7 Framework of Block CD Methods
See Algorithm 1. A one-variable CD by [2] for linear SVM is in Algorithm 2.

8 Details of Two-variable CD for Dual SVM without the Bias Term

8.1 Solving Two-variable Sub-problems (3.16)
For easy understanding, we rewrite (3.16) to a more general two-variable optimization problem:
\[
\min_{d_1, d_2} \frac{1}{2} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12} & Q_{22} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} + [p_1 \ p_2] \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}
\]
(8.24)

subject to \( L_1 \leq d_1 \leq U_1, \ L_2 \leq d_2 \leq U_2, \)
where \( L_1, U_1, L_2, U_2 \in R. \) The sub-problem (8.24) is the same as the one solved in [4], which studies two-variable CD for kernel SVM. We briefly describe their solution procedure before ours. They begin with considering (8.24) without constraints. With (3.15), the solution is easily seen as
\[
(8.25) \quad d_1^* = \frac{-Q_{22}p_1 + Q_{12}p_2}{Q_{11}Q_{22} - Q_{12}^2}, \quad d_2^* = \frac{-Q_{11}p_2 + Q_{12}p_1}{Q_{11}Q_{22} - Q_{12}^2}.
\]
Let the objective function of (8.24) be \( \hat{f}(d_1, d_2). \) If \( (d_1^*, d_2^*) \) is infeasible with \( d_1^* > U_1 \) and \( d_2^* \in [L_2, U_2], \) an optimal solution must be on the line of \( d_1 = U_1. \) A conceptual proof is in Figure 5a: if a solution \( \hat{d} \) is not on this line, then the line segment connecting \( \hat{d} \) and \( d^* \) leads to a point on \( d_1 = U_1 \) with a smaller function value because of the strict convexity of the function \( \hat{f}(d_1, d_2). \) Thus by fixing \( d_1 = U_1 \) one can solve a one-variable optimization problem to get the optimal solution \( \hat{d}_2. \) That is,
\[
\hat{d}_1 = P[d_1^*], \quad \hat{d}_2 = \arg \min_{d_2 \in [L_2, U_2]} \hat{f}(\hat{d}_1, d_2),
\]
where
\[
P[d_i] = \min(U_i, \max(L_i, d_i)), \quad \forall i = 1, 2,
\]
is a projection operation. However, if
\[
(8.26) \quad d_1^* > U_1, \quad d_2^* > U_2,
\]
the above argument can only imply that the solution must be on either \( d_1 = U_1 \) or \( d_2 = U_2; \) see the illustration in Figure 5b. Thus [4] propose solving two one-dimensional problems where one is by fixing \( d_1 = U_1 \) and the other is by fixing \( d_2 = U_2. \) Then they compare two objective values to decide the solution.

Algorithm 1 A framework of block CD methods
1: Let \( \alpha \) be a feasible point
2: while \( \alpha \) is not optimal do
3: \quad Select a working set \( B \)
4: \quad Solve the sub-problem (2.5)
5: \quad Update \( \alpha \) by \( \alpha_B \leftarrow \alpha_B + d_B \)
6: end while

Algorithm 2 A one-variable CD by [2] for linear SVM
1: Input: Specify a feasible \( \alpha \)
2: calculate \( u = \sum_j y_j \alpha_j x_j \)
3: while \( \alpha \) is not optimal do
4: \quad Obtain the permuted indices \( \{\pi(1), \pi(2), \ldots, \pi(l)\} \)
5: \quad for \( j = 1, \ldots, l \) do
6: \quad \quad \( i \leftarrow \pi(j) \)
7: \quad \quad \quad \( G \leftarrow \begin{cases} y_i u^T x_i & (\text{l1 loss}) \\ y_i u^T x_i - 1 + \frac{\alpha}{2c} & (\text{l2 loss}) \end{cases} \)
8: \quad \quad \quad \alpha_i \leftarrow \alpha_i + d \)
9: \quad \quad \( u \leftarrow u + d y_i x_i \)
10: \quad end for
11: end while
12: \( w \leftarrow u \)
13: output: \( (w, \alpha) \) as approximate primal and dual solutions.
If \( d^*_1 \geq u_1 \) and \( d^*_2 \in \{l_2, u_2\} \), then the solution is on \( d_1 = u_1 \).

(a) If \( d^*_1 \geq u_1 \) and \( d^*_2 \geq u_2 \), the solution is on either \( d_2 = u_2 \) or \( d_2 = u_2 \).

Figure 5: Illustrations of different situations of \( d^* \), the solution without constraints.

\[
\begin{align*}
\n d_2 &= u_2 \\
 d_2 &= l_2 \\
 d_2 &= l_2 \\
 d_1 &= l_1 \\
 d_1 &= u_1 \\
\end{align*}
\]

(b) If \( d^*_1 \geq u_1 \) and \( d^*_2 \geq u_2 \), the solution is on \( d_1 = u_1 \).

Figure 6: We can check the optimality condition at the point \( (P[d^*_1], P[d^*_2]) = (U_1, U_2) \) to decide which line the optimal solution is at.

However, this implementation is slightly complicated. From Figure 1, eight out-of-boundary cases must be considered. Further for the situation in Figure 1b, it would be better if we solve one rather than two one-variable sub-problems.

To have a simple procedure, we notice that for the situation in Equation (8.26) and have

\[
(P[d^*_1], P[d^*_2]) = (U_1, U_2).
\]

We then consider two cases. For the first one in Figure 6a, the optimal solution of (8.24) is on the line

\[
d_2 = U_2 \quad \text{and satisfies } \nabla_2 f(P[d^*_1], P[d^*_2]) \leq 0,
\]

while for the second, it is on

\[
d_2 = U_2 \quad \text{and satisfies } \nabla_1 f(P[d^*_1], P[d^*_2]) \leq 0.
\]

In other words, the optimality condition of \( d_1 \) has been satisfied. Therefore, the optimal solution must be on the line of \( d_1 = U_1 \). The case of Figure 6b can be formally extended to the following result.

**Theorem 8.1.** Assume \( \bar{d}_1 = P[d^*_1] \) is bounded and \( (P[d^*_1], P[d^*_2]) \) satisfies the optimality condition at \( d_1 \); that is,

\[
Q_{11} P[d^*_1] + Q_{12} P[d^*_2] + p_1 \begin{cases} 
\leq 0 & \text{if } P[d^*_1] = U_1, \\
\geq 0 & \text{if } P[d^*_1] = L_1.
\end{cases}
\]

Then \( (\bar{d}_1, \bar{d}_2) \) with

\[
\bar{d}_2 = \arg \min_{d_2 \in \{L_2, U_2\}} \frac{1}{2} [\bar{d}_1, d_2] [Q_{11} Q_{12} Q_{22}] [\bar{d}_1, d_2] + [p_1 \ p_2] \left[ \begin{array}{c} \bar{d}_1 \\ d_2 \end{array} \right]
\]

is an optimal solution of (8.24).

The remaining task is to have a clever setting so that we do not need to separately handle the eight cases, where \((d_1^*, d_2^*)\) is not in the feasible region.

While the strategy of checking (8.28) avoids solving two one-variable problems and comparing their objective values, it seems we still need to check all eight regions separately. Fortunately, we can handle Figure 5a and part of Figure 5b together because for the situation in Figure 5a, the following theorem shows that \((P[d^*_1], P[d^*_2])\) also satisfies the optimality condition of \( d_1 \).

**Theorem 8.2.** If

\[
d^*_1 \notin (L_1, U_1), \quad d^*_2 \in [L_2, U_2],
\]

then \((P[d^*_1], P[d^*_2])\) satisfies the optimality condition of \( d_1 \).

Therefore, by Theorem 8.1, we can cover a rather general situation by checking the optimality condition at \((P[d^*_1], P[d^*_2])\). Further, Theorem 8.1 can hold if the roles of \( d_1^* \) and \( d_2^* \) are swapped. To ensure that every \( d^* \) in the situation of Figure 5b is covered (i.e., Theorem 8.1 on either \( d^*_1 \) or \( d^*_2 \) is applicable), we need the following theorem.

**Theorem 8.3.** If

\[
d^*_1 \notin (L_1, U_1), \quad d^*_2 \notin (L_2, U_2),
\]

then \((P[d^*_1], P[d^*_2])\) satisfies either the optimality condition of \( d_1 \) or \( d_2 \).

All proofs are given in Section III of additional materials.
Based on the above theorems we can derive a simple procedure for solving (8.24). To begin, if \( d_1^* \notin (L_1, U_1) \) then we know that \( P[d_1^*] \) is bounded. We may apply Theorem 8.1 by checking if \( (P[d_1^*], P[d_2^*]) \) satisfies the optimality condition of \( d_1^* \). If it does, then (8.30) is an optimal solution.

There are two remaining situations:

\[
\begin{align*}
(8.31) & \quad d_1^* \in (L_1, U_1) \\
(8.32) & \quad \text{or} \\
& \quad d_1^* \notin (L_1, U_1) \text{ and } (P[d_1^*], P[d_2^*]) \text{ does not satisfy (8.29).}
\end{align*}
\]

For both situations, we argue that

\[
(8.33) \quad d_2 = P[d_2^*],
\]

\[
(8.34) \quad d_1^* = \min(U_1, \max(L_1, -\frac{Q_{12} \bar{d}_2 + p_1}{Q_{11}}))
\]

is an optimal solution. For (8.31), we can further consider two situations.

\[
(8.35) \quad d_2^* \in [L_2, U_2],
\]

\[
(8.36) \quad d_2^* \notin [L_2, U_2].
\]

If (8.35) holds, then

\[
P[d_1^*] = d_1^* \text{ and } P[d_2^*] = d_2^*
\]

are already an optimal solution. Though we do not need to apply (8.33), if we do, then \( d_1^* = d_1^* \) is obtained. On the other hand, if (8.36) holds, then from Theorem 8.2, \( (P[d_1^*], P[d_2^*]) \) satisfies the optimality condition of \( d_2^* \).

With the boundedness of \( P[d_2^*] \), we can apply Theorem 8.1 to have (8.33).

For the situation of (8.32), we argue that \( d_2^* \notin [L_2, U_2] \). Otherwise, \( d_2^* \in [L_2, U_2] \) and \( d_1^* \notin (L_1, U_1) \) imply from Theorem 8.2 that \( (P[d_1^*], P[d_2^*]) \) satisfies the optimality condition of \( d_1^* \), a contradiction to the condition in (8.32). Next, the property \( d_2^* \notin [L_2, U_2] \), (8.32) and Theorem 8.3 imply that \( (P[d_1^*], P[d_2^*]) \) must satisfy the optimality condition of \( d_2^* \).

A summary of the procedure is in Algorithm II of supplementary materials, in which we switch back to \( \alpha, \alpha_j \) from \( d_1, d_2 \) for practical implementations. Clearly, by using the gradient information rather than comparing objective values, the procedure becomes simple and short.

**8.2 Proof of Linear Convergence** We prove the linear convergence of the two-variable CD by using (3.19) for the working-set selection.

[5] consider two classes of problems (see their Assumptions 2.1 and 2.2) from the following convex optimization problem

\[
\min f(\alpha), \quad \alpha \in \mathcal{X},
\]

where \( f(\alpha) \) is proper convex, and \( \mathcal{X} \) is nonempty, closed, and convex. It is shown in Section 3.1 of [5] that the dual problem of both 1-loss SVM and 2-loss SVM are within the problems considered by them.\(^1\) [5] then analyze feasible-descent algorithms, where at the current the current and the next iterates satisfy

\[
\begin{align*}
(8.37) & \quad \alpha^{k+1} = [\alpha^k - \omega_k \nabla f(\alpha^k) + e^k]_+, \\
(8.38) & \quad \|e^k\| \leq \beta \|\alpha^k - \alpha^{k+1}\|, \\
(8.39) & \quad f(\alpha^k) - f(\alpha^{k+1}) \geq \gamma \|\alpha^k - \alpha^{k+1}\|^2,
\end{align*}
\]

where \( \inf \omega > 0, \beta > 0, \gamma > 0, \) and \( [\cdot]_+ \) is the following convex projection operator to the set \( \mathcal{X} \):

\[
(8.40) \quad [x]_+ = \arg\min_{y \in \mathcal{X}} \|x - y\|.
\]

For dual SVM,

\[
\mathcal{X} = [0, C_1] \times \cdots \times [0, C_l].
\]

From (8.40),

\[
[\alpha]_+ = [\max(\min(\alpha_1, C_1), 0), \ldots, \max(\min(\alpha_l, C_l), 0)]^T.
\]

Based on Theorem 2.8 of [5], we can prove the following linear-convergence result.

**Theorem 8.4.** The two-variable CD for dual 1-loss and 2-loss SVM has global linear convergence. To be specific, the method converges Q-linearly with

\[
f(\alpha^{k+1}) - f^* \leq \frac{\phi}{\phi + \gamma} (f(\alpha^k) - f^*), \quad \forall k \geq 0,
\]

where \( \kappa \) is the error bound constant,

\[
\phi = \left( \rho + \frac{1 + \beta}{\omega} \right) \left( 1 + \kappa \frac{1 + \beta}{\omega} \right),
\]

and \( \omega = \min_k (1, \inf \omega_k) \).

For 1-loss SVM, \( \kappa \) is derived in (7) of [5], and for 2-loss SVM,

\[
\kappa = 2(1 + \rho) \max_{i=1 \ldots l} C_i,
\]

where \( \rho = \lambda_{max}(Q) \), the largest eigenvalue of \( Q \), is the Lipschitz constant of \( \nabla f(\alpha) \).

**Proof.** To begin, we show that two-variable CD is a special case of the feasible-descent algorithms. The three conditions (8.37)-(8.39) are satisfied with

\[
\omega_k = 1, \quad \beta = 1 - \lambda + \sqrt{\rho}, \quad \gamma = \frac{\lambda}{2},
\]

where \( \lambda \) is the proximal term parameter in (3.13).

\(^1\)Note that for 1-loss SVM, they point out that some zero data instances must be removed first. This can be easily handled before solving the optimization problem.
We consider one iteration to be the collection of CD steps to go over all variables. From (3.19), we let
\[ B_1 = (\pi(1), \pi(2)), \ldots, B_l = (\pi(l - 1), \pi(l)) \]
be the working sets considered in one iteration. Let
\[ \alpha^{k+1,1}, \alpha^{k+1,2}, \ldots, \alpha^{k+1,l} = \alpha^{k+1} \]
be solutions updated after each CD step, and we consider
\[ \alpha^1 = \alpha^{1,1} = \alpha^{1,2} = \ldots = \alpha^{1,i}. \]
Because \( \alpha^{k,i} \) is not changed before we obtain \( \alpha^{k+1,i} \), \( d_B \) in (2.5) corresponds to \( \alpha^{k+1,i} - \alpha^{k,i} \). From the optimality condition of the sub-problem (3.13), we have for all \( i = 1, \ldots, l \),
\[
(\alpha^{k+1,i})^+ = \left[ \alpha^{k+1,i} - \nabla f(\alpha^{k+1,i}) - \lambda(\alpha^{k+1,i} - \alpha^{k,i}) \right]^+. 
\]
With
\[
\alpha^{k+1,i} = \alpha^{k+1,j}, \quad \forall j = 1, \ldots, \bar{l}, \] (8.41)
we can rewrite (8.41) as
\[
\alpha^{k+1,i} = \left[ \alpha^{k+1,i} - \nabla f(\alpha^{k+1,i}) - \lambda(\alpha^{k+1,i} - \alpha^{k,i}) \right]^+. 
\]
Next, we let
\[
\alpha^{k+1} = \left[ \alpha^k - \nabla f(\alpha^k) + e^k \right]^+, 
\]
where from (8.43)
\[
e^k = \alpha^{k+1,i} - \alpha^k + \nabla f(\alpha^k) - \nabla f(\alpha^{k+1,i}) - \lambda(\alpha^{k+1,i} - \alpha^k), \] (8.42)
where \( \lambda = \frac{1 + \rho}{\sigma} \), where \( \rho \) is the Lipschitz constant of \( \nabla f(\alpha) \), and \( f(\alpha) \) is \( \sigma \) strongly convex. For SVM we have
\[
\| \nabla f(\alpha_1) - \nabla f(\alpha_2) \| = \| Q(\alpha_1 - \alpha_2) \| \leq \lambda_{\text{max}} \| \alpha_1 - \alpha_2 \|, \]
where \( \rho = \lambda_{\text{max}} \) can be the Lipschitz constant. For the \( \sigma \) value, from (2.4),
\[
(\alpha_1 - \alpha_2)^T Q(\alpha_1 - \alpha_2) \geq \min_{i=1,\ldots,l} \left( \frac{1}{2C_i} \right) \| \alpha_1 - \alpha_2 \|^2. 
\]
Thus,
\[
\kappa = \frac{1 + \rho}{\sigma} = 2(1 + \lambda_{\text{max}}) \max_{i=1,\ldots,l} C_i. 
\]
\( \square \)

8.3 Shrinking Technique Because of bound constraints \( 0 \leq \alpha_i \leq C_i \), it is well developed in SVM literature that some bounded components can be tentatively removed in the optimization process. Then we solve smaller problems to reduce the running time, a strategy usually referred to as the shrinking technique [3]. Though several ways are available to implement the shrinking technique, we extend the one proposed by [2] to the two-variable situation. For a bound-constrained convex problem like (2.3), \( \alpha \) is optimal if and only if the
following projected gradient is zero.

\[ \nabla^P_i f(\alpha) = \begin{cases} 
\nabla_i f(\alpha) & \text{if } 0 < \alpha_i < C_i, \\
\min(0, \nabla_i f(\alpha)) & \text{if } \alpha_i = 0, \\
\max(0, \nabla_i f(\alpha)) & \text{if } \alpha_i = C_i.
\end{cases} \]

For the one-variable CD, let each cycle of updating all the remained variables be an “outer iteration.” Assume at the \((k-1)\)th outer iteration we have the following sequence of iterates.

\[ \alpha^{k-1,1}, \alpha^{k-1,2}, \ldots, \alpha^{k-1,l}, \]

where \(l\) is the number of remained variables at the beginning of the outer iteration. We further assume that at \(\alpha^{k-1,j}\), the index \(i_j\) is selected for possible update. \[2\] define the following two values to indicate the violation of the optimality condition.

\[ M^{k-1} = \max_j \nabla^P_{i_j} f(\alpha^{k-1,j}), \quad m^{k-1} = \min_j \nabla^P_{i_j} f(\alpha^{k-1,j}). \]

Then at each CD step of the next (i.e., the \(k\)th) outer iteration, before updating \(\alpha_{k,j}\) to \(\alpha_{k+1,j}\), the variable \(\alpha_{i,j}\) is shrunken if one of the following two conditions holds:

\[ \alpha_{k,j} = 0 \quad \text{and} \quad \nabla_{i,j} f(\alpha^{k,j}) > M^{k-1}, \]

\[ \alpha_{k,j} = C_i \quad \text{and} \quad \nabla_{i,j} f(\alpha^{k,j}) < m^{k-1}, \]

where

\[ M^{k-1} = \begin{cases} 
M^{k-1} & \text{if } M^{k-1} > 0, \\
\infty & \text{otherwise,} 
\end{cases} \]

\[ m^{k-1} = \begin{cases} 
m^{k-1} & \text{if } m^{k-1} < 0, \\
-\infty & \text{otherwise.} 
\end{cases} \]

In (8.46), \(M^{k-1}\) must be strictly positive, so \[2\] set it be \(\infty\) if \(M^{k-1} \leq 0\). The situation for \(m^{k-1}\) is similar. Details of one-variable CD with shrinking can be found in appendix of \[2\].

To extend the above setting to two-variable block CD, one issue is that we no longer have the concept of outer iterations. The reason is that from Section 3, a random working-set selection is practically more viable. Therefore, we can treat a fixed number of CD steps as an outer iteration in order to calculate the above \(M^{k-1}\) and \(m^{k-1}\) values. We choose \(l\) so that the frequency of refreshing \(M^{k-1}\) and \(m^{k-1}\) is similar to that of one-variable CD. Algorithm I of additional materials summarizes the two-variable block CD with the shrinking implementation.

9 Additional Discussion on Two-variable CD Methods for Linear SVM with a Bias Term

9.1 Solving the Sub-problem: Difference from SVM Without the Bias Term

Interestingly, though it is easy to derive a solution procedure for solving (4.20), a comparison shows that Algorithm II of additional materials for solving (8.24) is shorter in terms of the code length. One reason is that in Algorithm II, gradient information (or optimality condition) is used to avoid the exhaustive check of all out-of-boundary cases of \(\alpha_i\) or \(\alpha_j\). Further, for solving (4.20), we must separately handle the situations of \(y_i = y_j\) and \(y_i = -y_j\).

9.2 Difference Between Linear and Kernel Situations

We point out a difference in solving (4.20) between linear and kernel situations. For kernel, \[1\], as mentioned in Section 2.3, consider a greedy working set selection by using the gradient information, so their selected set satisfies

\[ -y_i y_j \nabla_i f(\alpha) + \nabla_j f(\alpha) \neq 0. \]

Thus, even if

\[ Q_{ii} - 2y_i y_j Q_{ij} + Q_{jj} = 0, \]

the minimum of (4.20)

\[ -y_i y_j \nabla_i f(\alpha) + \nabla_j f(\alpha) = 0, \]

then the minimum of (4.20) is

\[ d_j = 0. \]

Therefore, the selected pair is not useful to reduce the function value. We can conduct a simple check on (9.47) before solving the two-variable sub-problem.

References


