A Coordinate Gradient Descent Method for Linearly Constrained Smooth Optimization

Sangwoon Yun
Mathematics, National University of Singapore
Singapore

INFORMS Annual Meeting
November 5, 2007
(Joint work with Paul Tseng)
A COORDINATE GRADIENT DESCENT METHOD FOR LINEARLY CONSTRAINED SMOOTH OPTIMIZATION

Talk Outline

• SVM (Dual) Quadratic Program
• General Problem Model
• Coordinate Gradient Descent Method
• Convergence Results
• Complexity Bound
• Index Subset Selection
• Numerical Experience on SVM QP
• Extension
• Conclusions & Future Work
A Coordinate Gradient Descent Method for Linearly Constrained Smooth Optimization

**SVM (Dual) Quadratic Program**

\[
\begin{align*}
\min_{x} & \quad \frac{1}{2} x^T Q x - e^T x \\
\text{subject to} & \quad 0 \leq x_i \leq C, \quad i = 1, \ldots, n, \\
& \quad a^T x = 0,
\end{align*}
\]

where \( a \in \{-1, 1\}^n \), \( 0 < C \leq \infty \), \( e = [1, \ldots, 1]^T \), \( Q \in \mathbb{R}^{n \times n} \) is a sym. pos. semidef. with \( Q_{ij} = a_i a_j K(z_i, z_j) \), \( K: \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R} \) (“kernel function”), and \( z_i \in \mathbb{R}^p \) (“\( i \)th data point”), \( i = 1, \ldots, n \).

Popular choices of \( K \):

- **Linear kernel** \( K(z_i, z_j) = z_i^T z_j \)

- **Radial basis function kernel** \( K(z_i, z_j) = \exp(-\gamma \|z_i - z_j\|^2) \)

- **Sigmoid kernel** \( K(z_i, z_j) = \tanh(\gamma z_i^T z_j) \)

where \( \gamma \) is a constant.

\( Q \) is an \( n \times n \) fully dense matrix and even indefinite. (\( n \geq 5000 \))

Interior-point methods cannot be directly applied, except in the case of linear kernel.
Previous methods

Decomposition methods based on iterative block-coordinate descent have become popular for solving SVM QP.

- Joachims (98)
- Platt (99)
- Chang et al. (00)
- Keerthi et al. (00)
- Hush and Scovel (03)
- Palagi and Sciandrone (05)
- Fan et al. (05)

Decomposition methods use search directions of small support (i.e., few nonzeros) and achieve linear convergence under additional assumptions such as $Q$ being positive definite.
General Problem Model

\[
\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad f(x) \\
\text{s.t.} & \quad x \in X := \{x \mid l \leq x \leq u, \ Ax = b\},
\end{align*}
\]

\(f : \mathbb{R}^n \to \mathbb{R}\) is smooth.

\(A \in \mathbb{R}^{m \times n}, \ b \in \mathbb{R}^m, \) and \(l \leq u\) (possibly with \(-\infty\) or \(\infty\) components).

- For SVM QP, \(f\) is quadratic (possibly nonconvex) and \(m = 1\).
Coord. Gradient Descent Method

Descent direction.

For \( x \in X \), choose \( \mathcal{J}(\neq \emptyset) \subseteq \mathcal{N} = \{1, \ldots, n\} \) and \( H \succ 0_n \). Then solve

\[
\min_{x+d \in X, \ d_j = 0 \ \forall \ j \not\in \mathcal{J}} \{ \nabla f(x)^T d + \frac{1}{2} d^T H d \}.
\]

Let \( d_H(x; \mathcal{J}) \) and \( q_H(x; \mathcal{J}) \) be the opt. soln and obj. value of the direc. subprob.

Facts:

- \( q_H(x; \mathcal{N}) = 0 \iff x \in X \) is a stationary point of \( f \) over \( X \). \hspace{1cm} \text{stationarity}

- \( q_H(x; \mathcal{J}) \leq -\frac{1}{2} d^T H d \) where \( d = d_H(x; \mathcal{J}) \).
Choose $\alpha$: Armijo rule

Choose $\alpha$ to be the largest element of $\{\beta^k\}_{k=0,1,...}$ satisfying

$$f(x + \alpha d) - f(x) \leq \sigma \alpha q_H(x; \mathcal{J}) \quad (0 < \beta < 1, 0 < \sigma < 1).$$

For a QP, the minimization rule or the limited minimization rule can also be used.

Choose $\mathcal{J}$: Gauss-Southwell-$q$ rule

$$q_D(x; \mathcal{J}) \leq \upsilon q_D(x; \mathcal{N}),$$

Where $0 < \upsilon \leq 1$, $D > 0_n$ is diagonal.
Convergence Results

**Global convergence** If

- \( 0 < \lambda \leq \lambda_i(D), \lambda_i(H) \leq \bar{\lambda} \ \forall i, \)

- \( J \) is chosen by Gauss-Southwell-\( q \) rule,

- \( \alpha \) is chosen by Armijo rule,

then every cluster point of the \( x \)-sequence generated by CGD method is a stationary point of \( f \) over \( X \).
A Coordinate Gradient Descent Method for Linearly Constrained Smooth Optimization

Local convergence rate  If

- $0 < \lambda \leq \lambda_i(D), \lambda_i(H) \leq \bar{\lambda} \forall i,$
- $J$ is chosen by Gauss-Southwell-$q$ rule,
- $\alpha$ is chosen by Armijo rule,

in addition, if $f$ satisfies any of the following assumptions, then the $x$-sequence generated by CGD method converges at R-linear rate.

**C1** $f$ is strongly convex. $\nabla f$ is Lipschitz cont. on $X$

**C2** $f$ is (nonconvex) quadratic. (e.g., SVM QP)

**C3** $f(x) = g(Ex) + q^T x$, where $E \in \mathbb{R}^{m \times n}$, $q \in \mathbb{R}^n$, $g$ is strongly convex, $\nabla g$ is Lipschitz cont. on $\mathbb{R}^m$.

**C4** $f(x) = \max_{y \in Y} \{(Ex)^T y - g(y)\} + q^T x$, where $Y \subseteq \mathbb{R}^m$ is polyhedral, $E \in \mathbb{R}^{m \times n}$, $q \in \mathbb{R}^n$, $g$ is strongly convex, $\nabla g$ is Lipschitz cont. on $\mathbb{R}^m$. 

– Typeset by FontTeX –
**Notes:**

Proof of convergence rate uses a local error bound

- **Error Bound**

\[
\text{dist}(x, X^*) \leq \kappa \|d_I(x; \mathcal{N})\|_2 \quad \text{whenever} \quad \|d_I(x; \mathcal{N})\|_2 \leq \epsilon,
\]

for some \(\kappa > 0, \epsilon > 0\), where \(X^*\) denotes the set of stationary points of \(f\) over \(X\) and \(\text{dist}(x, X^*) = \min_{x^* \in X^*} \|x - x^*\|_2\).
Complexity Bound

• $0 < \lambda \leq \lambda_i(D), \lambda_i(H) \leq \bar{\lambda} \ \forall i$,

• $J$ is chosen by Gauss-Southwell-$q$ rule,

• $\alpha$ is chosen by Armijo rule,

in addition, if $f$ is convex with Lipschitz continuous gradient, then the number of iterations for achieving $\epsilon$-optimality is

$$O \left( \frac{Lr^0}{\nu \epsilon} + \max \left\{ 0, \frac{L}{\nu} \ln \left( \frac{e^0}{r^0} \right) \right\} \right),$$

where $r^0 = \max_{x \in X} \{ \text{dist}(x, X^*)^2 \mid f(x) \leq f(x^0) \}$, $e^0 = f(x^0) - \min_{x \in X} f(x)$, and $L$ is a Lipschitz constant.

The constant in $O(\cdot)$ depends on $\lambda$, $\bar{\lambda}$, $\sigma$, $\beta$.

When specialized to SVM QP, our complexity bound for achieving $\epsilon$-optimality compares favorably with existing bounds.
Index Subset Selection

**Elementary vector** (Rockafellar, 1969)

- For any \( d \in \mathbb{R}^n \), the support of \( d \) is \( \text{supp}(d) := \{ j \in \mathcal{N} \mid d_j \neq 0 \} \).

- A \( d' \) is *conformal* to \( d \) if \( \text{supp}(d') \subseteq \text{supp}(d) \) and \( d'_j d_j \geq 0 \ \forall \ j \in \mathcal{N} \).

- A nonzero \( d \) is an *elementary vector* of \( \text{Null}(A) \) if \( d \in \text{Null}(A) \) and there is no nonzero \( d' \in \text{Null}(A) \) that is conformal to \( d \) and \( \text{supp}(d') \neq \text{supp}(d) \).

- Each elementary vector \( d \) satisfies \( |\text{supp}(d)| \leq \text{rank}(A) + 1 \).
A Coordinate Gradient Descent Method for Linearly Constrained Smooth Optimization

Find $\mathcal{J}$ with $|\mathcal{J}| = 2$ in $O(n)$ oper. (SVM QP, $m = 1$)

- Step 1: Find $d_D(x; \mathcal{N})$ in $O(n)$ oper. by solving a cont. quad. knapsack problem:

$$
\begin{align*}
\min_{d} & \quad \frac{1}{2} d^T D d + \nabla f(x)^T d \\
\text{subject to} & \quad l \leq x + d \leq u, \\
& \quad Ad = 0,
\end{align*}
$$

Where $D \succ 0_n$ is diagonal.

- Step 2: Find a \textit{conformal realization} of $d_D(x; \mathcal{N})$:

$$
d_D(x; \mathcal{N}) = \sum_{i=1}^{r} d^i \text{ where } d^i \text{ is an elementary vector of } \text{Null}(A)
$$

and $r \leq n - 1$.

Choose $\mathcal{J} = \text{supp}(\overline{d}^i)$ where $\overline{i} \in \arg\min_{i \in \{1, \ldots, r\}} g^T d^i + \frac{1}{2} (d^i)^T D d^i$.

This finds a $\mathcal{J}$ satisfying $|\mathcal{J}| = 2$ and $q_D(x; \mathcal{J}) \leq \frac{1}{n-1} q_D(x; \mathcal{N})$ in $O(n)$ oper.
Numerical Experience on SVM QP

• Implement CGD method in Fortran.

• Choose $\mathcal{J}$ by Gauss-Southwell-$q$ rule with

$$D = \text{diag} \left[ \max\{Q_{jj}, 10^{-5}\} \right]_{j=1,...,n},$$

as described in previous slide.

• Our implementation of the CGD method has the form

$$x^{\text{new}} = x + d_Q(x; \mathcal{I}),$$

with $|\mathcal{J}| = 2$. This corresponds to the CGD method with $\alpha$ chosen by the minimization rule. (The choice of $H$ is actually immaterial here.)

• Compute $d_D(x, \mathcal{N})$ and $q_D(x; \mathcal{N})$ by using a linear-time Fortran code $\text{klvfo}$ provided by Krzysztof Kiwiel.
A Coordinate Gradient Descent Method for Linearly Constrained Smooth Optimization

- \( x^0 = 0 \): \( O(n) \) opers. to compute gradient \( Qx^0 - e \).
  (for general \( x^0 \), \( O(n^2) \) opers.)

- \( O(n) \) opers. per iteration to update gradient \( Qx - e \) since \( |\mathcal{J}| = 2 \).

- The CGD method is terminated when \( -q_D(x; \mathcal{N}) \leq 10^{-5} \).

- Additional refinements such as caching most recently used columns of \( Q \) and using supports of 3 elementary vectors for a conformal realization of \( d_D(x; \mathcal{N}) \) are used to speed up the method.

- Numerical tests on some large two-class data classification problems.

- Comparison with LIBSVM (version 2.83), which chooses \( J \) differently, but with the same cardinality of 2.
A COORDINATE GRADIENT DESCENT METHOD FOR LINEARLY CONSTRAINED SMOOTH OPTIMIZATION

Test results ($\gamma = 1/p$ :default values of LIBSVM)

<table>
<thead>
<tr>
<th>Data</th>
<th>$n/p$</th>
<th>$C$/kernel</th>
<th>LIBSVM</th>
<th>CGD-3pair</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>iter/obj/cpu</td>
<td>iter/obj/cpu</td>
</tr>
<tr>
<td>a7a</td>
<td>16100/122</td>
<td>1/lin</td>
<td>64108/-5699.253/1.3</td>
<td>56869/-5699.246/6.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/lin</td>
<td>71328/-56875.57/4.6</td>
<td>322000/-56873.58/32.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1/rbf</td>
<td>4109/-5899.071/1.3</td>
<td>4481/-5899.070/1.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/rbf</td>
<td>10385/-55195.29/1.4</td>
<td>16068/-55195.30/2.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1/sig</td>
<td>3941/-6095.529/1.7</td>
<td>4201/-6095.529/1.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/sig</td>
<td>9942/-57878.56/1.7</td>
<td>10890/-57878.57/1.8</td>
</tr>
<tr>
<td>ijcnn1</td>
<td>49990/22</td>
<td>1/lin</td>
<td>16404/-8590.158/3.0</td>
<td>20297/-8590.155/6.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/lin</td>
<td>155333/-85441.01/4.2</td>
<td>155274/-85441.00/46.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1/rbf</td>
<td>5713/-8148.187/4.6</td>
<td>6688/-8148.187/3.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/rbf</td>
<td>6415/-61036.54/3.5</td>
<td>12180/-61036.54/4.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1/sig</td>
<td>6796/-9156.916/7.0</td>
<td>6856/-9156.916/5.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/sig</td>
<td>10090/-8898.40/6.4</td>
<td>12420/-8898.39/6.5</td>
</tr>
<tr>
<td>w7a</td>
<td>24692/300</td>
<td>1/lin</td>
<td>66382/-765.4115/0.4</td>
<td>72444/-765.4116/8.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/lin</td>
<td>662877/-7008.306/1.1</td>
<td>493842/-7008.307/60.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1/rbf</td>
<td>1550/-1372.011/0.4</td>
<td>1783/-1372.010/0.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/rbf</td>
<td>4139/-10422.69/0.4</td>
<td>4491/-10422.70/0.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1/sig</td>
<td>1477/-1427.453/0.4</td>
<td>2020/-1427.455/0.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10/sig</td>
<td>2853/-11668.85/0.3</td>
<td>5520/-11668.86/0.9</td>
</tr>
</tbody>
</table>

- CGD-3pair is slower than LIBSVM when the linear kernel is used, due to the greater times spent in finding $d_D(x; \mathcal{N})$ and for updating the gradient.
- CGD-3pair is comparable to LIBSVM in speed and solution quality for nonlinear kernel.
Extension

In order to find sparse solution, a nonsmooth function $P$ is added in the objective function (e.g. $P(x) = \|x\|_1$).

**Linearly Constrained Nonsmooth Optimization**

$$\begin{align*}
\min_{x \in \mathbb{R}^n} \quad & f(x) + cP(x) \\
\text{s.t.} \quad & x \in X := \{x \mid l \leq x \leq u, Ax = b\}.
\end{align*}$$

$P : \mathbb{R}^n \to (-\infty, \infty]$ is proper, convex, lsc, and $P(x) = \sum_{j=1}^n P_j(x_j)$ ($x = (x_1, \ldots, x_n)^T$).

The CGD method can be extended to solve the linearly constrained nonsmooth optimization problem.
Conclusions & Future Work

1. The CGD method is the first globally convergent block-coordinate update method for general linearly constrained optimization.

2. It is implementable in $O(n)$ opers. per iteration when $f$ is quadratic and $m = 1$ and is suited for large scale problems with $n$ large and $m$ small.

3. For SVM QP, numerical results show that CGD method can be competitive with state-of-the-art SVM code on large data classification problems when a nonlinear kernel is used.

4. The CGD-3pair can be further speeded up by omitting infrequently updated components from computation (“shrinkage”), as is done in state-of-the-art SVM codes LIBSVM and SVM$^{light}$.

5. For large-scale applications such as $\nu$-SVM, $m = 2$. A conformal realization can be found in $O(n \log n)$ operations when $m = 2$. However, this can still be slow. Can this be improved to $O(n)$ operations?
Tseng, P. and Yun S., A coordinate gradient descent method for linearly constrained smooth optimization and support vector machines training. Tseng, P. and Yun S., A coordinate gradient descent method for constrained nonsmooth optimization and bi-level optimization. (PDF file available at http://www.math.washington.edu/~sangwoon/)
Support Vector Classification

- Training points: \( z_i \in \mathbb{R}^p, i = 1, \ldots, n \).

- Consider a simple case with two classes (linear separable case):

  Define a vector \( a \):

  \[
  a_i = \begin{cases} 
  1 & \text{if } z_i \text{ in class 1} \\
  -1 & \text{if } z_i \text{ in class 2}
  \end{cases}
  \]

- A hyperplane \( 0 = w^T z - b \) separates data with the maximal margin. Margin is the distance of the hyperplane to the nearest of the positive and negative points. Nearest points lie on the planes \( \pm 1 = w^T z - b \).
A COORDINATE GRADIENT DESCENT METHOD FOR LINEARLY CONSTRAINED SMOOTH OPTIMIZATION

Negative Examples

Positive Examples

Maximize distances to nearest points

Space of possible inputs
SVM Optimization Problem

• The (original) Optimization Problem

\[
\min_{w,b} \quad \frac{1}{2} \|w\|_2^2 \\
\text{s.t.} \quad a_i (w^T z_i - b) \geq 1, \quad i = 1, \ldots, n.
\]

• The Modified Optimization Problem (allows, but penalizes, the failure of a point to reach the correct margin, by Cortes and Vapnik, 1995)

\[
\min_{w,b,\xi} \quad \frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \quad a_i (w^T z_i - b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n.
\]
Caching and Other Choices of $\mathcal{J}$

Using $k1vfo$ and updating the gradient are the dominant computations.

- Cache the most recently used columns of $Q$, up to a user-specified limit $\maxCN$, when updating the gradient $Qx - e$.

- There exists at least one elementary vector in this realization whose support $\mathcal{J}$ satisfies

$$q_D(x; \mathcal{J}) \leq \frac{1}{n - 1} q_D(x; \mathcal{N}).$$

- From among all such $\mathcal{J}$, we find the best one (i.e., has the least $q_Q(x; \mathcal{J})$ value) and make this our choice for index subset.

- In addition, find from among all such $\mathcal{J}$ the second-best and third-best ones, if they exist. (In our tests, they always exist.)
• If the second-best one is disjoint from the best one, we make it the next index subset, and if the third-best one is disjoint from both the best and the second-best, we make it the second-next index subset.

• The procedure of selecting 3 (possible) pairs of the index subset is repeated at least once every 3 consecutive iterations.