Sparse Linear Programming via Primal and Dual Augmented Coordinate Descent

Ian E.H. Yen * **Kai Zhong** * **Cho-Jui Hsieh** † **Pradeep Ravikumar** * **Inderjit S. Dhillon** *
* University of Texas at Austin † University of California at Davis
* {ianyen,pradeepr,inderjit}@cs.utexas.edu zhongkai@ices.utexas.edu
† chohsieh@ucdavis.edu

Abstract

Over the past decades, Linear Programming (LP) has been widely used in different areas and considered as one of the mature technologies in numerical optimization. However, the complexity offered by state-of-the-art algorithms (i.e. interior-point method and primal, dual simplex methods) is still unsatisfactory for problems in machine learning with huge number of variables and constraints. In this paper, we investigate a general LP algorithm based on the combination of Augmented Lagrangian and Coordinate Descent (AL-CD), giving an iteration complexity of $O((\log(1/\epsilon))^2)$ with $O(nnz(A))$ cost per iteration, where $nnz(A)$ is the number of non-zeros in the $m \times n$ constraint matrix A, and in practice, one can further reduce cost per iteration to the order of non-zeros in columns (rows) corresponding to the active primal (dual) variables through an active-set strategy. The algorithm thus yields a tractable alternative to standard LP methods for large-scale problems of sparse solutions and $nnz(A) \ll mn$. We conduct experiments on large-scale LP instances from ℓ_1-regularized multi-class SVM, Sparse Inverse Covariance Estimation, and Nonnegative Matrix Factorization, where the proposed approach finds solutions of 10^{-3} precision orders of magnitude faster than state-of-the-art implementations of interior-point and simplex methods. [1]

1 Introduction

Linear Programming (LP) has been studied since the early 19th century and has become one of the representative tools of numerical optimization with wide applications in machine learning such as ℓ_1-regularized SVM [1], MAP inference [2], nonnegative matrix factorization [3], exemplar-based clustering [4, 5], sparse inverse covariance estimation [6], and Markov Decision Process [7]. However, as the demand for scalability keeps increasing, the scalability of existing LP solvers has become unsatisfactory. In particular, most algorithms in machine learning targeting large-scale data have a complexity linear to the data size [8, 9, 10], while the complexity of state-of-the-art LP solvers (i.e. Interior-Point method and Primal, Dual Simplex methods) is still at least quadratic in the number of variables or constraints [11].

The quadratic complexity comes from the need to solve each linear system exactly in both simplex and interior point method. In particular, the simplex method, when traversing from one corner point to another, requires solution to a linear system that has dimension linear to the number of variables or constraints, while in an Interior-Point method, finding the Newton direction requires solving a linear system of similar size. While there are sparse variants of *LU* and *Cholesky decomposition* that can utilize the sparsity pattern of matrix in a linear system, the worst-case complexity for solving such system is at least quadratic to the dimension except for very special cases such as a tri-diagonal or band-structured matrix.

[1] Our solver has been released here: http://www.cs.utexas.edu/~ianyen/LPsparse/

For interior point method (IPM), one remedy to the high complexity is employing an iterative method such as Conjugate Gradient (CG) to solve each linear system inexactly. However, this can hardly tackle the ill-conditioned linear systems produced by IPM when iterates approach boundary of constraints [12]. Though substantial research has been devoted to the development of preconditioners that can help iterative methods to mitigate the effect of ill-conditioning [12, 13], creating a preconditioner of tractable size is a challenging problem by itself [13]. Most commercial LP software thus still relies on exact methods to solve the linear system.

On the other hand, some dual or primal (stochastic) sub-gradient descent methods have cheap cost for each iteration, but require $O(1/\epsilon^2)$ iterations to find a solution of ϵ precision, which in practice can even hardly find a feasible solution satisfying all constraints [14].

Augmented Lagrangian Method (ALM) was invented early in 1969, and since then there have been several works developed Linear Program solver based on ALM [15, 16, 17]. However, the challenge of ALM is that it produces a series of *bound-constrained quadratic problems* that, in the traditional sense, are harder to solve than linear system produced by IPM or Simplex methods [17]. Specifically, in a Projected-CG approach [18], one needs to solve several linear systems via CG to find solution to the bound-constrained quadratic program, while there is no guarantee on how many iterations it requires. On the other hand, Projected Gradient Method (PGM), despite its guaranteed iteration complexity, has very slow convergence in practice. More recently, Multi-block ADMM [19, 20] was proposed as a variant of ALM that, for each iteration, only updates one pass (or even less) blocks of primal variables before each dual update, which however, requires a much smaller step size in the dual update to ensure convergence [20, 21] and thus requires large number of iterations for convergence to moderate precision. To our knowledge, there is still no report on a significant improvement of ALM-based methods over IPM or Simplex method for Linear Programming.

In the recent years, Coordinate Descent (CD) method has demonstrated efficiency in many machine learning problems with bound constraints or other non-smooth terms [9, 10, 22, 23, 24, 25] and has solid analysis on its iteration complexity [26, 27]. In this work, we show that CD algorithm can be naturally combined with ALM to solve Linear Program more efficiently than existing methods on large-scale problems. We provide an $O((\log(1/\epsilon))^2)$ iteration complexity of the Augmented Lagrangian with Coordinate Descent (AL-CD) algorithm that bounds the total number of CD updates required for an ϵ-precise solution, and describe an implementation of AL-CD that has cost $O(nnz(A))$ for each pass of CD. In practice, an active-set strategy is introduced to further reduce cost of each iteration to the active size of variables and constraints for *primal-sparse* and *dual-sparse* LP respectively, where a *primal-sparse* LP has most of variables being zero, and a *dual-sparse* LP has few binding constraints at the optimal solution. Note, unlike in IPM, the conditioning of each subproblem in ALM does not worsen over iterations [15, 16]. The AL-CD framework thus provides an alternative to interior point and simplex methods when it is infeasible to exactly solving an $n \times n$ (or $m \times m$) linear system.

2 Sparse Linear Program

We are interested in solving linear programs of the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) = c^T x \\ s.t. \quad & A_I x \leq b_I \ , \ A_E x = b_E \\ & x_j \geq 0, \ \ j \in [n_b] \end{aligned} \tag{1}$$

where A_I is m_I by n matrix of coefficients and A_E is m_E by n. Without loss of generality, we assume non-negative constraints are imposed on the first n_b variables, denoted as x_b, such that $x = [x_b; \ x_f]$ and $c = [c_b; \ c_f]$. The inequality and equality coefficient matrices can then be partitioned as $A_I = [A_{I,b} \ A_{I,f}]$ and $A_E = [A_{E,b} \ A_{E,f}]$. The dual problem of (1) then takes the form

$$\begin{aligned} \min_{y \in \mathbb{R}^m} \quad & g(y) = b^T y \\ s.t. \quad & -A_b^T y \leq c_b \ , \ -A_f^T y = c_f \\ & y_i \geq 0, \ \ i \in [m_I]. \end{aligned} \tag{2}$$

where $m = m_I + m_E$, $b = [b_I; b_E]$, $A_b = [A_{I,b}; A_{E,b}]$, $A_f = [A_{I,f}; A_{E,f}]$, and $y = [y_I; y_E]$. In most of LP occur in machine learning, m and n are both at scale in the order 10^5 ~10^6, for which an algorithm with cost $O(mn)$, $O(n^2)$ or $O(m^2)$ is unacceptable. Fortunately, there are usually various types of sparsity present in the problem that can be utilized to lower the complexity.

First, the constraint matrix $A = [A_I; A_E]$ are usually pretty sparse in the sense that $nnz(A) \ll mn$, and one can compute matrix-vector product Ax in $O(nnz(A))$. However, in most of current LP solvers, not only matrix-vector product but also a linear system involving A needs to be solved, which in general, has cost much more than $O(nnz(A))$ and can be up to $O(\min(n^3, m^3))$ in the worst case. In particular, the simplex-type methods, when moving from one corner to another, requires solving a linear system that involves a sub-matrix of A with columns corresponding to the basic variables [11], while in an interior point method (IPM), one also needs to solve a *normal equation* system of matrix AD_tA^T to obtain the Newton direction, where D_t is a diagonal matrix that gradually enforces complementary slackness as IPM iteration t grows [11]. While one remedy to the high complexity is to employ iterative method such as *Conjugate Gradient (CG)* to solve the system inexactly within IPM, this approach can hardly handle the ill-conditionedness occurs when IPM iterates approaches boundary [12]. On the other hand, the Augmented Lagrangian approach does not have such asymptotic ill-conditionedness and thus an iterative method with complexity linear to $O(nnz(A))$ can be used to produce sufficiently accurate solution for each sub-problem.

Besides sparsity in the constraint matrix A, two other types of structures, which we termed *primal* and *dual sparsity*, are also prevalent in the context of machine learning. A *primal-sparse* LP refers to an LP with optimal solution x^* comprising only few non-zero elements, while a *dual-sparse* LP refers to an LP with few binding constraints at optimal, which corresponds to the non-zero dual variables. In the following, we give two examples of sparse LP.

L1-Regularized Support Vector Machine The problem of L1-regularized multi-class Support Vector Machine [1]

$$\begin{aligned} \min_{w_m, \xi_i} \quad & \lambda \sum_{m=1}^{k} \|w_m\|_1 + \sum_{i=1}^{l} \xi_i \\ s.t. \quad & w_{y_i}^T x_i - w_m^T x_i \geq e_i^m - \xi_i, \ \forall (i, m) \end{aligned} \tag{3}$$

where $e_i^m = 0$ if $y_i = m$, $e_i^m = 1$ otherwise. The task is dual-sparse since among all samples i and class k, only those leads to misclassification will become binding constraints. The problem (3) is also primal-sparse since it does feature selection through ℓ_1-penalty. Note the constraint matrix in (3) is also sparse since each constraint only involves two weight vectors, and the pattern x_i can be also sparse.

Sparse Inverse Covariance Estimation The Sparse Inverse Covariance Estimation aims to find a sparse matrix Ω that approximate the inverse of Covariance matrix. One of the most popular approach to this solves a program of the form [6]

$$\begin{aligned} \min_{\Omega \in \mathbb{R}^{d \times d}} \quad & \|\Omega\|_1 \\ s.t. \quad & \|S\Omega - I_d\|_{max} \leq \lambda \end{aligned} \tag{4}$$

which is primal-sparse due to the $\|.\|_1$ penalty. The problem has a dense constraint matrix, which however, has special structure where the coefficient matrix S can be decomposed into a product of two low-rank and (possibly) sparse n by d matrices $S = Z^T Z$. In case Z is sparse or $n \ll d$, this decomposition can be utilized to solve the Linear Program much more efficiently. We will discuss on how to utilize such structure in section 4.3.

3 Primal and Dual Augmented Coordinate Descent

In this section, we describe an Augmented Lagrangian method (ALM) that carefully tackles the sparsity in a LP. The choice between Primal and Dual ALM depends on the type of sparsity present in the LP. In particular, a primal AL method can solve a problem of few non-zero variables more efficiently, while dual ALM will be more efficient for problem with few binding constraints. In the following, we describe the algorithm only from the primal point of view, while the dual version can be obtained by exchanging the roles of primal (1) and dual (2).

Algorithm 1 (Primal) Augmented Lagrangian Method

Initialization: $y^0 \in \mathbb{R}^m$ and $\eta_0 > 0$.
repeat
1. Solve (6) to obtain (x^{t+1}, ξ^{t+1}) from y^t.
2. Update $y^{t+1} = y^t + \eta_t \begin{bmatrix} A_I x^{t+1} - b_I + \xi^{t+1} \\ A_E x^{t+1} - b_E \end{bmatrix}$.
3. $t = t + 1$.
4. Increase η_t by a constant factor if necessary.

until $\|[A_I x^t - b_I]_+\|_\infty \leq \epsilon_p$ and $\|A_E x^t - b_E\|_\infty \leq \epsilon$.

3.1 Augmented Lagrangian Method (Dual Proximal Method)

Let $g(y)$ be the dual objective function (2) that takes ∞ if y is infeasible. The primal AL algorithm can be interpreted as a *dual proximal point* algorithm [16] that for each iteration t solves

$$y^{t+1} = \underset{y}{argmin} \quad g(y) + \frac{1}{2\eta_t}\|y - y^t\|^2. \tag{5}$$

Since $g(y)$ is nonsmooth, (5) is not easier to solve than the original dual problem. However, the dual of (5) takes the form:

$$\begin{aligned} \min_{x,\,\xi} \quad & F(x,\xi) = c^T x + \frac{\eta_t}{2}\left\| \begin{array}{c} A_I x - b_I + \xi \\ A_E x - b_E \end{array} + \frac{1}{\eta_t}\begin{bmatrix} y_I^t \\ y_E^t \end{bmatrix} \right\|^2 \\ s.t. \quad & x_b \geq 0,\ \xi \geq 0, \end{aligned} \tag{6}$$

which is a bound-constrained quadratic problem. Note given (x, ξ) as Lagrangian Multipliers of (5), the corresponding y minimizing Lagrangian $\mathcal{L}(x, \xi, y)$ is

$$y(x,\xi) = \eta_t \begin{bmatrix} A_I x - b_I + \xi \\ A_E x - b_E \end{bmatrix} + \begin{bmatrix} y_I^t \\ y_E^t \end{bmatrix}, \tag{7}$$

and thus one can solve (x^*, ξ^*) from (6) and find y^{t+1} through (7). The resulting algorithm is sketched in Algorithm 1. For problem of medium scale, (6) is not easier to solve than a linear system due to non-negative constraints, and thus an ALM is not preferred to IPM in the traditional sense. However, for large-scale problem with $m \times n \gg nnz(A)$, the ALM becomes advantageous since: (i) the conditioning of (6) does not worsen over iterations, and thus allows iterative methods to solve it approximately in time proportional to $O(nnz(A))$. (ii) For a primal-sparse (dual-sparse) problem, most of primal (dual) variables become binding at zero as iterates approach to the optimal solution, which yields a potentially much smaller subproblem.

3.2 Solving Subproblem via Coordinate Descent

Given a dual solution y_t, we employ a variant of Randomized Coordinate Descent (RCD) method to solve subproblem (6). First, we note that, given x, the part of variables in ξ can minimized in closed-form as

$$\xi(x) = [b_I - A_I x - y_I^t/\eta_t]_+, \tag{8}$$

where function $[v]_+$ truncates each element of vector v to be non-negative as $[v]_{+i} = \max\{v_i, 0\}$. Then (6) can be re-written as

$$\begin{aligned} \min_{x} \quad & \hat{F}(x) = c^T x + \frac{\eta_t}{2}\left\| \begin{array}{c} [A_I x - b_I + y_I^t/\eta_t]_+ \\ A_E x - b_E + y_E^t/\eta_t \end{array} \right\|^2 \\ s.t. \quad & x_b \geq 0. \end{aligned} \tag{9}$$

Algorithm 2 RCD for subproblem (6)

INPUT: $\eta_t > 0$ and $(x^{t,0}, w^{t,0}, v^{t,0})$ satisfying relation (11), (12).
OUTPUT: $(x^{t,k}, w^{t,k}, v^{t,k})$
repeat
1. Pick a coordinate j uniformly at random.
2. Compute $\nabla_j \hat{F}(x)$, $\nabla_j^2 \hat{F}(x)$.
3. Obtain Newton direction d_j^*.
4. Do line search (15) to find step size.
5. Update $x^{t,k+1} \leftarrow x^{t,k} + \beta^r d_j^*$.
6. Maintain relation (11), (12).
7. $k \leftarrow k + 1$.

until $\|d^*(x)\|_\infty \leq \epsilon_t$.

Algorithm 3 PN-CG for subproblem (6)

INPUT: $\eta_t > 0$ and $(x^{t,0}, w^{t,0}, v^{t,0})$ satisfying relation (11), (12).
OUTPUT: $(x^{t,k}, w^{t,k}, v^{t,k})$
repeat
1. Identify active variables $\mathcal{A}^{t,k}$.
2. Compute $[\nabla_j F(x)]_{\mathcal{A}^{t,k}}$ and set $\mathcal{D}^{t,k}$.
3. Find Newton direction $d^*_{\mathcal{A}^{t,k}}$ with CG.
4. Find step size via projected line search.
5. Update $x^{t,k+1} \leftarrow (x^{t,k} + \beta^r d_j^*)_+$.
6. Maintain relation (11), (12).
7. $k \leftarrow k + 1$.

until $\|d^*_{A^{t,k}}\|_\infty \leq \epsilon_t$.

Denote the objective function as $\hat{F}(x)$. The gradient of (9) can be expressed as

$$\nabla \hat{F}(x) = c + \eta_t A_I^T [w]_+ + \eta_t A_E^T v \tag{10}$$

where

$$w = A_I x - b_I + y_I^t/\eta_t \tag{11}$$

$$v = A_E x - b_E + y_E^t/\eta_t, \tag{12}$$

and the (generalized) Hessian of (9) is

$$\nabla^2 \hat{F}(x) = \eta_t A_I^T D(w) A_I + \eta_t A_E^T A_E, \tag{13}$$

where $D(w)$ is an m_I by m_I diagonal matrix with $D_{ii}(w) = 1$ if $w_i > 0$ and $D_{ii} = 0$ otherwise.

The RCD algorithm then proceeds as follows. In each iteration k, it picks a coordinate from $j \in \{1, .., n\}$ uniformly at random and minimizes w.r.t. the coordinate. The minimization is conducted by a single-variable Newton step, which first finds the Newton direction d_j^* through minimizing a quadratic approximation

$$\begin{aligned} d_j^* = \underset{d}{argmin} \quad & \nabla_j \hat{F}(x^{t,k}) d + \frac{1}{2} \nabla_j^2 \hat{F}(x^{t,k}) d^2 \\ s.t. \quad & x_j^{t,k} + d \geq 0, \end{aligned} \tag{14}$$

and then conducted a line search to find the smallest $r \in \{0, 1, 2, ...\}$ satisfying

$$\hat{F}(x^{t,k} + \beta^r d_j^* e_j) - \hat{F}(x^{t,k}) \leq \sigma \beta^r (\nabla_j \hat{F}(x^{t,k}) d_j^*). \tag{15}$$

for some line-search parameter $\sigma \in (0, 1/2]$, $\beta \in (0, 1)$, where e_j denotes a vector with only jth element equal to 1 and all others equal to 0. Note the single-variable problem (14) has closed-form solution

$$d_j^* = \left[x_j^{t,k} - \nabla_j \hat{F}(x_j^{t,k}) / \nabla_j^2 \hat{F}(x_j^{t,k}) \right]_+ - x_j^{t,k}, \tag{16}$$

which in a naive implementation, takes $O(nnz(A))$ time due to the computation of (11) and (12). However, in a clever implementation, one can maintain the relation (11), (12) as follows whenever a coordinate x_j is updated by $\beta^r d_j^*$

$$\begin{bmatrix} w^{t,k+1} \\ v^{t,k+1} \end{bmatrix} = \begin{bmatrix} w^{t,k} \\ v^{t,k} \end{bmatrix} + \beta^r d_j^* \begin{bmatrix} a_j^I \\ a_j^E \end{bmatrix}, \tag{17}$$

where $a_j = [a_j^I;\ a_j^E]$ denotes the jth column of A_I and A_E. Then the gradient and (generalized) second-derivative of jth coordinate

$$\begin{aligned} \nabla_j \hat{F}(x) &= c_j + \eta_t \langle a_j^I, [w]_+ \rangle + \eta_t \langle a_j^E, v \rangle \\ \nabla_j^2 \hat{F}(x) &= \eta_t \left(\sum_{i: w_i > 0} (a_{i,j}^I)^2 + \sum_i (a_{i,j}^E)^2 \right) \end{aligned} \tag{18}$$

can be computed in $O(nnz(a_j))$ time. Similarly, for each coordinate update, one can evaluate the difference of function value $\hat{F}(x^{t,k} + d_j^* e_j) - \hat{F}(x^{t,k})$ in $O(nnz(a_j))$ by only computing terms related to the jth variable.

The overall procedure for solving subproblem is summarized in Algorithm 2. In practice, a random permutation is used instead of uniform sampling to ensure that every coordinate is updated once before proceeding to the next round, which can speed up convergence and ease the checking of stopping condition $\|d^*(x)\|_\infty \leq \epsilon_t$, and an active-set strategy is employed to avoid updating variables with $d_j^* = 0$. We describe details in section 4

3.3 Convergence Analysis

In this section, we prove the iteration complexity of AL-CD method. Existing analysis [26, 27] shows that Randomized Coordinate Descent can be up to n times faster than Gradient-based methods in certain conditions. However, to prove a global linear rate of convergence the analysis requires objective function to be strongly convex, which is not true for our sub-problem (6). Here we follow the approach in [28, 29] to show global linear convergence of Algorithm 2 by utilizing the fact that, when restricted to a constant subspace, (6) is strongly convex. All proofs will be included in the appendix.

Theorem 1 (Linear Convergence)**.** *Denote F^* as the optimum of* (6) *and $\bar{x} = [x; \xi]$. The iterates $\{\bar{x}^k\}_{k=0}^{\infty}$ of the RCD Algorithm 2 has*

$$\mathbb{E}[F(\bar{x}^{k+1})] - F^* \leq \left(1 - \frac{1}{\gamma n}\right) \left(\mathbb{E}[F(\bar{x}^k)] - F^*\right), \tag{19}$$

where

$$\gamma = \max\left\{16\eta_t M\theta(F^0 - F^*)\ ,\ 2M\theta(1 + 4L_g^2)\ ,\ 6\right\},$$

$M = \max_{j\in[\bar{n}]} \|\bar{a}_j\|^2$ is an upper bound on coordinate-wise second derivative, and L_g is local Lipschitz-continuous constant of function $g(z) = \eta_t\|z - b + y_t/\eta_t\|^2$, and θ is constant of Hoffman's bound that depends on the polyhedron formed by the set of optimal solutions.

Then the following theorem gives a bound on the number of iterations required to find an ϵ_0-precise solution in terms of the proximal minimization (5).

Theorem 2 (Inner Iteration Complexity)**.** *Denote $y(\bar{x}^k)$ as the dual solution* (7) *corresponding to the primal iterate $\bar{x}^k$. To guarantee*

$$\|y(\bar{x}^k) - y^{t+1}\| \leq \epsilon_0 \tag{20}$$

with probability $1 - p$, it suffices running RCD Algorithm 2 for number of iterations

$$k \geq 2\gamma n \log\left(\sqrt{\frac{2(F(\bar{x}^0) - F^*)}{\eta_t p}}\frac{1}{\epsilon_0}\right).$$

Now we prove the overall iteration complexity of AL-CD. Note that existing linear convergence analysis of ALM on Linear Program [16] assumes exact solutions of subproblem (6), which is not possible in practice. Our next theorem extends the linear convergence result to cases when subproblems are solved *inexactly*, and in particular, shows the total number of coordinate descent updates required to find an ϵ-accurate solution.

Theorem 3 (Iteration Complexity)**.** *Denote $\{\hat{y}^t\}_{t=1}^{\infty}$ as the sequence of iterates obtained from inexact dual proximal updates, $\{y^t\}_{t=1}^{\infty}$ as that generated by exact updates, and y_{S^*} as the projection of y to the set of optimal dual solutions. To guarantee $\|\hat{y}^t - \hat{y}^t_{S_*}\| \leq 2\epsilon$ with probability $1 - p$, it suffices to run Algorithm 1 for*

$$T = (1 + \frac{1}{\alpha}) \log\left(\frac{LR}{\epsilon}\right) \tag{21}$$

outer iterations with $\eta_t = (1 + \alpha)L$, and solve each sub-problem (6) *by running Algorithm 2 for*

$$k \geq 2\gamma n \left(\log\left(\frac{\omega}{\epsilon}\right) + \frac{3}{2}\log\left((1 + \frac{1}{\alpha})\log\frac{LR}{\epsilon}\right)\right) \tag{22}$$

inner iterations, where L is a constant depending on the polyhedral set of optimal solutions, $\omega = \sqrt{\frac{2(1+\alpha)L(F^0 - F^*)}{p}}$, $R = \|\mathbf{prox}_{\eta_t g}(y^0) - y^0\|$, and F^0, F^* are upper and lower bounds on the initial and optimal function values of subproblem respectively.

3.4 Fast Asymptotic Convergence via Projected Newton-CG

The RCD algorithm converges to a solution of moderate precision efficiently, but in some problems a higher precision might be required. In such case, we transfer the subproblem solver from RCD to a *Projected Newton-CG (PN-CG)* method after iterates are close enough to the optimum. Note the Projected Newton method does not have global iteration complexity but has fast convergence for iterates very close to the optimal.

Denote $F(x)$ as the objective in (9). Each iterate of PN-CG begins by finding the set of *active variables* defined as

$$\mathcal{A}^{t,k} = \{j | x_j^{t,k} > 0 \vee \nabla_j F(x^{t,k}) < 0\}. \tag{23}$$

Then the algorithm fixes $x_j^{t,k} = 0, \forall j \notin A^{t,k}$ and solves a Newton linear system w.r.t. $j \in \mathcal{A}^{t,k}$

$$[\nabla^2_{\mathcal{A}^{t,k}} F(x^{t,k})]d = -[\nabla_{\mathcal{A}^{t,k}} F(x^{t,k})] \tag{24}$$

to obtain direction d^* for the current active variables. Let $d_{A^{t,k}}$ denotes a size-n vector taking value in d^* for $j \in A^{t,k}$ and taking value 0 for $j \notin A^{t,k}$. The algorithm then conducts a *projected line search* to find smallest $r \in \{0, 1, 2, ...\}$ satisfying

$$F([x^{t,k} + \beta^r d_{A^{t,k}}]_+) - F(x^{t,k}) \leq \sigma\beta^r(\nabla_j F(x^{t,k}) d_{A^{t,k}}), \tag{25}$$

and update x by $x^{t,k+1} \leftarrow (x^{t,k} + \beta^r d_j^*)_+$. Compared to interior point method, one key to the tractability of this approach lies on the conditioning of linear system (24), which does not worsen as outer iteration t increases, so an iterative *Conjugate Gradient (CG)* method can be used to obtain accurate solution without factorizing the Hessian matrix. The only operation required within CG is the Hessian-vector product

$$[\nabla^2_{\mathcal{A}^{t,k}} F(x^{t,k})]s = \eta_t \ [A_I^T D(w^{t,k}) A_I + A_E^T A_E]_{\mathcal{A}^{t,k}} \ s, \tag{26}$$

where the operator $[.]_{\mathcal{A}^{t,k}}$ takes the sub-matrix with row and column indices belonging to $\mathcal{A}^{t,k}$. For a *primal or dual-sparse* LP, the product (26) can be evaluated very efficiently, since it only involves non-zero elements in columns of A_I, A_E belonging to the active set, and rows of A_I corresponding to the binding constraints for which $D_{ii}(w^{t,k}) > 0$. The overall cost of the product (26) is only

$$O\left(nnz([A_I]_{\mathcal{D}^{t,k},\mathcal{A}^{t,k}}) + nnz([A_E]_{:,\mathcal{A}^{t,k}})\right),$$

where $\mathcal{D}^{t,k} = \{i | w_i^{t,k} > 0\}$ is the set of current binding constraints. Considering that the computational bottleneck of PN-CG is on the *CG* iterations for solving linear system (24), the efficient computation of product (26) reduces the overall complexity of PN-CG significantly. The whole procedure is summarized in Algorithm 3.

4 Practical Issues

4.1 Precision of Subproblem Minimization

In practice, it is unnecessary to solve subproblem (6) to high precision, especially for iterations of ALM in the beginning. In our implementation, we employ a two-phase strategy, where in the first phase we limit the cost spent on each sub-problem (6) to be a constant multiple of $nnz(A)$, while in the second phase we dynamically increment the AL parameter η_t and inner precision ϵ_t to ensure sufficient decrease in the primal and dual infeasibility respectively. The two-phase strategy is particularly useful for primal or dual-sparse problem, where sub-problem in the latter phase has smaller active set that results in less computation cost even when solved to high precision.

4.2 Active-Set Strategy

Our implementation of Algorithm 2 maintains an active set of variables $\mathcal{A}$, which initially contains all variables, but during the RCD iterates, any variable x_j binding at 0 with gradient $\nabla_j F$ greater than a threshold δ will be excluded from $\mathcal{A}$ till the end of each subproblem solving. $\mathcal{A}$ will be re-initialized after each dual proximal update (7). Note in the initial phase, the cost spent on each subproblem is a constant multiple of $nnz(A)$, so if $|\mathcal{A}|$ is small one would spend more iterations on the active variables to achieve faster convergence.

4.3 Dealing with Decomposable Constraint Matrix

When we have a m by n constraint matrix $A = UV^T$ that can be decomposed into product of an $m \times r$ matrix U and a $r \times n$ matrix V^T, if $r \ll \min\{m, n\}$ or $nnz(U) + nnz(V) \ll nnz(A)$, we can re-formulate the constraint $Ax \leq b$ as $Uz \leq b$, $V^T x = z$ with auxiliary variables $z \in \mathbb{R}^r$. This new representation reduce the cost of Hessian-vector product in Algorithm 3 and the cost of each pass of CD in Algorithm 2 from $O(nnz(A))$ to $O(nnz(U) + nnz(V))$.

5 Numerical Experiments

Table 1: Timing Results (in sec. unless specified o.w.) on Multiclass L1-regularized SVM

Data	n_b	m_I	P-Simp.	D-Simp.	Barrier	D-ALCD	P-ALCD
rcv1	4,833,738	778,200	$>$ 48hr	$>$ 48hr	$>$ 48hr	3,452	**3,155**
news	2,498,415	302,765	$>$ 48hr	37,912	$>$ 48hr	**148**	395
sector	11,597,992	666,848	$>$ 48hr	9,282	$>$ 48hr	**1,419**	2,029
mnist	75,620	540,000	6,454	2,556	73,036	**146**	7,207
cod-rna.rf	69,537	59,535	86,130	5,738	$>$ 48hr	3,130	**2,676**
vehicle	79,429	157,646	3,296	143.33	8,858	**31**	598
real-sim	114,227	72,309	$>$ 48hr	49,405	89,476	**179**	297

Table 2: Timing Results (in sec. unless specified o.w.) on Sparse Inverse Covariance Estimation

Data	n_b	m_I	m_E	n_f	P-Simp	D-Simp	Barrier	D-ALCD	P-ALCD
textmine	60,876	60,876	43,038	43,038	$>$ 48hr	$>$ 48hr	$>$ 48hr	43,096	**18,507**
E2006	55,834	55,834	32,174	32,174	$>$ 48hr	$>$ 48hr	94623	$>$ 48hr	**4,207**
dorothea	47,232	47,232	1,600	1,600	3,980	103	82	47	**38**

Table 3: Timing Results (in sec. unless specified o.w.) for Nonnegative Matrix Factorization.

Data	n_b	m_I	P-Simp.	D-Simp.	Barrier	D-ALCD	P-ALCD
micromass	2,896,770	4,107,438	$>$ 96hr	$>$ 96hr	280,230	12,966	**12,119**
ocr	6,639,433	13,262,864	$>$ 96hr	$>$ 96hr	284,530	**40,242**	$>$ 96hr

In this section, we compare the AL-CD algorithm with state-of-the-art implementation of interior point and primal, dual Simplex methods in commercial LP solver CPLEX, which is of top efficiency among many LP solvers as investigated in [30]. For all experiments, the stopping criteria is set to require both primal and dual infeasibility (in the ℓ_∞-norm) smaller than 10^{-3} and set the initial sub-problem tolerance $\epsilon_t = 10^{-2}$ and $\eta_t = 1$. The LP instances are generated from L1-SVM (3), Sparse Inverse Covariance Estimation (4) and Nonnegative Matrix Factorization [3]. For the Sparse Inverse Covariance Estimation problem, we use technique introduced in section 4.3 to decompose the low-rank matrix S, and since (4) results in d independent problems for each column of the estimated matrix, we report result on only one of them. The data source and statistics are included in the appendix.

Among all experiments, we observe that the proposed primal, dual AL-CD methods become particularly advantageous when the matrix A is sparse. For example, for text data set *rcv1*, *real-sim* and *news* in Table 1, the matrix A is particularly sparse and AL-CD can be orders of magnitude faster than other approaches by avoiding solving $n \times n$ linear system exactly. In addition, the dual-ALCD (also dual simplex) is more efficient in L1-SVM problem due to the problem's strong dual sparsity, while the primal-ALCD is more efficient on the primal-sparse Inverse Covariance estimation problem. For the Nonnegative Matrix Factorization problem, both the dual and primal LP solutions are not particularly sparse due to the choice of matrix approximation tolerance (1% of #samples), but the AL-CD approach is still comparably more efficient.

Acknowledgement We acknowledge the support of ARO via W911NF-12-1-0390, and the support of NSF via grants CCF-1320746, CCF-1117055, IIS-1149803, IIS-1320894, IIS-1447574, DMS-1264033, and NIH via R01 GM117594-01 as part of the Joint DMS/NIGMS Initiative to Support Research at the Interface of the Biological and Mathematical Sciences.

References

[1] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. *NIPS*, 2004.

[2] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[3] N. Gillis and R. Luce. Robust near-separable nonnegative matrix factorization using linear optimization. *JMLR*, 2014.

[4] A. Nellore and R. Ward. Recovery guarantees for exemplar-based clustering. *arXiv.*, 2013.

[5] I. Yen, X. Lin, K. Zhong, P. Ravikumar, and I. Dhillon. A convex exemplar-based approach to MAD-Bayes dirichlet process mixture models. In *ICML*, 2015.

[6] M. Yuan. High dimensional inverse covariance matrix estimation via linear programming. *JMLR*, 2010.

[7] D. Bello and G. Riano. Linear programming solvers for Markov decision processes. In *Systems and Information Engineering Design Symposium*, pages 90–95, 2006.

[8] T. Joachims. Training linear svms in linear time. In *KDD*. ACM, 2006.

[9] C. Hsieh, K. Chang, C. Lin, S.S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *ICML*, volume 307. ACM, 2008.

[10] G. Yuan, C. Hsieh K. Chang, and C. Lin. A comparison of optimization methods and software for large-scale l1-regularized linear classification. *JMLR*, 11, 2010.

[11] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 2006.

[12] J. Gondzio. Interior point methods 25 years later. *EJOR*, 2012.

[13] J. Gondzio. Matrix-free interior point method. *Computational Optimization and Applications*, 2012.

[14] V.Eleuterio and D.Lucia. Finding approximate solutions for large scale linear programs. *Thesis*, 2009.

[15] Evtushenko, Yu. G, Golikov, AI, and N. Mollaverdy. Augmented lagrangian method for large-scale linear programming problems. *Optimization Methods and Software*, 20(4-5):515–524, 2005.

[16] F. Delbos and J.C. Gilbert. Global linear convergence of an augmented lagrangian algorithm for solving convex quadratic optimization problems. 2003.

[17] O. Güler. Augmented lagrangian algorithms for linear programming. *Journal of optimization theory and applications*, 75(3):445–470, 1992.

[18] J. Moré J and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1(1):93–113, 1991.

[19] M. Hong and Z. Luo. On linear convergence of alternating direction method of multipliers. *arXiv*, 2012.

[20] H. Wang, A. Banerjee, and Z. Luo. Parallel direction method of multipliers. In *NIPS*, 2014.

[21] C.Chen, B.He, Y.Ye, and X.Yuan. The direct extension of admm for multi-block convex minimization problems is not necessarily convergent. *Mathematical Programming*, 2014.

[22] I.Dhillon, P.Ravikumar, and A.Tewari. Nearest neighbor based greedy coordinate descent. In *NIPS*, 2011.

[23] I. Yen, C. Chang, T. Lin, S. Lin, and S. Lin. Indexed block coordinate descent for large-scale linear classification with limited memory. In *KDD*. ACM, 2013.

[24] I. Yen, S. Lin, and S. Lin. A dual-augmented block minimization framework for learning with limited memory. In *NIPS*, 2015.

[25] K. Zhong, I. Yen, I. Dhillon, and P. Ravikumar. Proximal quasi-Newton for computationally intensive l1-regularized m-estimators. In *NIPS*, 2014.

[26] P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1-2):1–38, 2014.

[27] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[28] P. Wang and C. Lin. Iteration complexity of feasible descent methods for convex optimization. *The Journal of Machine Learning Research*, 15(1):1523–1548, 2014.

[29] I. Yen, C. Hsieh, P. Ravikumar, and I.S. Dhillon. Constant nullspace strong convexity and fast convergence of proximal methods under high-dimensional settings. In *NIPS*, 2014.

[30] B. Meindl and M. Templ. Analysis of commercial and free and open source solvers for linear optimization problems. *Eurostat and Statistics Netherlands*, 2012.

[31] A.J. Hoffman. On approximate solutions of systems of linear inequalities. *Journal of Research of the National Bureau of Standards*, 49(4):263–265, 1952.

[32] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.

[33] I. Yen, T. Lin, S. Lin, P. Ravikumar, and I. Dhillon. Sparse random feature algorithm as coordinate descent in Hilbert space. In *NIPS*, 2014.

6 Appendix A — Proof for Convergence Analysis

6.1 Linear Convergence of Augmented Lagrangian Method

Theorem 4. *Let $\{y^t\}_{t=0}^{\infty}$ be the sequences of dual variables produced by Algorithm 1 and $\{(x^t, \xi_t)\}_{t=0}^{\infty}$ be the corresponding sequence of solutions to the primal Augmented Lagrangian problem. Denote*

$$\Delta^t = \frac{1}{\eta_t}(y^{t+1} - y^t) = \begin{bmatrix} A_I x^t - b_I + \xi^t \\ A_E x^t - b_E \end{bmatrix} \in \partial g(y^t). \tag{27}$$

and $\Pi_{S^*}(y^t)$ as the projection of y^t to the set of optimal dual solutions. Then we have

$$\|y^t - \Pi_{S^*}(y^t)\| \leq L\|\Delta^t\| \tag{28}$$

and

$$\|\Delta^{t+1}\| \leq \min\left(\frac{L}{\eta_t}, 1\right)\|\Delta^t\|, \tag{29}$$

where $L := L(S^*, y^0) > 0$ is a constant depending on the solution set S^* and initial distance to this set $R = \|y^0 - \Pi_{S^*}(y^0)\|$.

Proof. This theorem is a special case of the linear convergence proof in [16]. In particular, the Linear Program (1) can be written as

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) = c^T x \\ s.t. \quad & \begin{bmatrix} A_I & I \\ A_E & O \end{bmatrix} \begin{bmatrix} x \\ \xi \end{bmatrix} = \begin{bmatrix} b_I \\ b_E \end{bmatrix}, \\ & x_j \geq 0, \;\; j = 1...n_b \\ & \xi_i \geq 0, \;\; i = 1...m_I, \end{aligned} \tag{30}$$

which is a special case of the Quadratic Programming formulation analyzed in [16] with quadratic term $Q = 0$ (which is positive semi-definite). The analysis assumes all iterates y^t to be within a bounded distance R to the optimal solution set, which is satisfied with $R = \|y^0 - \Pi_{S^*}(y^t)\|$ since by non-expansiveness of proximal operator, we have

$$\|y^{t+1} - \Pi_{S^*}(y^{t+1})\| \leq \|y^{t+1} - \Pi_{S^*}(y^t)\| = \|\mathbf{prox}_g(y^t) - \mathbf{prox}_g(\Pi_{S^*}(y^t))\| \leq \|y^t - \Pi_{S^*}(y^t)\|,$$

where

$$\mathbf{prox}_g(y^t) = \underset{y}{argmin}\; g(y) + \frac{\eta_t}{2}\|y - y^t\|^2,$$

and thus the distance of each iterate to the optimal set is bounded by $R = \|y^0 - \Pi_{S^*}(y^0)\|$. Inequalities (28), (29) then follow from Proposition 4.4 and Theorem 4.5 of [16] respectively, where the constant L is defined through characteristics of S^* and an upper bound R on the distance to solution set. □

We then have following outer iteration complexity for Algorithm 1, assuming each proximal subproblem (6) is solved exactly.

Corollary 1 (Outer Iteration Complexity)**.** *Setting $\eta_t \geq \eta = (1 + \alpha)L$, we have*

$$\|y^t - \Pi_{S^*}(y^t)\| \leq \epsilon$$

by performing

$$t \geq (1 + \frac{1}{\alpha}) \log\left(\frac{L\|\Delta^0\|}{\epsilon}\right)$$

iterations of Algorithm (1)*, where $\|\Delta^0\| = \|\mathbf{prox}_{\eta_t g}(y^0) - y^0\|$.*

Proof. For $\eta_t \geq \eta = (1 + \alpha)L$, we have

$$\|\Delta^{t+1}\| \leq (1 - \frac{1}{z})\|\Delta^t\|,$$

where $z = (1 + \frac{1}{\alpha})$, and thus for

$$t \geq (1 + \frac{1}{\alpha}) \log \left(\frac{L\|\Delta^0\|}{\epsilon} \right),$$

we have

$$\|\Delta^t\| \leq (1 - \frac{1}{z})^{z \log \frac{L\|\Delta^0\|}{\epsilon}} \|\Delta^0\| \leq (e^{-1})^{\log \frac{L\|\Delta^0\|}{\epsilon}} \|\Delta_0\| \leq \frac{\epsilon}{L},$$

and therefore by (28), $\|y^t - \Pi_{S^*}(y^t)\| \leq \epsilon$. □

6.2 Linear Convergence of Randomized Coordinate Descent on Subproblem (6)

In this section, we prove linear convergence of Algorithm 2 to the optimum of sub-problem (6) by exploiting the fact that objective (6), though not being strongly convex, has strong convexity when restricted to a constant linear subspace [28, 29]. In particular, denote $\bar{n} = n + m_I$ and

$$\bar{x} = \begin{bmatrix} x \\ \xi \end{bmatrix} \in \mathbb{R}^{\bar{n}} \ , \ \bar{c} = \begin{bmatrix} c \\ 0 \end{bmatrix} \ , \ \bar{A} = \begin{bmatrix} A_I & I \\ A_E & O \end{bmatrix}.$$

We can express the objective (6) as

$$\min_{x, x_b \geq 0, \xi \geq 0} \quad F(\bar{x}) = \bar{c}^T \bar{x} + g(\bar{A}\bar{x}), \tag{31}$$

where

$$g(z) = \frac{\eta_t}{2} \|z - b - \frac{1}{\eta_t} y^t\|^2$$

is η_t-strongly convex w.r.t. z and therefore $F(\bar{x})$ is strongly convex when restricted to the space $\mathcal{N}^\perp$, where $\mathcal{N} = Null(\bar{A})$ is the Nullspace of constraint matrix $\bar{A}$. Formally, a Constant Nullspace Strongly Convex (CNSC) function has the following properties.

Lemma 1 (CNSC [29])**.** *Let $\mathcal{N} = Null(\bar{A})$ be the Nullspace of $\bar{A}$ and $H = \nabla^2 F(\bar{x})$ be the Hessian matrix of* (31)*. For any $\bar{x} \in \mathbb{R}^{\bar{n}}$, we can express it as $\bar{x} = u + v$ where $u = \Pi_{\mathcal{N}}(\bar{x})$, $v = \Pi_{\mathcal{N}^\perp}(\bar{x})$ s.t.*

$$Hu = 0 \tag{32}$$

and

$$v^T H v \geq m\|v\|^2, \tag{33}$$

for some $m > 0$.

Proof. The Hessian of (31) can be written as

$$\nabla^2 F(\bar{x}) = H = \eta_t \bar{A}^T \bar{A}$$

and thus (32) can be easily verified. On the other hand, (33) holds with $m = \eta_t \lambda_{min} > 0$, where λ_{min} denotes minimum *positive* eigenvalue of $\bar{A}^T \bar{A}$. □

Then we can profile the optimal solution of (31) with the following condition.

Lemma 2 (Optimality Condition)**.** *Express subproblem objective* (31) *as*

$$F(\bar{x}) + h(\bar{x}),$$

where $h(\bar{x}) = \sum_{j \in [\bar{n}] \setminus \{n_b+1 \ldots n_b+n_f\}} h_j(\bar{x}_j)$ with

$$h_j(\bar{x}) = \begin{cases} 0 & , \bar{x}_j \geq 0 \\ \infty & , o.w.. \end{cases} \tag{34}$$

Then there are unique ρ^*, s^* and t^* s.t. $\bar{x}^*$ is optimal solution of (31) *iff*

$$-\nabla F(\bar{x}^*) = -\bar{c} - \nabla g(t^*) = \rho^* \in \partial h(\bar{x}) \tag{35}$$

and $\bar{c}^T \bar{x}^* = s^*$ and $\bar{A}\bar{x}^* = t^*$.

Proof. The first-order condition implies (35) to be necessary and sufficient for $\bar{x}^*$ to be optimal, so we only need to verify the uniqueness of ρ^*, s^* and t^*. Consider two solutions $\bar{x}_1$, $\bar{x}_2$ that are both optimal. Denote $\Delta x = \bar{x}_1 - \bar{x}_2$. By convexity of $h(\bar{x})$, we have

$$\langle -\nabla F(\bar{x}_1) + \nabla F(\bar{x}_2), \Delta x \rangle = \langle \rho_1 - \rho_2, \Delta x \rangle \geq 0, \tag{36}$$

Note for quadratic $F(\bar{x})$, the Hessian $\nabla^2 F(\bar{x}) = H$ is constant and thus

$$\nabla F(\bar{x}_1) - \nabla F(\bar{x}_2) = H\Delta x. \tag{37}$$

Then by CNSC condition (32), we have

$$\langle -\nabla F(\bar{x}_1) + \nabla F(\bar{x}_2), \Delta x \rangle = \langle -H\Delta x, \Delta x \rangle = -\Delta v^T H \Delta v \tag{38}$$

where $\Delta v = \Pi_{\mathcal{N}^\perp}(\Delta x)$ is the projection of Δx onto the subspace $\mathcal{N}^\perp$. Then by CNSC (33),

$$-\Delta v^T H \Delta v \leq -m\|\Delta v\|^2$$

for some $m > 0$, but (36) implies

$$-\Delta v^T H \Delta v \geq 0.$$

Then the above two inequalities can simultaneously hold only if $\Delta v = 0$, which means the optimal v^* as well as $t^* = \bar{A}\bar{x}^* = \bar{A}v^*$ are unique. Furthermore, the optimal $\rho^* = -\bar{c} - \nabla g(t^*)$ and $s^* = F^* - g(t^*)$ are also unique. □

From Lemma 2, the set of optimal solutions forms a polyhedral set satisfying (i) $\bar{A}\bar{x} = t^*$, (ii) $\bar{c}^T\bar{x} = s^*$ and (iii) $x_b \geq 0$, $\xi \geq 0$. Then we can bound the distance of any point $\bar{x}$ to the polyhedral set by the amount of infeasibility to the three (in)equalities based on Hoffman's bound introduced as follows.

Lemma 3 (Hoffman's Bound)**.** *Let $\mathcal{S} = \{x \in \mathbb{R}^d \mid Ax \leq b,\ Ex = c\}$ be a polyhedral set. Then for any point $x \in \mathbb{R}^d$,*

$$\|x - \Pi_{\mathcal{S}}(x)\|_2^2 \leq \theta \left\| \begin{array}{c} [Ax - b]_+ \\ Ex - c \end{array} \right\|_2^2 \tag{39}$$

where $\Pi_{\mathcal{S}}(x) = arg\min_{y \in \mathcal{S}} \|y - x\|$ is the projection of x to the set $\mathcal{S}$, and $\theta > 0$ is a constant depending on the polyhedral set $\mathcal{S}$.

Proof. The Hoffman's bound first appears in [31] and a proof for the ℓ_2-norm's version (39) and the definition of the constant $\theta(\mathcal{S})$ can be found in [28] (lemma 4.3). □

Note for any feasible descent method (such as Coordinate Descent method), all iterates $\{\bar{x}^k\}_{k=1}^\infty$ are feasible, and therefore one can bound the distance of any iterate to the set of optimal solutions by the amount of infeasibility to the two conditions $\bar{A}\bar{x} = t^*$, $\bar{c}^T\bar{x} = s^*$ as

$$\|\bar{x} - \Pi_{\mathcal{S}}(\bar{x})\|^2 \leq \theta(\mathcal{S}) \left(\|\bar{A}\bar{x} - t^*\|^2 + \|\bar{c}^T\bar{x} - s^*\|^2 \right), \tag{40}$$

which plays an important role in the proof of linear convergence of Randomized Coordinate Descent on the CNSC function (31). Now we move on to lemmas specific to Algorithm 2. For simplicity, we will analyze RCD that employs a conservative step size $1/\nabla^2_{jj}\bar{F}(\bar{x}) = 1/(\eta_t\|\bar{a}_j\|^2)$ instead of the one using dynamic line search (15). However, the result only differs by a constant factor $\sigma\beta$ (line search parameter) on the descent amount.

Lemma 4 (Descent Amount)**.** *The expected descent amount for each RCD update of Algorithm 2 has*

$$\mathbb{E}[F(\bar{x}^{k+1})] - F(\bar{x}^k) \leq \frac{1}{n}\left(\min_\delta\ h(\bar{x}^k + \delta) + \langle \nabla F(\bar{x}^k), \delta \rangle + \frac{M\eta_t}{2}\|\delta\|^2 \right), \tag{41}$$

where $M \geq \max_{j \in [n]} \|\bar{a}_j\|^2$ is an upper bound on the coordinate-wise second derivative, $\bar{a}_j$ is the j-th column of $\bar{A}$.

Proof. First, notice that Algorithm 2 maintains ξ (i.e. $\bar{x}_{n+1},..,\bar{x}_{n+m_f}$) to be optimal given other variables x through equation (8), so we have

$$0 = \min_{\delta_j} \; h_j(\bar{x}_j + \delta_j) + \nabla_j F(\bar{x}^k)\delta_j + \frac{M\eta_t}{2}\delta_j^2 \;,\; j = n+1, ..., n+m_I. \tag{42}$$

Therefore, the algorithm picks coordinate uniformly from $\{1...n\}$ (without $\{n+1, ..., n+m_I\}$) to update. Note the constant

$$M \geq \frac{1}{\eta_t} \max_{j\in[n]} |\nabla^2_{jj} F(\bar{x})| = \max_{j\in[\bar{n}]} \|\bar{a}_j\|^2$$

upper bounds the coordinate-wise second-derivative of both $F(\bar{x})$ and $\hat{F}(x) = \min_\xi F(x,\xi)$. Therefore, denote e_j as vector of all zeros except value 1 at the j-th coordinate. We have

$$\begin{aligned}
F(\bar{x}^{k+1}) - F(\bar{x}^k) &= F(x^{k+1}, \xi(x^{k+1})) - F(x^k, \xi^k) \\
&\leq F(x^{k+1}, \xi^k) - F(x^k, \xi^k) \\
&= \min_{\delta_j} \; h_j(x_j^k + \delta_j) + \nabla_j F(x^k)\delta_j + \frac{\eta_t\|\bar{a}_j\|^2}{2}\delta_j^2 \\
&\leq \min_{\delta_j} \; h_j(x_j^k + \delta_j) + \nabla_j F(x^k)\delta_j + \frac{M\eta_t}{2}\delta_j^2.
\end{aligned}$$

Taking expectation of LHS and RHS w.r.t. j yields the result. □

Finally, notice that the function $g(z) = \frac{\eta_t}{2}\|z - b + y^t/\eta_t\|^2$ is locally Lipschitz-continuous with constant $L_g = \eta_t R_z$ for z satisfying $\|z - b + y^t/\eta_t\| \leq R_z$, that is,

$$|g(z_1) - g(z_2)| \leq L_g \|z_1 - z_2\| \tag{43}$$

for $\forall z_1, z_2$ with $\|z_1 - b + y^t/\eta_t\| \leq R_z$, $\|z_2 - b + y^t/\eta_t\| \leq R_z$, where L_g is an upper bound on the magnitude of dual iterates $\|y^{t+1}\| = \|\eta_t(\bar{A}\bar{x}^k - b) + y^t\|$.

From simplicity of analysis, in the following, we slightly loosen upper bounds by setting constants $L_g \leftarrow \max(L_g, 1)$, $M \leftarrow \max(M, 1)$, $\theta \leftarrow \max(\theta, 1)$, such that $L_g, M, \theta \geq 1$. Then we are ready to prove the main theorem of this section.

Theorem 5 (Linear Convergence)**.** *The iterates $\{\bar{x}^k\}_{k=0}^\infty$ of RCD Algorithm satisfy*

$$\mathbb{E}[F(\bar{x}^{k+1})] - F^* \leq \left(1 - \frac{1}{n\gamma}\right)\left(\mathbb{E}[F(\bar{x}^k)] - F^*\right)$$

where F^* is the optimum of (6) *and*

$$\gamma = \max\left\{16\eta_t M\theta(F^0 - F^*) \;,\; 2M\theta(1 + 4L_g^2) \;,\; 6\right\}.$$

Proof. Let $\bar{x}^* = \Pi_{\mathcal{S}}(\bar{x}^k)$ be the projection of $\bar{x}^k$ to the set of optimal solutions. From Lemma 4, we have

$$\begin{aligned}
\mathbb{E}[F(\bar{x}^{k+1})] - F(\bar{x}^k) &\leq \frac{1}{n}\left(\min_\delta \; h(\bar{x}^k + \delta) + \langle \nabla F(\bar{x}^k), \delta\rangle + \frac{M\eta_t}{2}\|\delta\|^2\right) \\
&\leq \frac{1}{n}\left(\min_\delta \; h(\bar{x}^k + \delta) + F(\bar{x}^k + \delta) - F(\bar{x}^k) + \frac{M\eta_t}{2}\|\delta\|^2\right) \\
&\leq \frac{1}{n}\left(\min_{\alpha\in[0,1]} \; F(\bar{x}^k + \alpha(\bar{x}^* - \bar{x}^k)) - F(\bar{x}^k) + \frac{M\eta_t\alpha^2}{2}\|\bar{x}^* - \bar{x}^k\|^2\right) \\
&\leq \frac{1}{n}\left(\min_{\alpha\in[0,1]} \; -\alpha(F(\bar{x}^k) - F(\bar{x}^*)) + \frac{M\eta_t\alpha^2}{2}\|\bar{x}^* - \bar{x}^k\|^2\right),
\end{aligned} \tag{44}$$

where the second and fourth inequality follow from the convexity of $F(\bar{x})$, and the third inequality follows from the fact that both $\bar{x}^*$ and $\bar{x}^k$ are feasible ($h(\bar{x}^*) = h(\bar{x}^k) = 0$). Now based on the error bound inequality (40), we discuss two cases.

Case 1: $4L_g^2\|\bar{A}\bar{x} - t^*\|^2 < (\bar{c}^T\bar{x} - s^*)^2$.

In this case, we have

$$\begin{aligned}\|\bar{x}^k - \bar{x}^*\|^2 &\leq \theta\left(\|\bar{A}\bar{x}^k - t^*\|^2 + \|\bar{c}^T\bar{x}^k - s^*\|^2\right)\\ &\leq \theta\left(\frac{1}{4L_g^2} + 1\right)(\bar{c}^T\bar{x}^k - s^*)^2 \leq 2\theta(\bar{c}^T\bar{x}^k - s^*)^2\end{aligned} \tag{45}$$

and

$$|\bar{c}^T\bar{x}^k - s^*| \geq 2L_g\|\bar{A}\bar{x}^k - t^*\| \geq 2|g(\bar{A}\bar{x}^k) - g(t^*)|.$$

Note in this case, $\bar{c}^T\bar{x}^k - s^*$ must be non-negative. Otherwise,

$$\begin{aligned}F(\bar{x}^k) - F^* &= g(\bar{A}\bar{x}^k) - g(t^*) + (\bar{c}^T\bar{x}^k - s^*)\\ &\leq |g(\bar{A}\bar{x}^k) - g(t^*)| - |\bar{c}^T\bar{x}^k - s^*|\\ &\leq -\frac{1}{2}|\bar{c}^T\bar{x}^k - s^*| < 0,\end{aligned}$$

leads to contradiction (since $\bar{x}^k$ is feasible, $F(\bar{x}^k)$ cannot be smaller than F^*). Therefore, we have

$$\begin{aligned}F(\bar{x}^k) - F^* &= g(\bar{A}\bar{x}^k) - g(t^*) + \bar{c}^T\bar{x}^k - s^*\\ &\geq -|g(\bar{A}\bar{x}^k) - g(t^*)| + \bar{c}^T\bar{x}^k - s^*\\ &\geq \frac{1}{2}(\bar{c}^T\bar{x}^k - s^*).\end{aligned} \tag{46}$$

Combining (44), (45), and (46), we have

$$\begin{aligned}\mathbb{E}[F(\bar{x}^{k+1})] - F(\bar{x}^k) &\leq \frac{1}{n}\min_{\alpha\in[0,1]} -\frac{\alpha}{2}(\bar{c}^T\bar{x}^k - s^*) + \frac{2\eta_t M\theta\alpha^2}{2}(\bar{c}^T\bar{x}^k - s^*)^2\\ &= \begin{cases} -1/(16\eta_t M\theta n) & ,\ 1/(4\eta_t M\theta(\bar{c}^T\bar{x}^k - s^*)) \leq 1\\ -\frac{1}{4n}(\bar{c}^T\bar{x} - s^*) & ,\ o.w.\end{cases}\end{aligned}$$

Furthermore, we have

$$-\frac{1}{16\eta_t M\theta n} \leq -\frac{1}{16\eta_t M\theta n(F^0 - F^*)}\left(F(\bar{x}^*) - F^*\right)$$

where $F^0 = F(\bar{x}^0)$, and

$$-\frac{1}{4n}(\bar{c}^T\bar{x} - s^*) \leq -\frac{1}{6n}(F(\bar{x}^k) - F^*)$$

since $F(\bar{x}^k) - F^* \leq |g(\bar{A}\bar{x}^k) - g(t^*)| + \bar{c}^T\bar{x}^k - s^* \leq \frac{3}{2}(\bar{c}^T\bar{x}^k - s^*)$. In summary, for Case 1 we obtain

$$\mathbb{E}[F(\bar{x}^{k+1})] - F^* \leq (1 - \frac{1}{n\gamma_1})\left(\mathbb{E}[F(\bar{x}^k)] - F^*\right) \tag{47}$$

where

$$\gamma_1 = \max\left\{16\eta_t M\theta(F^0 - F^*)\ ,\ 6\right\}. \tag{48}$$

Case 2: $4L_g^2\|\bar{A}\bar{x}^k - t^*\|^2 \geq (\bar{c}^T\bar{x}^k - s^*)^2$.

In this case, we have

$$\|\bar{x}^k - \bar{x}^*\|^2 \leq \theta\left(1 + 4L_g^2\right)\|\bar{A}\bar{x}^k - t^*\|^2, \tag{49}$$

and by strong convexity of $g(z)$,

$$F(\bar{x}^k) - F^* \geq \bar{c}^T(\bar{x}^k - \bar{x}^*) + \nabla g(t^*)^T\bar{A}(\bar{x}^k - \bar{x}^*) + \frac{\eta_t}{2}\|\bar{A}\bar{x}^k - t^*\|^2.$$

Adding inequality $0 = h(\bar{x}^k) - h(\bar{x}^*) \geq \langle\rho^*, \bar{x}^k - \bar{x}^*\rangle$ for some $\rho^* \in \partial h(\bar{x}^*)$ to the above gives

$$F(\bar{x}^k) - F^* \geq \frac{\eta_t}{2}\|\bar{A}\bar{x}^k - t^*\|^2 \tag{50}$$

since $\rho^* + \bar{c} + \nabla g(t^*)^T \bar{A} = \rho^* + \nabla F(\bar{x}^*) = 0$. Combining (44), (49), and (50), we obtain

$$\begin{aligned}\mathbb{E}[F(\bar{x}^{k+1})] - F(\bar{x}^k) &\leq \frac{1}{n} \min_{\alpha\in[0,1]} -\alpha(F(\bar{x}^k) - F^*) + \frac{M\theta(1+4L_g^2)\alpha^2}{2}\left(F(\bar{x}^k) - F^*\right) \\ &= -\frac{1}{2M\theta(1+4L_g^2)n}\left(F(\bar{x}^k) - F^*\right)\end{aligned} \tag{51}$$

Combining results of Case 1 (47) and Case 2 (51), and taking expectation on both sides w.r.t. the history leads to the result. □

We then bounds the number of iterations required to achieve ϵ sub-optimality with high probability $1-p$ by the following corollary.

Corollary 2 (Inner Iteration Complexity)**.** *To guarantee*

$$F(\bar{x}^k) - F^* \leq \epsilon \tag{52}$$

with probability $1-p$, it suffices running RCD Algorithm 2 for

$$k \geq \gamma n \log\left(\frac{F(\bar{x}^0) - F^*}{\epsilon p}\right)$$

iterations, where γ is constant defined in Theorem 5.

Proof. We use the Theorem 1 of [26] to transfer the linear convergence in expectation (19) into iteration complexity. To do this, we express (19) in the form

$$\mathbb{E}[F(\bar{x}^{k+1})] - F^* \leq \left(1 - \frac{1}{c}\right)\left(\mathbb{E}[F(\bar{x}^k)] - F^*\right),$$

with $c = \gamma n$, and then apply the theorem to show that $c\log(\frac{1}{\epsilon p})$ updates suffice to guarantee $F(\bar{x}^k) - F^* \leq \epsilon$ with probability $1-p$. □

To relate the solution quality of sub-problem (6) to the outer proximal iterations (5), we need to bound not only the function difference in primal but also the distance to the exact solution $y^{t+1} = \mathbf{prox}_{\eta_t g}(y^t)$ to the proximal update (5). To achieve this, we transfer the bound on $F(\bar{x}^k) - F^*$ to that on $\|y(\bar{x}^k) - y^{t+1}\|$.

Corollary 3. *To guarantee*

$$\|y(\bar{x}^k) - y^{t+1}\| \leq \epsilon_0 \tag{53}$$

with probability $1-p$, it suffices running RCD for

$$k \geq 2\gamma n \log\left(\sqrt{\frac{2\eta_t(F(\bar{x}^0) - F^*)}{p}}\frac{1}{\epsilon_0}\right)$$

iterations.

Proof. Given the primal iterate $\bar{x}^k$, the corresponding dual iterate $y(\bar{x}^k)$ is maintained through (7), written as

$$y(\bar{x}^k) = \eta_t(\bar{A}\bar{x}^k - b) + y^t.$$

Therefore,

$$\|y(\bar{x}^k) - y^{t+1}\| = \|\bar{A}(\bar{x}^k - \bar{x}^k_{\mathcal{S}})\|. \tag{54}$$

To bound (54) by the function value difference, note that

$$F(\bar{x}^k) - F(\bar{x}^k_{\mathcal{S}}) = \langle \nabla F(\bar{x}^k_{\mathcal{S}}), \bar{x}^k - \bar{x}^k_{\mathcal{S}}\rangle + \frac{1}{2}(\bar{x}^k - \bar{x}^k_{\mathcal{S}})^T \nabla^2 F(\bar{x}^k_{\mathcal{S}})(\bar{x}^k - \bar{x}^k_{\mathcal{S}})$$

and since

$$0 = h(\bar{x}^k) - h(\bar{x}^k_{\mathcal{S}}) \geq \langle \rho^*, \bar{x}^k - \bar{x}^k_{\mathcal{S}}\rangle$$

($\rho^* \in \partial h(\bar{x}^k_{\mathcal{S}})$ is the unique subgradient at optimal defined in (35)), together we get

$$F(\bar{x}^k) - F(\bar{x}^k_{\mathcal{S}}) \geq \frac{1}{2}(\bar{x}^k - \bar{x}^k_{\mathcal{S}})^T \nabla^2 F(\bar{x}^k_{\mathcal{S}})(\bar{x}^k - \bar{x}^k_{\mathcal{S}}) = \frac{\eta_t}{2}\|\bar{A}(\bar{x}^k - \bar{x}^k_{\mathcal{S}})\|^2,$$

which, combined with (54), leads to the bound

$$\|y(\bar{x}^t) - y^{t+1}\| \leq \sqrt{2\eta_t \left(F(\bar{x}^k) - F(\bar{x}^k_{\mathcal{S}})\right)}.$$

Therefore, to guarantee $\|y(\bar{x}^k) - y^{t+1}\| \leq \epsilon_0$, it suffices to have $F(\bar{x}^k) - F(\bar{x}^k_{\mathcal{S}}) \leq \frac{\epsilon_0^2}{2\eta_t}$, which can be achieved with high probability $1-p$ by running RCD Algorithm 2 for

$$k \geq \gamma n \log\left(\frac{2\eta_t(F(\bar{x}^0) - F^*)}{\epsilon_0^2 p}\right) = 2\gamma n \log\left(\sqrt{\frac{2\eta_t(F(\bar{x}^0) - F^*)}{p}}\frac{1}{\epsilon_0}\right) \tag{55}$$

according to Corollary 2. □

6.3 Overall Iteration Complexity of AL-CD

This section combines the linear convergence of Augmented Lagrangian (AL) and Coordinate Descent (CD) to give an overall iteration complexity that bounds the number of RCD updates required for AL-CD to find an LP solution of ϵ precision.

The first key lemma bounds the approximation error incurred in the outer iterates when solving inner sub-problems in an inexact fashion.

Lemma 5 (Inexact Proximal Map)**.** *Suppose, for a given dual iterate y^t, each sub-problem* (6) *is solved inexactly s.t.*

$$\|\hat{y}^{t+1} - \mathbf{prox}_{\eta_t g}(y^t)\| \leq \epsilon_0. \tag{56}$$

Then let $\{\hat{y}^t\}_{t=1}^{\infty}$ be the sequence of iterates produced by inexact proximal updates and $\{y^t\}_{t=1}^{\infty}$ as that generated by exact updates. After t iterations, we have

$$\|\hat{y}^t - y^t\| \leq t\epsilon_0. \tag{57}$$

Proof. By the non-expansiveness of proximal operation,

$$\begin{aligned}\|\hat{y}^{t+1} - y^{t+1}\| &\leq \|\hat{y}^{t+1} - \mathbf{prox}_{\eta_t g}(\hat{y}^t)\| + \|\mathbf{prox}_{\eta_t g}(\hat{y}^t) - y^{t+1}\| \\ &\leq \epsilon_0 + \|\mathbf{prox}_{\eta_t g}(\hat{y}^t) - \mathbf{prox}_{\eta_t g}(y^t)\| \\ &\leq \epsilon_0 + \|\hat{y}^t - y^t\|.\end{aligned}$$

Recursively applying the above inequality leads to the conclusion (57). □

Note the above implies that, if an exact AL method performs t outer iterations to achieve ϵ-precise solution, then solving each subproblem with precision $\epsilon_0 = \epsilon/t$ makes only an additional ϵ approximation error in the overall result. This insight turns out to give the following main theorem.

Theorem 6 (Iteration Complexity)**.** *Denote $\{\hat{y}^t\}_{t=1}^{\infty}$ as the sequence of iterates obtained from inexact dual proximal updates and $\{y^t\}_{t=1}^{\infty}$ as that generated by exact updates. To guarantee $\|\hat{y}^t - \hat{y}^t_{S_*}\| \leq 2\epsilon$ with probability $1-p$, it suffices to run Algorithm 1 for*

$$T = (1 + \frac{1}{\alpha}) \log\left(\frac{LR}{\epsilon}\right) \tag{58}$$

outer iterations with $\eta_t = (1+\alpha)L$, and solve each sub-problem (6) *by running Algorithm 2 for*

$$k \geq 2\gamma n \left(\log\left(\frac{\omega}{\epsilon}\right) + \frac{3}{2}\log\left((1 + \frac{1}{\alpha})\log\frac{LR}{\epsilon}\right)\right) \tag{59}$$

inner iterations, where $\omega = \sqrt{\frac{2(1+\alpha)L(F^0 - F^*)}{p}}$, $R = \|\Delta^0\|$.

Proof. Since

$$\|\hat{y}^t - \hat{y}^t_{S_*}\| \leq \|\hat{y}^t - y^t_{S_*}\| \leq \|y^t - y^t_{S_*}\| + \|\hat{y}^t - y^t\|,$$

to guarantee $\|\hat{y}^t - \hat{y}^t_{S_*}\| \leq 2\epsilon$, it suffices to let $\|y^t - y^t_{S_*}\| < \epsilon$ and $\|\hat{y}^t - y^t\| < \epsilon$, where the former can be guaranteed as long as the number of outer iterations

$$T = (1 + \frac{1}{\alpha}) \log \left(\frac{L\|\Delta^0\|}{\epsilon} \right)$$

by Corollary 1. To ensure $\|\hat{y}^t - y^t\| < \epsilon$, according to Lemma 5, it suffices to solve each proximal subproblem to precision $\epsilon_0 = \epsilon/T$. To guarantee that the T subproblems are all solved to precision $\epsilon_0 = \epsilon/T$ with probability $1 - p$, we require each of them to hold with probability $1 - p/T$ independently, which can be guaranteed by running RCD on each subproblem for

$$k \geq 2\gamma n \log \left(\sqrt{\frac{2(1+\alpha)L(F_t(\bar{x}^0) - F_t^*)}{p}} \frac{T^{3/2}}{\epsilon} \right)$$

inner iterations (Corollary 3), where $F_t(\bar{x})$ denotes the objective of t-th subproblem. To remove the dependency of k on t, we bound the term $F_t(\bar{x}^0) - F_t^*$ by $F^0 - F^*$, where $F^* \leq F_t^*$ is a lower bound on the optimal function value of subproblem, which exists as long as the original LP is bounded below, and $F^0 \geq F_t(\bar{x}^0)$ is a bound on the initial function value of each sub-problem, which exists as long as each subproblem is initialized by the solution of previous subproblem, and each subproblem is solved with precision $\epsilon_0 = \epsilon/T$. Then to guarantee the above inequality, it suffices to have

$$k \geq 2\gamma n \left(\log \left(\frac{\omega}{\epsilon} \right) + \frac{3}{2} \log \left((1 + \frac{1}{\alpha}) \log \frac{LR}{\epsilon} \right) \right),$$

where $\omega = \sqrt{\frac{2(1+\alpha)L(F^0 - F^*)}{p}}$, $R = \|\Delta^0\|$. □

7 Appendix-B. Data Statistics

All data sets for experiments of L1-regularized SVM can be found in the LIBSVM dataset repository, where the data set *cod-rna.rf* uses $D = 5000$ Fourier Random Features [32, 33] to approximate the effect of Gaussian RBF kernel. We choose $\lambda = 1$ for all L1-regularized SVM problems except for *cod-rna.rf* we use $\lambda = 10$ to increase the primal sparsity. The data sets *textmine*, *E2006* for Sparse Inverse Covariance Estimation are also obtained from LIBSVM dataset repository, while the *micromass*, *dorothea* are taken from UCI Machine Learning repository. For Sparse Inverse Covariance Estimation, we excluded features of frequency less than 10. The *ocr* data set is taken from `http://ai.stanford.edu/~btaskar/ocr/`. For Non-negative Matrix Factorization, we set the matrix approximation tolerance to be 0.01 times number of samples.

Table 4: Data Statistics for L1-SVM

Data set	#Samples	#Features	NNZ	#class	n_b	m_I
rcv	15564	47236	1028284	53	4833738	778200
news	15935	62061	1272569	20	2498415	302765
sector	6412	55197	1045412	105	11597992	666848
mnist	60000	780	8994156	10	75620	540000
cod-rna.rf	59535	5000	297675000	2	69537	59535
viecle	78823	100	7882300	3	79429	157646
real-sim	72309	20958	3709083	2	114227	72309

Table 5: Data Statistics for Sparse Inverse Covariance Selection

Data set	#Samples	#Features	NNZ	n_b	m_I	m_E	n_f
textmine	21519	30438	2283179	60876	60876	43038	43038
E2006	16087	27917	19640157	55834	55834	32174	32174
dorothea	800	23616	463088	47232	47232	1600	1600

Table 6: Data Statistics for Convex Nonnegative Matrix Factorization

Data set	#Samples	#Features	NNZ	n_b	m_I
micromass	931	1,299	106,292	2,896,770	4,107,438
ocr	52,152	127	1,466,486	6,639,433	13,262,864