates converge to $x^\star$ with an $O(\rho k^{-1})$ convergence rate. To achieve faster convergence, a scheme of Nesterov's acceleration [30] can be applied. Given our algorithm map $M(x)$, Nesterov's acceleration yields an update

$$\begin{aligned} z_k &= x_k + \frac{k-1}{k+d-1}(x_k - x_{k-1}) \\ x_{k+1} &= M(z_k), \end{aligned}$$

where d is typically chosen to be 3. It is known that Nesterov's acceleration for the general proximal gradient algorithm achieves an $O(k^{-2})$ [34] and some numerical experiments support that results from acceleration have better accuracy.

Convergence analysis of the proximal distance algorithm is closely related to that of the classical penalty method [3]. If f is continuous and coercive and C is a compact set, then the proximal distance iterates x_k are bounded and $\text{dist}(x_k, C)^2$ converges to 0 as ρ_k tends to ∞. Similarly, the solutions y_k of $\min_x \ h_{\rho_k}(x)$ are bounded and $\text{dist}(y_k, C)^2$ converges to 0. Recall that the proximal distance iterates have $O(\rho k^{-1})$ convergence rate for a fixed ρ. Therefore, one should increase ρ_k slowly to ∞. Summing up the results, we organize the proximal distance algorithm in Algorithm 1.

Algorithm 1: Proximal distance algorithm with Nesterov's acceleration (Keys, Zhou, and Lange, 2019)

```
Input  : $\rho_{initial}$, $\rho_{inc}$, $\rho_{max}$          # setting for penalty parameter
         $K_{max}$, $k_\rho$                    # setting for the number of iterations
         $\epsilon_{loss}$, $\epsilon_{dist}$                # setting for termination
Output: $x^{opt}$ : solution of $\min_x f(x)$ subject to $x \in C$
1  $\rho \leftarrow \rho_{initial}$
2  $x_0 = x_1 = 0$
3  $q_0 = q_1 = \inf$
4  for $k = 2, \cdots, K_{max}$ do
5      $z_k \leftarrow x_{k-1} + \frac{k-1}{k+2}(x_{k-1} - x_{k-2})$        # Nesterov's acceleration
6      $x_k \leftarrow \text{prox}_{\rho^{-1} f}(P_C(z_k))$          # Proximal distance update
7      $q_k \leftarrow f(x_k)$
8      $d_k \leftarrow \text{dist}(x_k, C)$
9      if $|q_k - q_{k-1}| < \epsilon_{loss}$ and $d_k < \epsilon_{dist}$ then
10         return $x^{opt} \leftarrow x_k$
11     else
12         if $mod(k_\rho, k) = 0$ then
13             $\rho \leftarrow \min(\rho_{max}, \rho \times \rho_{inc})$      # Update penalty constant
14             $x_{k-1} \leftarrow x_k$
15         end
16     end
17 end
```

Chapter 3

Primal Dual Hybrid Gradient method

In this chapter, we consider the saddle point problem

$$\min_{x\in\mathbb{R}^n}\max_{y\in\mathbb{R}^m}\mathcal{L}(x,y)=\min_{x\in\mathbb{R}^n}\max_{y\in\mathbb{R}^m} y^TDx+f(x)-h^\star(y), \tag{PD}$$

where $f:\mathbb{R}^n\to[0,\infty]$ and $h:\mathbb{R}^m\to[0,\infty]$ are proper, convex, and closed functions, $h^\star(y)$ is a convex conjugate of h, and $D\in\mathbb{R}^{m\times n}$. Under a mild regularity condition, there exists a solution $(x^\star,y^\star)$ of (PD). Indeed, $x^\star$ is a primal solution of

$$\min_{x\in\mathbb{R}^n} f(x)+h(Dx) \tag{P}$$

and $y^\star$ is a solution of the dual formulation

$$\max_{y\in\mathbb{R}^m}\ -f^\star(-D^Ty)-h^\star(y). \tag{D}$$

We will assume that at least one solution $(x^\star, y^\star)$ of (PD) exists, which therefore satisfies the first-order optimality condition

$$\begin{aligned} -D^T y^\star &\in \partial f(x^\star) \\ Dx^\star &\in \partial h^\star(y^\star), \end{aligned} \tag{3.1}$$

where ∂f and $\partial h^\star$ are the subdifferentials of f and $h^\star$ respectively. Note that our problem (1.1) is equivalent to solving

$$\min_{x\in\mathbb{R}^n} f(x) + \delta_C(Dx).$$

Chambolle and Pock presented a first-order primal dual algorithm [11] summarized in Algorithm 2.

Algorithm 2 contains calculation of $\mathrm{prox}_{\tau f}$ and $\mathrm{prox}_{\sigma h^\star}$. In case of $h = \delta_C$ and C is convex, Moreau's decomposition entails

$$\mathrm{prox}_{\sigma h^\star}(y) = y - \sigma \mathrm{prox}_{\sigma^{-1}h}(\sigma^{-1}y) = y - \sigma P_C(\sigma^{-1}y). \tag{3.2}$$

Therefore, Algorithm 2 only involves the proximity operator of f and the projection map onto C.

If one skips the inertia step $\bar{x}_{k+1} = 2x_{k+1} - x_k$ in line 8 of Algorithm 2, the proposed algorithm reduces to the classical Arrow-Hurwicz method [1]. Zhu and Chan applied the method to the Rudin, Osher, and Fatemi (ROF) image denoising problem [32] in [36] and Esser, Zhang, and Chan generalized it to solve (PD) in [16]. He and Yuan [20] studied the role of a general inertia step.

For convergence analysis of primal dual hybrid gradient methods, a preduality gap function $\mathcal{G}(z^\star, z) = \mathcal{L}(x^\star, y) - \mathcal{L}(x, y^\star)$, where $z = (x, y)$ and $z^\star = (x^\star, y^\star)$, is used in general. It is because that the nonnegativity of the duality gap $\mathcal{G}^\star(z^\star) = \sup_z \mathcal{G}(z^\star, z)$ guarantees that $z^\star$ is a primal dual solution to (PD) in the presence of bounded domains. That is, the convergence rate

Algorithm 2: Primal dual hybrid algorithm 1 (Chambolle and Pock, 2011)

```
Input  : $\tau, \sigma$                                   # setting for step sizes
         $K_{max}$                                        # setting for the number of iterations
         $\epsilon_{loss}, \epsilon_{var}$                # setting for termination
Output: $(x^{opt}, y^{opt})$ : solution of $\min_x \max_y y^T Dx + f(x) - h^\star(y)$
1  $\rho \leftarrow \rho_{initial}$
2  $x_0 = x_1 = 0$
3  $y_0 = y_1 = 0$
4  $q_0 = q_1 = \inf$
5  for $k = 2, \cdots, K_{max}$ do
6      $y_{k+1} = \text{prox}_{\sigma h^\star}(y_k + \sigma D\bar{x}_k)$
7      $x_{k+1} = \text{prox}_{\tau f}(x_k - \tau D^T y_{k+1})$
8      $\bar{x}_{k+1} = 2x_{k+1} - x_k$
9      $q_k \leftarrow f(x_k)$
10     $d_x \leftarrow \|x_k - x_{k-1}\|$
11     $d_y \leftarrow \|y_k - y_{k-1}\|$
12     if $|q_k - q_{k-1}| < \epsilon_{loss}$ and $d_x < \epsilon_{var}$ and $d_y < \epsilon_{var}$ then
13         return $x^{opt} \leftarrow x_k$
14     end
15 end
```

of algorithm can be measured by how fast $\mathcal{G}^\star(z^\star)$ approaches to zero. The convergence rate of gap function is often analyzed in terms of an averaged solution sequence, yielding an 'ergodic rate' [11, 23, 24, 27]. Chambolle and Pock showed that a sequence of solutions (x_k, y_k) of Algorithm 2 is bounded and achieves $O(k^{-1})$ ergodic convergence rate when $\tau\sigma\|D\|^2 < 1$.

As mentioned earlier, there are functions that do not have the proximity operator in closed form, which means that it is unable to use Algorithm 2. Under (C2), Loris and Verhoeven [27], Condat [14], and Vũ [35] considered replacing a proximity operator step with a gradient descent step. In this line of research, Ko, Yu, and Won proposed an unified class of algorithms solving (PD) from a perspective of monotone operator theory [24]. By our assumptions, the first-order optimality condition (3.1) becomes

$$\begin{aligned} 0 &= \nabla f(x^\star) + D^T y^\star \\ y^\star &\in \partial h(Dx^\star), \end{aligned} \tag{3.3}$$

which is equivalent to an inclusion problem

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} \in \begin{pmatrix} \nabla f & D^T \\ -D & \partial h^\star \end{pmatrix} \begin{pmatrix} x^\star \\ y^\star \end{pmatrix} =: T(z^\star).$$

The set valued operator T is split into $T = F + G$, where F is a maximally monotone operator

$$F = \begin{pmatrix} O & D^T \\ -D & \partial h^\star \end{pmatrix}$$

and G is a $1/L_f$-cocoercive operator

$$G = \begin{pmatrix} \nabla f & O \\ O & O \end{pmatrix}$$

[2]. A preconditioned forward-backward splitting [33] for solving (3.3) is

$$\begin{aligned}\tilde{z}_{k+1} &= (I + M^{-1}F)^{-1}(I - M^{-1}G)(z_k) \qquad (3.4)\\ z_{k+1} &= (1-\rho_k)z_k + \rho_k \tilde{z}_{k+1},\end{aligned}$$

where $z_k = (x_k, y_k), \tilde{z}_k = (\tilde{x}_k, \tilde{y}_k)$, and the preconditioner $M \succ 0$. Let

$$M = \begin{pmatrix} \frac{1}{\tau}I & C^T \\ C & \tau(CC^T - DD^T) + \frac{1}{\sigma}I \end{pmatrix}$$

and choose C such that $CD^T = DC^T$, then the unified algorithm class summarized on Algorithm 3 is generated from (3.4).

Ko, Yu, and Won [24] showed that Algorithm 3 converges if

$$\frac{1}{\tau} > \frac{L_f}{2} \text{ and } \left(\frac{1}{\tau} - \frac{L_f}{2}\right)\left(\frac{1}{\sigma} - -\tau\|D\|_2^2\right) > \frac{\tau L_f}{2}\|C\|_2^2. \qquad (3.5)$$

Convergence rate of Algorithm 3 is also provided. Ko, Yu, and Won also proved that it satisfies $O(k^{-1})$ ergodic convergence rate of the pre-duality gap and $o(1/\sqrt{k+1})$ convergence rate for the unaveraged solution. Moreover, they proposed the acceleration of Algorithm 3, which entails $O(L_f k^{-2} + \|D\|k^{-1})$ convergence rate. However, algorithm without an acceleration may converge faster in practice due to its computational cost.

For implementation, we consider two special cases of Algorithm 3. Letting $C = O$, we can recover the algorithm

$$\begin{aligned}\tilde{x}_{k+1} &= x_k - \tau(\nabla f(x_k) + D^T y_k) \qquad \text{(LV)}\\ y_{k+1} &= (1-\rho_k)y_k + \rho_k \text{prox}_{\sigma h^\star}(y_k + \sigma D\tilde{x}_{k+1})\\ x_{k+1} &= (1-\rho_k)x_k + \rho_k(\tilde{x}_{k+1} - \tau D^T(y_{k+1} - y_k)),\end{aligned}$$

which was formerly studied by Loris and Verhoeven [27]. With $C = -D$,

Algorithm 3: Primal dual hybrid method 2 (Ko, Yu, and Won, 2019)

```
Input  : $\tau, \sigma, \{\rho_k\}$                # setting for step sizes
         $C$                                       # precondition matrix
         $K_{max}$                                 # setting for the number of iterations
         $\epsilon_{loss}, \epsilon_{var}$         # setting for termination
Output: $(x^{opt}, y^{opt})$ : solution of $\min_x \max_y y^T Dx + f(x) - h^\star(y)$
1  $\rho \leftarrow \rho_{initial}$
2  $x_0 = x_1 = 0$
3  $y_0 = y_1 = 0$
4  $q_0 = q_1 = \text{inf}$
5  for $k = 2, \cdots, K_{max}$ do
6      $\tilde{y}_{k+1} = \text{prox}_{\sigma h^\star}(\sigma D x_k + \tau\sigma(C - D)\nabla f(x_k) + (\tau\sigma D(C - D)^T + I)y_k)$
7      $\tilde{x}_{k+1} = x_k - \tau(\nabla f(x_k) - C^T y_k + (C + D)^T \tilde{y}_{k+1})$
8      $x_{k+1} = (1 - \rho_k)x_k + \tilde{x}_{k+1}$
9      $y_{k+1} = (1 - \rho_k)y_k + \tilde{y}_{k+1}$
10     $q_k \leftarrow f(x_k)$
11     $d_x \leftarrow \|x_k - x_{k-1}\|$
12     $d_y \leftarrow \|y_k - y_{k-1}\|$
13     if $|q_k - q_{k-1}| < \epsilon_{loss}$ and $d_x < \epsilon_{var}$ and $d_y < \epsilon_{var}$ then
14         return $x^{opt} \leftarrow x_k$
15     end
16 end
```

Algorithm 3 becomes

$$\begin{aligned}
\tilde{x}_{k+1} &= x_k - \tau(\nabla f(x_k) + D^T y_k) \\
x_{k+1} &= (1-\rho_k)x_k + \rho_k \tilde{x}_{k+1} \\
y_{k+1} &= (1-\rho_k)y_k + \rho_k \mathrm{prox}_{\sigma h^\star}(y_k + \sigma D(2\tilde{x}_{k+1} - x_k)),
\end{aligned} \tag{CV}$$

which is due to Condat [14] and Vũ [35]. Region of parameters τ and σ for convergence is directly induced from (3.5) and selected C.

Condat [14] and Vũ [35] studied a generalized primal dual hybrid gradient method that solves an extension of (PD):

$$\min_{x\in\mathbb{R}^n} f(x) + g(x) + h(Dx),$$

where f is a function with assumptions in Algorithm 3, $g : \mathbb{R}^n \to [0,\infty]$, and $h : \mathbb{R}^m \to [0,\infty]$ are a proper, convex, and closed function with closed form proximity operator and $D \in \mathbb{R}^{m\times n}$. The proposed algorithm is summarized in Algorithm 4. Since the over-relaxation term $2\tilde{x}_{k+1} - x_k$ imposes asymmetry of Algorithm 4, one can generate its dual version by switching the role of the primal variable x and the dual variable y.

It is known that Algorithm 4 converges if $(1/\tau - \sigma\|D\|^2) \geq L_f/2$. Chambolle and Pock showed that it has $O(k^{-1})$ ergodic convergence rate [12]. Moreover, Ko and Won proposed a class of accelerated algorithm that achieves known optimal rate $O(L_f k^{-2} + \|D\|_2 k^{-1})$ [23]. Under specific parameter setting, the accelerated algorithm reduces to Algorithm 4. For the reason noted earlier, we only implemented algorithm without an acceleration in following experiments.

It is worth noting that Algorithm 4 is a generalization of former algorithms. If $f \equiv 0$, it reduces to Algorithm 2 with relaxation steps. Since $\mathrm{prox}_{\tau g}(x) = x$ when $g \equiv 0$, Algorithm CV can be interpreted as a special case of Algorithm 4 with $g \equiv 0$.

Algorithm 4: Primal dual hybrid gradient method 3 (Condat, 2013)

```
Input  : $\tau, \sigma, \{\rho_k\}$                # setting for step sizes
         $C$                                        # precondition matrix
         $K_{max}$                                  # setting for the number of iterations
         $\epsilon_{loss}, \epsilon_{var}$          # setting for termination
Output: $(x^{opt}, y^{opt})$ :
         solution of $\min_x \max_y y^T Dx + f(x) + g(x) - h^\star(y)$
1  $\rho \leftarrow \rho_{initial}$
2  $x_0 = x_1 = 0$
3  $y_0 = y_1 = 0$
4  $q_0 = q_1 = \inf$
5  for $k = 2, \cdots, K_{max}$ do
6      $\tilde{x}_{k+1} = \mathrm{prox}_{\tau g}(x_k - \tau(\nabla f(x_k) + D^T y_k))$
7      $\tilde{y}_{k+1} = \mathrm{prox}_{\sigma h^\star}(y_k + \sigma D(2\tilde{x}_{k+1} - x_k))$
8      $x_{k+1} = (1 - \rho_k)x_k + \tilde{x}_{k+1}$
9      $y_{k+1} = (1 - \rho_k)y_k + \tilde{y}_{k+1}$
10     $q_k \leftarrow f(x_k)$
11     $d_x \leftarrow \|x_k - x_{k-1}\|$
12     $d_y \leftarrow \|y_k - y_{k-1}\|$
13     if $|q_k - q_{k-1}| < \epsilon_{loss}$ and $d_x < \epsilon_{var}$ and $d_y < \epsilon_{var}$ then
14         return $x^{opt} \leftarrow x_k$
15     end
16 end
```

Chapter 4

Numerical Experiments

In this chapter, we compare the performance of the proximal distance algorithm and the primal dual hybrid gradient methods through six numerical experiments. First four experiments of the proximal distance algorithm is already conducted in [22], refined to accomplish better accuracy and speed in this thesis. We provide how we formulate each problem appropriately and how projection maps and proximity operators are induced. Programming details like tuning schedules, selected parameters, and declaring convergence vary with problems are documented on Appendix. All programs were coded in the Julia programming language v1.2.0. Additionally, we used the R programming language v3.5.3 to conduct the stratified sampling in Chapter 4.6. For comprehensive comparison, we also solved problems by using existing software. If available, we used Julia modules 'MathProgBase.jl' and 'Convex.jl' to interface with several commercial solvers.

Algorithm parameters need caution. When employing the proximal distance algorithm, one should choose tuning schedule $\{\rho_k\}$ carefully due to the instability of the algorithm. If not, the updates would not converge to the optimal solution nor obtain feasibility. To allay the vulnerability, we started tests from small $\rho_{initial}$ and grow it by ρ_{inc} slowly until the algorithm converges. The choice of parameters τ and σ is rather easy in case of the primal dual hybrid gradient methods. It is known that the algorithm converges faster when parameters are selected near the boundary of convergence region [28]. Therefore, it is enough to pick τ first and compute σ from the region. Moreover, the choice of parameters has little effect on optimality or feasibility of solution.

4.1 Linear Programming

The first example is linear programming (LP), in which the loss and constraints are all affine. Several problems including ℓ_1, ℓ_∞, and quantile loss/constrain combinations can be reformulated as LP. For example, ℓ_1, ℓ_∞, and quantile regression and ℓ_1-penalized support vector machine can be formulated as LP. A general LP can be represented as standard form

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} v^T x \\ &\text{subject to } Ax = b \\ &\qquad\qquad x \geq 0, \end{aligned} \tag{4.1}$$

where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ [7]. There are a variety of effective methods for solving LP. In here, we compare those solvers with the proximal distance algorithm and the primal-dual hybrid gradient methods.

Using the proximal distance algorithm, LP can be solved in three ways. First, the affine constraints $Ax = b$ can be folded into the domain of the loss, thus the problem becomes

$$\min_{x \in \mathbb{R}^n} v^T x + \lambda^T (Ax - b)$$

$$\text{subject to } x \geq 0.$$

Then, one can generate the surrogate

$$g(x|x_k) = v^T x + \lambda^T (Ax - b) + \frac{\rho}{2} \|x - (x_k)_+\|^2,$$

where $(x_k)_+$ is the projection of x_k onto $\mathbb{R}^n_+$ and its stationary equation

$$v + \rho(x - (x_k)_+) + A^T \lambda = 0. \tag{4.2}$$

Assuming A has full row rank, solving (4.2) with respect to λ leads the update

$$x_{k+1} = (I - A^- A) \left((x_k)_+ - \frac{1}{\rho} v \right) + A^- b, \tag{4.3}$$

where $A^- = A^T (AA^T)^{-1}$ is the pseudo-inverse of A. Secondly, we fold the nonnegativity constraints into the domain of the loss. Let p_k be the projection of x_k onto the affine constraints set, then the surrogate function becomes

$$g(x|x_k) = v^T x + \frac{\rho}{2} \|x - p_k\|^2.$$

The minimizer of $g(x|x_k)$ can be computed elementwisely and nonnegativity constraints impose the update

$$x_{k+1,j} = \left(p_{k,j} - \frac{v_j}{\rho}, 0 \right)_+ \tag{4.4}$$

for $j = 1, \cdots, n$. Lastly, we can minimize

$$v^T x + \frac{\rho}{2} \text{dist}(Ax, b)^2 + \frac{\rho}{2} \text{dist}(x, \mathbb{R}^n_+)^2,$$

where $\mathbb{R}^n_+ = \{u \in \mathbb{R}^n : u_i \geq 0 \text{ for } i = 1, \cdots, n\}$. The update is directly induced by minimizing the distance majorized surrogate function:

$$x_{k+1} = (I + A^T A)^{-1} \left(A^T b + (x_k)_+ - \frac{1}{\rho} v \right). \tag{4.5}$$

To apply the primal dual hybrid gradient method, problem (4.1) can be rephrased as an unconstrained problem

$$\min_{x\in\mathbb{R}^n} v^T x + \delta_{\mathbb{R}^n_+}(x) + \delta_b(Ax),$$

where δ_C is again the indicator function of C. Using Algorithm 4 and its dual, we induce two iterates:

$$\begin{aligned} x_{k+1} &= \operatorname{prox}_{\tau g}(x_k - \tau(v + A^T y_k)) \\ \tilde{u}_{k+1} &= 2Ax_{k+1} - Ax_k \\ y_{k+1} &= \operatorname{prox}_{\sigma h^\star}(y_k + \sigma\tilde{u}_{k+1}) \end{aligned} \tag{4.6}$$

and

$$\begin{aligned} y_{k+1} &= \operatorname{prox}_{\sigma h^\star}(y_k + \sigma Ax_k) \\ \tilde{v}_{k+1} &= 2A^T y_{k+1} - A^T y_k \\ x_{k+1} &= \operatorname{prox}_{\tau g}(x_k - \tau(v + \tilde{v}_{k+1})), \end{aligned} \tag{4.7}$$

where $g(x) = \delta_{\mathbb{R}^n_+}(x)$ and $h(Ax) = \delta_{\{b\}}(Ax)$.

All projection maps and proximity operators have closed form representation. Projection onto the point b is obvious. Projection x onto the nonnegative orthant folds all negative components to zero, while the positive elements remain intact. One can attack the projection of x_k onto affine constraints by minimizing the Lagrangian

$$\frac{1}{2}\|p_k - x_k\|^2 + \lambda^T(Ap_k - b),$$

which leads to the solution

$$p_k = (I - A^- A)x_k + A^- b.$$

One can induce the proximity operator of the convex conjugate of an indicator functions by Moreau's decomposition (3.2).

Dimensions		Optima						
m	n	PDA1	PDA2	PDA3	PDHG1	PDHG2	Gurobi	Mosek
2	4	0.3364	0.3364	0.3364	0.3364	0.3364	0.3364	0.3364
4	8	0.5502	0.5502	0.5502	0.5503	0.5503	0.5502	0.5502
8	16	2.2750	2.2750	2.2750	2.2750	2.2750	2.2750	2.2750
16	32	2.3663	2.3663	2.3663	2.3662	2.3662	2.3663	2.3663
32	64	4.9383	4.9383	4.9383	4.9383	4.9383	4.9383	4.9383
64	128	15.7129	15.7129	15.7129	15.7130	15.7130	15.7129	15.7129
128	256	30.9085	30.9085	30.9085	30.9085	30.9085	30.9085	30.9085
256	512	65.8579	65.8578	65.8579	65.8577	65.8577	65.8576	65.8576

Dimensions		CPU Times						
m	n	PDA1	PDA2	PDA3	PDHG1	PDHG2	Gurobi	Mosek
2	4	0.0367	0.0338	0.0225	0.0001	0.0001	0.0064	0.0014
4	8	0.0319	0.0195	0.0181	0.0028	0.0027	0.0052	0.0017
8	16	0.0433	0.0379	0.0414	0.0030	0.0027	0.0054	0.0018
16	32	0.0610	0.0578	0.0700	0.0466	0.0364	0.0052	0.0020
32	64	0.1170	0.0991	0.1406	0.0180	0.0191	0.0069	0.0034
64	128	0.3391	0.4196	0.4598	0.1008	0.0800	0.0125	0.0152
128	256	0.4386	0.6243	0.6607	3.0376	3.1415	0.0527	0.0413
256	512	0.9082	1.5350	1.6671	2.1436	1.7653	0.2224	0.2511

Dimensions		Feasibility						
m	n	PDA1	PDA2	PDA3	PDHG1	PDHG2	Gurobi	Mosek
2	4	0.00000098	0.00000092	0.00000097	0.00000000	0.00000000	0.00000000	0.00000000
4	8	0.00000099	0.00000099	0.00000100	0.00000000	0.00000001	0.00000000	0.00000000
8	16	0.00000100	0.00000100	0.00000099	0.00000001	0.00000001	0.00000000	0.00000000
16	32	0.00000100	0.00000099	0.00000100	0.00000002	0.00000002	0.00000000	0.00000000
32	64	0.00000096	0.00000098	0.00000096	0.00000003	0.00000003	0.00000000	0.00000000
64	128	0.00000097	0.00000098	0.00000097	0.00000008	0.00000007	0.00000000	0.00000000
128	256	0.00000099	0.00000100	0.00000099	0.00000001	0.00000003	0.00000000	0.00000000
256	512	0.00000099	0.00000100	0.00000099	0.00000030	0.00000029	0.00000000	0.00000000

Table 4.1 Optima, CPU times in seconds, and feasibility for linear programming. Here m is the number of constraints and n is the number of variables. PDA1, PDA2, and PDA3 are results of the update (4.3), (4.4), and (4.5). PDHG1 and PDHG2 are from (4.6) and (4.7) respectively.

Table 4.1 compares the performance of proximal distance algorithm, primal dual hybrid gradient, and two commercial solvers, Gurobi [17] and Mosek [29]. In lower dimensions, the primal dual hybrid gradient methods outperform other methods. As dimensions increase, the performance of Gurobi and Mosek dominates others, especially primal dual hybrid gradient suffers slow convergence. For the proximal distance algorithm, folding affine constraints into domain of the loss seems to be the best tactic. Additionally, optima from the proximal distance iterates becomes inaccurate in large dimension, which suggests one needs to slow down the update of penalty parameter ρ_k as the size of problem grows for better accuracy.

4.2 Constrained Least Squares

Constrained least squares entails minimizing the linear least squares $\|Ax - y\|^2$ subject to $x \in C$. For example, the constraints C given as $\{u : Du = c\}$ yields the equality constrained least squares and the constraints that all elements of x should be nonnegative leads the nonnegative least squares. Under certain circumstances, reframing a constrained quadratic programming into constrained least squares programming has benefits [4]. For simulation, we used the probability-simplex constraint. That is, our problem has form:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} \frac{1}{2}\|y - Ax\|^2 \\ &\text{subject to } x \in S, \end{aligned} \tag{4.8}$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $S = \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i = 1, 0 \leq x_i \leq 1 \text{ for } i = 1, \cdots, n\}$.

Applying distance majorization to penalized objective, we can get the surrogate:

$$g(x|x_k) = \frac{1}{2}\|y - Ax\|^2 + \frac{\rho}{2}\|x - P_S(x_k)\|^2.$$

The proximal distance update

$$x_{k+1} = (A^TA + \rho I)^{-1}(A^Ty + \rho P_S(x_k)) \tag{4.9}$$

is induced by minimizing $g(x|x_k)$. For large scale and sparse problem, one may reform the surrogate

$$g(x|x_k) = \frac{1}{2}\left\|\begin{pmatrix} y \\ \sqrt{\rho}P_S(x_k)\end{pmatrix} - \begin{pmatrix} A \\ \sqrt{\rho}I\end{pmatrix}x\right\|^2 =: \frac{1}{2}\|\tilde{y} - \tilde{A}x\|^2,$$

which can be solved by fast stable conjugate gradient solvers.

The primal dual hybrid gradient method provides two algorithms for the problem. If we convert problem (4.8) into unconstrained problem

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}\|y - Ax\|^2 + \delta_S(x),$$

Algorithm LV and CV can be employed. Otherwise, the probability simplex set can be represented as intersection of

$$C_1 = \{u \in \mathbb{R}^n : 0 \le u_i \le 1 \text{ for } i = 1, \cdots, n\} \text{ and } C_2 = \left\{u \in \mathbb{R}^n : \sum_{i=1}^n u_i = 1\right\}.$$

Then, we can formulate the problem as

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}\|y - Ax\|^2 + \delta_{C_1}(x) + \delta_{\{1\}}(\mathbf{1}^Tx),$$

where $\mathbf{1} \in \mathbb{R}^n$ be a vector that all elements are 1, for which Algorithm 4 can be applied.

The projection map onto a vector $\mathbf{1}$ is trivial. To get projection onto C_1, it is enough to truncate all values into $[0, 1]$. Calculating the projection of y

Dimensions		Optima					
n	p	PDA	LV	CV	PDHG1	PDHG2	Gurobi
16	8	3.4919	3.4919	3.4919	3.4919	3.4919	3.4919
32	16	17.1458	17.1458	17.1458	17.1458	17.1458	17.1458
64	32	26.2897	26.2897	26.2897	26.2897	26.2897	26.2897
128	64	55.2645	55.2645	55.2645	55.2645	55.2645	55.2645
256	128	112.0550	112.0550	112.0550	112.0550	112.0550	112.0550
512	256	202.4702	202.4703	202.4703	202.4703	202.4703	202.4703

Dimensions		CPU Times					
n	p	PDA	LV	CV	PDHG1	PDHG2	Gurobi
16	8	0.0023	0.0002	0.0006	0.0001	0.0001	0.0049
32	16	0.0027	0.0003	0.0009	0.0002	0.0001	0.0052
64	32	0.0061	0.0004	0.0020	0.0058	0.0003	0.0058
128	64	0.0136	0.0012	0.0035	0.0036	0.0009	0.0082
256	128	0.0467	0.0046	0.0102	0.0111	0.0143	0.0206
512	256	0.1280	0.0145	0.0327	0.0362	0.0353	0.1569

Dimensions		Feasibility					
n	p	PDA	LV	CV	PDHG1	PDHG2	Gurobi
16	8	0.00000008	0.00000000	0.00000013	0.00000031	0.00000007	0.00000000
32	16	0.00000010	0.00000000	0.00000003	0.00000008	0.00000004	0.00000000
64	32	0.00000008	0.00000000	0.00000002	0.00000001	0.00000000	0.00000000
128	64	0.00000007	0.00000000	0.00000001	0.00000001	0.00000001	0.00000000
256	128	0.00000007	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
512	256	0.00000009	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000

Table 4.2 Optima, CPU times in seconds, and feasibility for simplex-constrained least squares. Here $\boldsymbol{A} \in \mathbb{R}^{n\times p}$. Result of (4.9) is PDA. LV and CV are from Algorithm LV and Algorithm CV respectively. PDHG1 is the result of Algorithm 4 and PDHG2 is that of its dual algorithm.

onto S is little tricky. Let z_i be ith largest element of y. From $j = 1$ to n, set $\lambda = \frac{1}{j}(\sum_{i=1}^{j} - 1)$. If $z_j > \lambda$ and $z_{j+1} \leq \lambda$, then $(P_S(y))_i := (y_i - \lambda)_+$ [25].

Table 4.2 compares the proximal distance algorithm, four primal dual hybrid gradient methods, and Gurobi solver. In any cases, Algorithm LV surpasses all other methods and the rest of primal dual hybrid gradient methods follow. Though the proximal distance algorithm is slow, it is still competitive with the commercial software Gurobi solver.

4.3 Closest Kinship Matrix

Kinship is an important topics in many fields of biology [19]. If two individuals have the same ancestor, they would share some common genes, so that the coefficient of kinship can be estimated by the fraction of sharing single nucleotide polymorphisms (SNP). For given pedigree, we can build a symmetric kinship matrix Z, which should have three properties: First, all entries of Z are nonnegative. Second, Z should be positive semidefinite. Lastly, diagonal entries of Z are all 1 unless inbreeding occurs. However, there is no guarantee that the empirical kinship matrix achieves such properties. Dropping the possibility of inbreeding, we can approximate the qualifying matrix from Z. In this regard, the problem can be formulated as follows:

$$\min_{X \in \mathbb{S}^n} \frac{1}{2}\|X - Z\|_F^2 \tag{4.10}$$
$$\text{subject to } X \in C_1 \cap C_2,$$

where $\mathbb{S}^n$ denotes the set of symmetric matrices, $C_1 = \{X \in \mathbb{R}^{n\times n} : X$ is positive semi-definite$\}$, and $C_2 = \{X \in \mathbb{R}^{n\times n} : X$ has only nonnegative entries and diagonal entries of X are $1\}$.

Similarly to LP, there are two ways to solve problem (4.10) using the proximal distance algorithm. One is to apply the proximal distance algorithm directly, which generates the surrogate

$$\begin{aligned} g(X|X_k) &= \frac{1}{2}\|X - Z\|_F^2 + \frac{\rho}{4}\|X - P_{C_1}(X_k)\|_F^2 + \frac{\rho}{4}\|X - P_{C_2}(X_k)\|_F^2 \\ &= \frac{1+\rho}{2}\left\|X - \frac{1}{1+\rho}Z - \frac{\rho}{2(1+\rho)}(P_{C_1}(X_k) + P_{C_2}(X_k))\right\|_F^2 + c_k, \end{aligned}$$

where c_k is an irrelevant constant. Hence, the update becomes

$$X_{k+1} = \frac{1}{1+\rho}Z + \frac{\rho}{2(1+\rho)}(P_{C_1}(X_k) + P_{C_2}(X_k)). \tag{4.11}$$

The other is attacking the problem by folding the positive semidefinite constraint into the domain of loss function. That is, the surrogate is

$$\begin{aligned} g(X|X_k) &= \frac{1}{2}\|X - Z\|_F^2 + \frac{\rho}{2}\|X - P_{C_2}(X_k)\|_F^2 \\ &= \frac{1+\rho}{2}\left\|X - \frac{1}{1+\rho}Z - \frac{\rho}{1+\rho}P_{C_2}(X_k)\right\|_F^2 + c_k \end{aligned}$$

and the update reduces to

$$X_{k+1} = P_{C_1}\left(\frac{1}{1+\rho}Z + \frac{\rho}{1+\rho}P_{C_2}(X_k)\right). \tag{4.12}$$

The unconstrained conversion of problem (4.10) is

$$\min_{X\in\mathbb{R}^n} \frac{1}{2}\|X - Z\|_F^2 + \delta_{C_1}(X) + \delta_{C_2}(X).$$

Let $g(X) = \delta_{C_1}(X)$ and $h(X) = \delta_{C_2}(X)$. We can directly apply Algorithm 4:

$$\begin{aligned} X_{k+1} &= \operatorname{prox}_{\tau g}(X_k - \tau(X_k - Z + Y_k)) \qquad (4.13) \\ &= P_{C_1}(X_k - \tau(X_k - Z + Y_k)) \\ u_{k+1} &= 2X_{k+1} - X_k \\ Y_{k+1} &= \operatorname{prox}_{\sigma h^\star}(Y_k + \sigma u_{k+1}), \\ &= Y_k + \sigma\left\{u_{k+1} - P_{C_2}\left(\frac{1}{\sigma}Y_k + u_{k+1}\right)\right\}. \end{aligned}$$

Additionally, switching $g(X) = \delta_{C_2}(X)$ and $h(X) = \delta_{C_1}(X)$ gives another updates.

Let X be a symmetric matrix and its eigenvalue decomposition $X = UDU^T = \sum_{i=1}^n \lambda_i u_i u_i^T$ where λ_i are eigenvalues and u_i are corresponding eigenvectors. It is well known that the projection of X onto the positive semidefinite cone C_1 with respect to the Euclidean or Frobenius norm is $P_{C_1}(X) = \sum_{i=1}^n (\lambda_i)_+ u_i u_i^T$ [7]. The projection map onto C_2 is much obvious: Folding negative entries to 0 and setting diagonal elements to 1.

Table 4.3 compares the proximal distance algorithm, the primal dual hybrid gradient methods, and Dykstra's algorithm [8]. Dykstra's algorithm tracks a primary sequence X_n and its companion sequence V_n via alternating projections. As a result, X_n converges to the projection of Z onto $C_1 \cap C_2$ and V_n converges to the Langrange multiplier [25]. For this problem, Dykstra's algorithm was the fastest and the primal dual hybrid gradients and the proximal distance algorithm follow. It is remarkable that the directly proximal distance algorithm is not only slow but also can be inaccurate. Switching $g(X)$ and $h(X)$ appears not to affect convergence of primal dual hybrid gradient methods.

4.4 Projection onto a Second-Order Cone Constraint

A second-order cone programming (SOCP) solves:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} f^T x \qquad (4.14)\\ \text{subject to } &\|A_i x + b_i\|_2 \le c_i^T x + d_i \;\; i = 1, \cdots, m \\ &Fx = g, \end{aligned}$$

Dimensions	Optima				
n	PDA1	PDA2	PDHG1	PDHG2	Dykstra
4	2.8576	2.8576	2.8576	2.8576	2.8576
8	18.7776	18.7776	18.7776	18.7776	18.7776
16	45.1218	45.1217	45.1218	45.1218	45.1218
32	169.7005	169.7005	169.7005	169.7005	169.7005
64	838.4236	838.4234	838.4234	838.4234	838.4234
128	3279.2287	3279.2276	3279.2276	3279.2276	3279.2276
256	14043.4731	14043.4608	14043.4605	14043.4605	14043.4605

Dimensions	CPU times				
n	PDA1	PDA2	PDHG1	PDHG2	Dykstra
4	0.0269	0.0239	0.0018	0.0054	0.0006
8	0.0592	0.0478	0.0021	0.0026	0.0006
16	0.1586	0.1486	0.0161	0.0114	0.0071
32	0.5988	0.5461	0.1641	0.1892	0.0897
64	3.3199	2.8882	1.3750	1.3809	0.7556
128	21.2916	21.0350	8.1270	8.3357	4.0588
256	129.4685	123.3340	76.9175	73.8031	39.0772

Dimensions	Feasibility				
n	PDA1	PDA2	PDHG1	PDHG2	Dykstra
4	0.00000088	0.00000038	0.00000000	0.00000000	0.00000000
8	0.00000043	0.00000043	0.00000000	0.00000000	0.00000000
16	0.00000086	0.00000070	0.00000122	0.00000000	0.00000000
32	0.00000098	0.00000098	0.00000126	0.00000000	0.00000000
64	0.00000077	0.00000065	0.00000127	0.00000000	0.00000000
128	0.00000082	0.00000089	0.00000122	0.00000000	0.00000000
256	0.00000086	0.00000090	0.00000123	0.00000000	0.00000000

Table 4.3 Optima, CPU times in seconds, and feasibility for the closest kinship matrix problem. Here the kinship matrix is $n \times n$. PDA1 and PDA2 are from the update (4.11) and (4.12) respectively. The result of (4.13) is PDHG1 and PDHG2 is the output of algorithm that indicator functions in the problem are switched.

where $x \in \mathbb{R}^n, A_i \in \mathbb{R}^{n_i \times n}$, and $F \in \mathbb{R}^{p \times n}$ [7]. Problem (4.14) contains second-order cone constraints, which can be written in simple form:

$$x \in C := \{u \in \mathbb{R}^n : \|Au + b\|_2 \leq c^T u + d\},$$

where $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, c \in \mathbb{R}^n$, and $d \in \mathbb{R}$. In here, we consider the projection of x onto C.

The key idea of the proximal distance solution is the variable splitting. Let $w = Au + b$ and $r = c^T u + d$. Defining $y = (u, w, r)$, $L = \{y \in \mathbb{R}^{m+n+1} : \|w\| \leq r\}$, and $M = \{y \in \mathbb{R}^{m+n+1} : w = Au + b \text{ and } r = c^T u + d\}$, the problem can be formulated as

$$\begin{aligned} \min_{u \in \mathbb{R}^n} \frac{1}{2}\|x - u\|^2 \\ \text{subject to } y \in L \cap M. \end{aligned} \tag{4.15}$$

It is rather easy to get projection onto the second-order cone constraint set L, thus folding constraint set M into the domain of loss could be helpful. Let

$$\begin{pmatrix} \tilde{w}_k \\ \tilde{r}_k \end{pmatrix} = P_L \begin{pmatrix} w_k \\ r_k \end{pmatrix},$$

then the surrogate becomes

$$g(y|y_k) = \frac{1}{2}\|x - u\|^2 + \frac{\rho}{2}\left\| \begin{pmatrix} w_k - \tilde{w}_k \\ r_k - \tilde{r}_k \end{pmatrix} \right\|^2 + \lambda^T(Au + b - w) + \theta(c^T u + d - r).$$

The stationary equation is

$$\begin{aligned} u - x + A^T\lambda + \theta c &= 0 \\ \rho(w - \tilde{w}_k) &= \lambda \\ \rho(r - \tilde{r}_k) &= \theta \end{aligned}$$

and thus the update is

$$
\begin{aligned}
u_{k+1} &= \left(A^T A + \frac{1}{\rho} I + cc^T\right)^{-1} \left(A^T(\tilde{w}_k - b) + (\tilde{r}_k - d)c + \frac{1}{\rho} x\right) \qquad (4.16)\\
w_{k+1} &= Au_{k+1} + b\\
r_{k+1} &= c^T u_{k+1} + d.
\end{aligned}
$$

To solve problem (4.15) by the primal dual hybrid gradient, we combine w and r as

$$
\begin{pmatrix} w \\ r \end{pmatrix} = \begin{pmatrix} A \\ c^T \end{pmatrix} u + \begin{pmatrix} b \\ d \end{pmatrix} =: \tilde{A}u + \tilde{b}.
$$

Then, unconstrained formulation of (4.14) is

$$
\min_{u \in \mathbb{R}^n} \frac{1}{2}\|x - u\|^2 + \delta_L(\tilde{A}u + \tilde{b}),
$$

which is equivalent to solving

$$
\min_{u \in \mathbb{R}^n} \frac{1}{2}\|x - u\|^2 + \delta_{L-\tilde{b}}(\tilde{A}u).
$$

Therefore, Algorithm LV and CV can be applied. Otherwise, one embraces the variable splitting and reforms (4.15) as

$$
\min_{x \in \mathbb{R}^n} \frac{1}{2} \left\| \begin{pmatrix} I & O & O \\ O & O & O \\ O & O & 0 \end{pmatrix} y - \begin{pmatrix} x \\ 0 \\ 0 \end{pmatrix} \right\|^2 + \delta_M(y) + \delta_L \left(\begin{pmatrix} O & O & O \\ O & I & O \\ O & O & 1 \end{pmatrix} y \right),
$$

so that uses Algorithm 4.

It is easy to check that the projection of (w, r) onto the second-order cone

L can be expressed as

$$P_L \begin{pmatrix} w \\ r \end{pmatrix} = \begin{cases} \begin{pmatrix} w \\ r \end{pmatrix}, & \text{if } \|w\| \le r \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, & \text{if} \|w\| \le -r \\ \dfrac{\|w\| + r}{2\|w\|} \begin{pmatrix} w \\ \|w\| \end{pmatrix}, & \text{otherwise.} \end{cases}$$

To get the projection map $P_M(y)$, we rewrite the constraint M as

$$M = \left\{ \begin{pmatrix} u \\ \tilde{z} \end{pmatrix} \in \mathbb{R}^{m+n+1} : \tilde{z} = \tilde{A}u + \tilde{b} \right\},$$

where $\tilde{z} = (w, r)$. Getting the projection of (p, q) onto M is equivalent to solving

$$P_M \begin{pmatrix} p \\ q \end{pmatrix} = \underset{u \in \mathbb{R}^n, \tilde{z} \in \mathbb{R}^{m+1}}{\operatorname{argmin}} \left\{ \frac{1}{2}\|p - u\|^2 + \frac{1}{2}\|q - \tilde{z}\|^2 + \lambda^T(\tilde{A}u + b - \tilde{z}) \right\}.$$

The stationary equations are

$$\begin{aligned} 0 &= u - p + \tilde{A}^T \lambda \\ 0 &= \tilde{z} - q - \lambda, \end{aligned}$$

thus the projection is

$$P_M \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} p - \tilde{A}^T(\tilde{A}\tilde{A}^T + I)^{-1}(\tilde{A}p - q + \tilde{b}) \\ q + (\tilde{A}\tilde{A}^T + I)^{-1}(\tilde{A}p - q + \tilde{b}) \end{pmatrix}.$$

Table 4.4 compares performance of the proximal distance algorithm, primal dual hybrid gradient methods, and two solvers, SCS [31] and Gurobi. It is

Dimensions		Optima					
m	n	PDA	LV	CV	PDHG	SCS	Gurobi
2	4	0.4775	0.4775	0.4775	0.4775	0.4775	0.4775
4	8	1.0617	1.0617	1.0617	1.0617	1.0617	1.0617
8	16	0.5399	0.5399	0.5399	0.5399	0.5399	0.5399
16	32	1.8113	1.8113	1.8113	1.8113	1.8113	1.8113
32	64	0.6211	0.6211	0.6211	0.6211	0.6211	0.6211
64	128	2.6805	2.6805	2.6805	2.6805	2.6805	2.6805
128	256	8.7678	8.7678	8.7678	8.7677	8.7678	8.7678
256	512	14.0124	14.0124	14.0124	14.0123	14.0124	14.0124

Dimensions		CPU Times					
m	n	PDA	LV	CV	PDHG	SCS	Gurobi
2	4	0.0188	0.0002	0.0002	0.0074	0.0008	0.0055
4	8	0.0282	0.0006	0.0006	0.0486	0.0010	0.0056
8	16	0.0084	0.0005	0.0005	0.0127	0.0018	0.0058
16	32	0.0105	0.0010	0.0011	0.0072	0.0028	0.0060
32	64	0.0221	0.0023	0.0081	0.0080	0.0171	0.0110
64	128	0.0584	0.0044	0.0156	0.0217	0.1758	0.0215
128	256	0.1397	0.0404	0.0996	0.0682	0.8971	0.0781
256	512	0.3849	0.1992	0.6323	0.1886	12.0251	0.3220

Table 4.4 Optima and CPU times in seconds for the second-order cone projection. Here m is the number of constraints and n is the number of variables. PDA is the result of the update (4.16). LV and CV are the output of Algorithm LV and Algorithm CV respectively. PDHG is from Algorithm 4.

clear that Algorithm LV and CV outperform in lower dimensions. While Algorithm LV seems to be scalable, Algorithm CV falls behind other algorithms except SCS. Algorithm 4 (in here denoted by PDHG) follows next in moderate dimensions. In spite of the poor performance in lower dimensions, the proximal distance algorithm is competitive with Gurobi. SCS solver becomes worse than any other methods.

4.5 Low Rank Matrix Completion

Let $M \in \mathbb{R}^{n_1 \times n_2}$ be a data matrix with missing data. There are various situations that filling missing entries of M from observed values. For example, in the well known Netflix problem [5], the data matrix consists of ratings that users give to movies. Observed values are the actual ratings that users gave and missing entries represent the ratings expected to be given by users. Therefore, one can build a recommendation system via filling missing parts of the data. To impute missing in data, we approach to the problem with a simple idea: The data is derived from a few factors. In the Netflix problem, users have a few unknown latent factors of preference and ratings are linear combination of them. With our principle, we can formulate the problem as follows:

$$\begin{aligned} \min_{X \in \mathbb{R}^{n_1 \times n_2}} & f(X) = \operatorname{rank}(X) \\ \text{subject to } & X_{ij} = M_{ij},\ i,j \in \Omega(M), \end{aligned} \tag{4.17}$$

where $\Omega(M)$ is a set of observed indices of M. Unfortunately, problem (4.17) is non-convex and NP-hard [10]. Candés and Recht made a detour of the problem by relaxing the rank function by the nuclear norm $\|X\|_* = \sum \sigma_i(X)$ where $\sigma_i(X)$ denotes ith singular value of X. With the nuclear norm, the problem becomes

$$\begin{gathered}\min_{X\in\mathbb{R}^{n_1\times n_2}} f(X)=\|X\|_* \\ \text{subject to } X_{ij}=M_{ij},\ i,j\in\Omega(M).\end{gathered}\tag{4.18}$$

Candés and Recht also give a condition that the matrix recovered by problem (4.18) is formally equivalent to that of (4.17). Therefore, we build solvers for (4.18) with the proximal distance algorithm and the primal dual hybrid gradient.

Let $C(M)=\{Y\in\mathbb{R}^{n_1\times n_2}: Y_{ij}=M_{ij} \text{ for } i,j\in\Omega(M)\}$, then constraints of the problem become $X\in C(M)$. Since $C(M)$ is a convex set, we can directly induce the proximal distance iterates:

$$X_{k+1}=\operatorname{prox}_{\rho^{-1}f}(P_{C(M)}(X_k))\tag{4.19}$$

and the primal dual hybrid gradient method from Algorithm 2:

$$\begin{aligned}Y_{k+1}&=\operatorname{prox}_{\sigma h^\star}(Y_k+\sigma\bar{X}_k)\\ X_{k+1}&=\operatorname{prox}_{\tau f}(X_k-\tau Y_{k+1})\\ \bar{X}_{k+1}&=2X_{k+1}-X_k,\end{aligned}\tag{4.20}$$

where $h(X)=\delta_{C(M)}(X)$.

We make the projection map of X onto $C(M)$ by replacing the elements of X in $\Omega(M)$ by those of M and leaving others intact. The proximity operator of the convex conjugate of $\delta_{C(M)}$ is computed by Moreau's decomposition (3.2). Let the rank of X be r and the singular value decomposition of X be $X=U\Sigma V^T$, where $U\in\mathbb{R}^{n_1\times r}$ and $V\in\mathbb{R}^{n_2\times r}$ are orthonormal columns and $\Sigma\in\mathbb{R}^{r\times r}$ is diagonal. Cai, Candés, and Shen defined the soft-thresholding operator of X for $\tau>0$ as

$$\mathcal{D}_\tau(X)=U\mathcal{D}_\tau(\Sigma)V^T, \mathcal{D}_\tau(\Sigma)=\operatorname{diag}((\sigma(X)-\tau)_+)$$

No.	size ($n \times n$)	rank (r)	m/d_r	m/n^2
1	500 × 500	10	5	0.20
2		50	3	0.57
3		100	2	0.72
4	1,000 × 1,000	10	6	0.12
5		50	4	0.39
6		100	3	0.57
7	2,000 × 2,000	10	6	0.06
8		50	4	0.20
9		100	3	0.29

Table 4.5 The properties of generated M.

and showed that the soft-thresholding operator is the proximity operator associated with the nuclear norm [9].

We picked the singular value thresholding algorithm as a competitor. The singular value thresholding algorithm is a type of Lagrange multiplier algorithm known as Uzawa's algorithm [9]. Since the singular value thresholding algorithm contains soft-thresholding operator step and projection onto $C(M)$, we expect to compare the performance of algorithms in similar conditions.

For the simulation, we generated two $n \times r$ matrices M_L and M_R having independent and identically distributed Gaussian entries and set $M = M_L M_R^T$, so that M becomes $n \times n$ matrix with rank r. The set of observed entries Ω is sampled uniformly at random among all sets of cardinality m. Table 4.5 shows the properties of the generated M. Note that an $n \times n$ matrix of rank r has $d_r := r(2n - r)$ degrees of freedom. Then, m/d_r is the ratio between the number of observations and degrees of freedom of M, which can be called the

	PDA			PDHG			SVT		
No.	Times	Iter.	Rel. err.	Times	Iter.	Rel. err.	Times	Iter.	Rel. err.
1	20.9	177	0.00013	10.3	83	0.00016	14.0	135	0.00018
2	17.2	114	0.00012	6.7	47	0.00020	14.2	135	0.00017
3	18.6	91	0.00017	8.6	50	0.00016	28.2	207	0.00022
4	143.1	266	0.00012	48.0	104	0.00013	43.6	121	0.00018
5	145.4	143	0.00012	42.4	58	0.00014	71.4	114	0.00016
6	176.2	133	0.00012	48.0	47	0.00018	99.1	130	0.00017
7	1848.2	498	0.00013	659.8	198	0.00014	333.8	122	0.00017
8	2470.5	412	0.00013	391.3	88	0.00016	407.7	123	0.00017
9	4216.8	384	0.00014	540.4	81	0.00017	815.6	167	0.00021

Table 4.6 CPU times in seconds, the number of iteration, and relative error for low rank matrix completion. Here No. is the experiment number. PDA and PDHG are results of the updates (4.19) and (4.20) respectively. SVT is from the singular value thresholding algorithm.

oversampling ratio. m/n^2 is the observed ratio.

Table 4.6 compares the proximal distance algorithm, the primal dual hybrid gradient, and the singular value thresholding algorithm. We report the running times, the iterations, and the relative error of the reconstruction

$$\text{Rel. err.} = \frac{\|X^{opt} - M\|_F}{\|M\|_F}$$

when the algorithms converge. We first remark that the iteration numbers of the singular value thresholding algorithm are not varying much in spite of the setup changes. On the other hand, the iteration numbers of the proximal distance algorithm and the primal dual hybrid gradient increase as the size of input grows. The primal dual hybrid gradient dominates others except experiment number 7 and the proximal distance algorithm was worst pick in the most of experiments.

4.6 Regression with Partially Ordered Coefficients

In regression analysis with categorical variables, one recasts such variables as dummy variables to see influence of each group. If the effect of groups is known to be ordered, one may expect that corresponding coefficients have clear ordered structures. However, several factors such as noise, multicolinearity, and sampling bias can cause disarrangement in the result. To force coefficients be ordered, one may give order constraints on them. For example, suppose that we have a categorical variable with q factors. Then, we can generate $q-1$ dummy variables $\beta_1, \beta_2, \cdots, \beta_{q-1}$. If such variable is known to be positively correlated with response, the regression problem becomes solving

$$\min_{\beta \in \mathbb{R}^p} f(\beta; y, X)$$
$$\text{subject to } 0 \leq \beta_1 \leq \cdots \leq \beta_{q-1}$$

with a proper loss f. Since the order constraints can be represented as

$$D_{\text{partial}} \beta_{\text{partial}} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & -1 & 1 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \\ \beta_{q-2} \\ \beta_{q-1} \end{pmatrix} \in \mathbb{R}^{q-1},$$

our algorithms can be applied to solve this problem. For simulation, we used the 'smbsimdf1' data from the R package 'smbinning'. The data consist of 2,500 rows and 21 predictors and a response of default. We dropped NAs out and recoded predictors to 45 dummies. Logistic regression is commonly used to predict default, thus our problem can be formulated as follows:

$$\min_{\beta\in\mathbb{R}^p} f(\beta; y, X) = \sum_{i=1}^{n}[\log(1+\exp(X_i^T\beta)) - y_i X_i^T\beta]$$
$$\text{subject to } D\beta \in \mathbb{R}_+^m$$

where $D \in \mathbb{R}^{m\times p}$ is given order constraints with respect to β.

Unfortunately, given $f(\beta)$ is not easy to get its proximity operator. Note that the gradient and upper bound of hessian are given as

$$\nabla f(\beta) = \sum_{i=1}^{n}\left(\frac{1}{\exp(-X_i^T\beta) - y_i}\right) X_i \tag{4.21}$$

and

$$\begin{aligned}\nabla^2 f(\beta) &= \sum_{i=1}^{n}\frac{\exp(X_i^T\beta)}{(1+\exp(X_i^T\beta))^2} X_i X_i^T \\ &\le \frac{1}{4}X^T X =: L\end{aligned} \tag{4.22}$$

since $\dfrac{x}{(1+x)^2} \le \dfrac{1}{4}$ for $x \in \mathbb{R}$. Our objective becomes

$$h_\rho(\beta) = f(\beta; y, X) + \frac{\rho}{2}\mathrm{dist}(D\beta, \mathbb{R}_+^m)^2$$

and our surrogate can be generated as

$$g(\beta|\beta_k) = f(\beta_k) + \nabla f(\beta_k)^T(\beta - \beta_k) + \frac{L}{2}\|\beta-\beta_k\|^2 + \frac{\rho}{2}\|D\beta - (D\beta_k)_+\|.$$

Therefore, the proximal distance update is

$$\begin{aligned}\beta_{k+1} &= \operatorname*{argmin}_{\beta\in\mathbb{R}^p} g(\beta|\beta_k) \\ &= (\rho D^T D + LI)^{-1}(\rho D^T(D\beta_k)_+ + L\beta_k - \nabla f(\beta_k)).\end{aligned} \tag{4.23}$$

Since (4.21) and (4.22) exist, the solution induced by Algorithm LV is

$$\begin{aligned}
\tilde{\beta}_{k+1} &= \beta_k - \tau(\nabla f(\beta) + D^T \gamma_k) \\
\gamma_{k+1} &= (1-\rho_k)\gamma_k + \rho_k \mathrm{prox}_{\sigma h^\star}(\gamma_k + \sigma D \tilde{\beta}_{k+1}) \\
\beta_{k+1} &= (1-\rho_k)\beta_k + \rho_k(\tilde{\beta}_{k+1} - \tau D^T(\gamma_{k+1} - \gamma_k))
\end{aligned} \tag{4.24}$$

and that generated by Algorithm CV is

$$\begin{aligned}
\bar{\beta}_{k+1} &= \beta_k - \tau(\nabla f(\beta_k) + D^T \gamma_k) \\
\tilde{\beta}_{k+1} &= 2\bar{\beta}_{k+1} - \beta_k \\
\beta_{k+1} &= (1-\rho_k)\beta_k + \rho_k \bar{\beta}_{k+1} \\
\gamma_{k+1} &= (1-\rho_k)\gamma_k + \rho_k \mathrm{prox}_{\sigma h^\star}(\gamma_k + \sigma D \tilde{\beta}_{k+1}),
\end{aligned} \tag{4.25}$$

where $h(\alpha) = I_{\mathbb{R}^m_+}(\alpha)$.

Although logistic regression can be represented as a geometric programming (GP), Convex.jl and Mosek do not support GP. In here, we formulated our problem as an exponential cone programming and used the SCS solver. Table 4.7 compares the performance of the proximal distance algorithm, Algorithm LV and CV, and the SCS solver. The proximal distance algorithm dominates others in every six setup. The proximal distance algorithm and the primal dual hybrid gradient methods are all scalable, while the SCS solver works poorly as dimension increases.

Dimensions		Optima				CPU Times			
m	n	PDA	LV	CV	SCS	PDA	LV	CV	SCS
5	912	295.1474	295.1474	295.1474	295.1475	0.8463	1.1550	1.1072	6.8652
9	912	297.7589	297.7589	297.7589	297.7589	0.8862	1.1262	1.1033	8.4962
15	912	298.9476	298.9476	298.9476	298.9476	1.0905	1.0986	1.0798	7.6411
5	1824	613.9417	613.9417	613.9417	613.9418	1.6125	1.8661	1.8304	24.1962
9	1824	614.2980	614.2980	614.2980	614.2981	1.4637	1.6648	1.6743	33.6969
15	1824	615.7252	615.7252	615.7252	615.7252	1.4924	1.7293	1.6909	32.1526

Dimensions		Feasibility			
m	n	PDA	LV	CV	SCS
5	912	0.00000001	0.00000000	0.00000000	0.00006151
9	912	0.00000001	0.00000001	0.00000010	0.00000086
15	912	0.00000001	0.00000002	0.00000020	0.00002790
5	1824	0.00000001	0.00000000	0.00000000	0.00001237
9	1824	0.00000001	0.00000001	0.00000012	0.00001840
15	1824	0.00000001	0.00000001	0.00000014	0.00003827

Table 4.7 Optima, CPU times in seconds, and feasibility for regression with partially ordered coefficients. Here m is the number of constraints and n is the number of observations. PDA is the output of the update (4.23). LV and CV are the result of the updates (4.24) and (4.25) respectively.

Chapter 5

Discussion

Unlike the proximal gradient method and the alternating direction method of multipliers, both proximal distance algorithm and primal dual hybrid gradient methods only involve evaluation of the simple proximity operator of f or the gradient ∇f and the projection map onto constraints set. Despite this similarity, the classes of problems the two algorithms dealing with are different. The proximal distance algorithm handles a broad range of constraint optimization problems. The iterate can be derived even if the objective and constraints are nonconvex. Keys, Zhou, and Lange provide some numerical examples of nonconvex problem solved by the proximal distance algorithm [22]. On the other hand, the primal dual hybrid gradient methods only consider convex problems.

Even in convex cases, convergence analysis of the proximal distance algorithm is still an open problem. Existing theories only guarantee the algorithm achieves an optimal solution when a penalty constant ρ_k is fixed and the limit point of iterates gains feasibility as ρ_k increases. In practice, applying the framework of Nesterov's acceleration improves the rate of convergence and even the quality of the solution. Therefore, convergence analysis with flexible penalty constants and acceleration should be studied to enhance the algorithm. Instability of the solution depending on tuning schedule would be alleviated with further research. Meanwhile, many literature investigated that of primal dual hybrid gradient methods and their accelerations. The optimality and feasibility of solution generated from the primal dual hybrid gradient method are attained in most cases. Hence, tuning parameters of the primal dual hybrid gradient is rather easier than that of the proximal distance algorithm.

The proximal distance algorithm provides various solutions depending on given problems. If the problem contains more than one constraint, the algorithm may exploit weighted average of squared distance penalties of constraint sets. Our numerical experiments imply that combining constraints improves the rate of convergence, which coincides with the result in [22]. Folding constraints into domain of the loss is highly recommended if allowed. Similarly, using Algorithm 3 performs slightly better than Algorithm 4 in our examples. When comparing Algorithm LV with Algorithm CV, in general close in terms of speed, but most often Algorithm LV was better. That the region of convergence of Algorithm LV contains that of Algorithm CV may buttress the results [24]. Additionally, there is no difference between the performance of Algorithm 4 and its dual.

Except LP, the primal dual hybrid gradient methods usually outperform the proximal distance algorithm, even the commercial solvers in some cases. The proximal distance algorithm works well in LP and regression with partially ordered coefficient problem and is comparable to competitors in others. However, when the variable is a matrix, the proximal distance algorithm often have trouble with scalability while the primal dual hybrid gradient shows great performance even in large dimensions.

Appendix A

Programming details

In here, we provide the details of our implementation such as tuning schedules, selected parameters, and declaring convergence of algorithm. For the proximal distance algorithm, we need tuning schedule of penalty parameter: $\rho_{initial}$, ρ_{inc}, ρ_{max}, and k_ρ. We halt the iterate when the loss varies less than ϵ_{loss} and distance between the update and the constraints is less than ϵ_{dist}. For the primal dual hybrid gradient, one should determine the step sizes τ and σ. If not specified, the inertia parameter ρ_k is typically chosen to be 1. The algorithm is stopped if change of the objective is less than ϵ_{loss} and that of the variables is less than ϵ_{var}.

A.1 Linear Programming

- PDA1, PDA2 and PDA3

Dimensions		Parameters					
m	n	$\rho_{initial}$	ρ_{inc}	ρ_{max}	k_ρ	ϵ_{loss}	ϵ_{dist}
All		1.0	1.1	10^{30}	100	10^{-6}	10^{-6}

- PDHG1 and PDHG2

Dimensions		Parameters			
m	n	τ	σ	ϵ_{loss}	ϵ_{var}
All		$\sqrt{\frac{n}{m\|A\|_2}}$	$\sqrt{\frac{m}{n\|A\|_2}}$	10^{-5}	10^{-5}

A.2 Constrained Least Squares

- PDA

Dimensions		Parameters					
n	p	$\rho_{initial}$	ρ_{inc}	ρ_{max}	k_ρ	ϵ_{loss}	ϵ_{dist}
All		1.0	1.5	10^{30}	20	10^{-7}	10^{-7}

- LV

Dimensions		Parameters				
n	p	τ	σ	ρ	ϵ_{loss}	ϵ_{var}
All		$\frac{1.8}{\|A\|_2^2}$	$\frac{\|A\|_2^2}{2}$	0.99	10^{-6}	10^{-6}

- CV

Dimensions		Parameters				
n	p	τ	σ	ρ	ϵ_{loss}	ϵ_{var}
All		$\frac{1.8}{\|A\|_2^2}$	$\frac{\|A\|_2^2}{18}$	0.9	10^{-6}	10^{-6}

- PDHG1 and PDHG2

Dimensions		Parameters			
m	n	τ	σ	ϵ_{loss}	ϵ_{var}
All		$\frac{4}{\|A\|_2^2}$	$\frac{\|A\|_2^2}{8n}$	10^{-6}	10^{-6}

A.3 Closest Kinship Matrix

- PDA1 and PDA2

Dimensions	Parameters					
n	$\rho_{initial}$	ρ_{inc}	ρ_{max}	k_ρ	ϵ_{loss}	ϵ_{dist}
All	1.0	3.0	10^{30}	100	10^{-7}	10^{-6}

- PDHG1 and PDHG2

Dimensions	Parameters			
n	τ	σ	ϵ_{loss}	ϵ_{var}
All	1	0.5	10^{-6}	10^{-6}

A.4 Projection onto a Second-Order Cone Constraint

- PDA

Dimensions		Parameters					
m	n	$\rho_{initial}$	ρ_{inc}	ρ_{max}	k_ρ	ϵ_{loss}	ϵ_{dist}
All		1.0	1.5	10^{30}	20	10^{-6}	10^{-5}

- LV

Dimensions		Parameters			
m	n	τ	σ	ϵ_{loss}	ϵ_{var}
All		0.2	$\frac{99}{20\|\tilde{A}\|_2^2}$	10^{-8}	10^{-4}

- CV

Dimensions		Parameters			
m	n	τ	σ	ϵ_{loss}	ϵ_{var}
All		0.2	$\frac{81}{20\|\tilde{A}\|_2^2}$	10^{-8}	10^{-4}

- PDHG

Dimensions		Parameters			
m	n	τ	σ	ϵ_{loss}	ϵ_{var}
All		1.98	0.005	10^{-8}	10^{-4}

A.5 Low Rank Matrix Completion

- PDA

	Parameters					
No.	$\rho_{initial}$	ρ_{inc}	ρ_{max}	k_ρ	ϵ_{loss}	ϵ_{dist}
1	0.1	1.5	∞	10	10^{-4}	10^{-4}
2	0.1	1.7	∞	10	10^{-4}	10^{-4}
3	0.1	2.0	∞	10	10^{-4}	10^{-4}
4	0.1	1.3	∞	10	10^{-4}	10^{-4}
5	0.1	1.5	∞	10	10^{-4}	10^{-4}
6	0.1	1.5	∞	10	10^{-4}	10^{-4}
7	0.1	1.15	∞	10	10^{-4}	10^{-4}
8	0.1	1.15	∞	10	10^{-4}	10^{-4}
9	0.1	1.15	∞	10	10^{-4}	10^{-4}

- PDHG

	Parameters			
No.	τ	σ	ϵ_{loss}	ϵ_{var}
1	50	$\frac{2}{101}$	10^{-4}	10^{-4}
2	50	$\frac{2}{101}$	10^{-4}	10^{-4}
3	50	$\frac{2}{101}$	10^{-4}	10^{-4}
4	100	$\frac{1}{101}$	10^{-4}	10^{-4}
5	100	$\frac{1}{101}$	10^{-4}	10^{-4}
6	100	$\frac{1}{101}$	10^{-4}	10^{-4}
7	100	$\frac{1}{101}$	10^{-4}	10^{-4}
8	200	$\frac{1}{202}$	10^{-4}	10^{-4}
9	200	$\frac{1}{202}$	10^{-4}	10^{-4}

A.6 Regression with Partially Ordered Coefficients

- PDA

Dimension		Parameters					
m	n	$\rho_{initial}$	ρ_{inc}	ρ_{max}	k_ρ	ϵ_{loss}	ϵ_{dist}
All		0.1	1.05	∞	20	10^{-10}	10^{-8}

- LV

Dimension		Parameters				
m	n	τ	σ	ρ	ϵ_{loss}	ϵ_{var}
All		$\frac{36}{5\|A\|_2^2}$	$\frac{11\|A\|_2^2}{80\|C\|_2^2}$	0.99	10^{-6}	10^{-6}

- CV

Dimension		Parameters				
m	n	τ	σ	ρ	ϵ_{loss}	ϵ_{var}
All		$\frac{36}{5\|A\|_2^2}$	$\frac{11\|A\|_2^2}{800\|C\|_2^2}$	0.99	10^{-6}	10^{-6}

Bibliography

[1] Kenneth J. Arrow, Hirofumi Azawa, Leonid Hurwicz, and Hirofumi Uzawa. *Studies in Linear and Non-linear Programming.* Stanford University Press, 1958.

[2] Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces.* Springer, 2011.

[3] Edward J. Beltrami. *An Algorithmic Approach to Nonlinear Analysis and Optimization.* Academic Press, 1970.

[4] Alberto Bemporad. A numerically stable solver for positive semi-definite quadratic programs based on nonnegative least squares. *IEEE Transactions on Automatic Control*, 63(2), 525-531, 2018.

[5] James Bennett and Stan Lanning. The Netflix Prize. *KDD Cup and Workshop*, 2007.

[6] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Treds in Machine Learning*, 3(1), 1-122, 2010.

[7] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[8] James P. Boyle and Richard L. Dykstra. *Advances in Order Restricted Statistical Inference: Proceedings of the Symposium on Order Restricted Statistical Inference*, 28–47, 1986.

[9] Jian-Feng Cai, Emmanuel J. Candés, and Zuowei Shen. A Singular Value Thresholding Algorithm for Matrix Completion. *SIAM Journal on Optimization*, 20(4), 1956-1982, 2010.

[10] Emmanuel J. Candés and Benjamin Recht. Exact Matrix Completion via Convex Optimization. *Foundations of Computational Mathematics*, 9(797), 2009.

[11] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1), 120-145, 2011.

[12] Antonin Chambolle and Thomas Pock. On the ergodic convergence rates of a first-order primal–dual algorithm. In *Mathematical Programming*, 159(1-2), 253-287, 2016.

[13] Patrick L. Combettes and Valérie R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4), 1168-1200, 2005.

[14] Laurent Condat. A primal–dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms. *Journal of Optimization Theory and Applications* 158(2), 460-479, 2013.

[15] Richard Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the American Mathematical Society*, 49(1), 1-23, 1943.

[16] Ernie Esser, Xiaoqun Zhang, and Tony F. Chan. A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science. *SIAM Journal on Imaging Sciences*, 3(4), 1015-1046, 2010.

[17] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*, 2019. URL: `http://www.gurobi.com`.

[18] A. A. Goldstein. Convex programming in Hilbert space. *Bulletin of the American Mathematical Society*, 70, 709-710, 1964.

[19] Jérôme Goudet, Tomas Kay, and Bruce S. Weir. How to estimate kinship. *Molecular Ecology*, 27(20), 4121–4135, 2018.

[20] Bingsheng He and Xiaoming Yuan. Convergence analysis of primal-dual algorithms for a saddle-point problem: From contraction perspective. *SIAM Journal on Imaging Sciences*, 5(1), 119–149, 2012.

[21] David R. Hunter and Kenneth Lange. A tutorial on MM algorithms. *American Statistician*, 58, 30-37, 2004.

[22] Kevin L. Keys, Hua Zhou, and Kenneth Lange. Proximal distance algorithms: Theory and practice. *Journal of Machine Learning Research*, 20(66), 1-38, 2019.

[23] Seyoon Ko and Joong-Ho Won. Optimal minimization of the sum of three convex functions with a linear operator. *Proceedings of Machine Learning Research*, 89, 1185-1194, 2019.

[24] Seyoon Ko, Donghyeon Yu, and Joong-Ho Won. Easily parallelizable and distributable class of algorithms for structured sparsity, with optimal acceleration. *Journal of Computational and Graphical Statistics*, 1-42, 2019.

[25] Kenneth Lange. *MM Optimization Algorithms*. SIAM, 2016.

[26] Kenneth Lange and Kevin L. Keys. The proximal distance algorithm. In *Proceedings of the 2014 International Congress of Mathematicians*, 95-116, 2014.

[27] Ignace Loris and Caroline Verhoeven. On a generalization of the iterative soft-thresholding algorithm for the case of non-separable penalty. In *Inverse Problems*, 27(12), 125007 (15pp), 2011.

[28] Feng Ma, Yiming Bi, and Bin Gao. A prediction–correction-based primal–dual hybrid gradient method for linearly constrained convex minimization. *Numerical Algorithms*, 82, 641–662, 2019.

[29] Mosek ApS. *The MOSEK optimizer API for C 9.1.11*, 2019. URL: `https://docs.mosek.com/9.1/capi/index.html`.

[30] Yu. E. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2), 372–376, 1983.

[31] Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3), 1042-1068, 2016.

[32] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena* 60, 259-268, 1992.

[33] Ernest K. Ryu and Stephen Boyd. A Primer on monotone operator methods survey. *Applied and Computational Mathematics*, 15(1), 3-43, 2016.

[34] Weijie Su, Stephen Boyd, and Emmanuel J. Candés. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. *Journal of Machine Learning Research*, 17(153), 1-43, 2016.

[35] B. C. Vũ. A splitting algorithm for dual monotone inclusions involving cocoercive operators. *Advances in Computational Mathematics* 38(3), 667-681, 2013.

[36] Mingqiang Zhu and Tony Chan, An efficient primal-dual hybrid gradient algorithm For total variation image restoration. *UCLA CAM Report*, 08-34.

국문초록

본 논문에서는 제약조건에 선형연산자가 포함된 볼록 최적화 문제에 대한 근위 거리 알고리즘과 원시-쌍대 복합 경사법을 비교한다. 타 알고리즘들이 제약조건의 선형연산자로 인한 계산상의 어려움을 겪는데 비해, 두 알고리즘은 간단한 업데이트 식으로 문제를 쉽게 해결할 수 있다. 본 논문에서는 근위 거리 알고리즘과 원시-쌍대 복합 경사법을 소개하고, 이들을 여러 통계적 문제들에 적용한 해답을 제시하고, 결과값을 비교하였다. 이 때, 원시-쌍대 복합 경사법의 최적해는 파라미터의 선택의 영향을 적게 받는 반면 근위 거리 알고리즘의 최적해는 파라미터가 부적절하게 선택될 경우 최적성이나 안정성을 잃을 수 있다. 마지막으로, 결과값을 통하여 원시-쌍대 복합 경사법이 뛰어난 성능을 보이며 근위 거리 알고리즘 또한 현존하는 알고리즘에 뒤지지 않음을 확인하였다.

주요어: 볼록 최적화, 근위 거리 알고리즘, 원시-쌍대 복합 경사법, 통계계산

학번: 2018-27696