A Coordinate Gradient Descent Method for Structured Nonsmooth Optimization

Sangwoon Yun
Mathematics, National University of Singapore

May 21, 2008
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION

Outline

I. Nonsmooth Separable Minimization

II. Linearly Constrained Smooth Minimization

III. Extensions
I. Nonsmooth Separable Minimization
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION

Outline

- Bound-constrained Optimization & \( \ell_1 \)-regularized Convex Minimization
- General Problem Model: Nonsmooth Separable Minimization
- Coordinate Gradient Descent Method
- Convergence Results
- Numerical Experience on \( \ell_1 \)-regularized Convex Minimization
Bound-constrained Optimization & $\ell_1$-regularized Convex Minimization

Bound-constrained optimization problem

$$\min_{l \leq x \leq u} f(x),$$

where $f : \mathbb{R}^N \to \mathbb{R}$ is smooth, $l \leq u$ (possibly with $-\infty$ or $\infty$ components).

Can be reformulated as the following unconstrained optimization problem:

$$\min_x f(x) + P(x),$$

where $P(x) = \begin{cases} 0 & \text{if } l \leq x \leq u \\ \infty & \text{else} \end{cases}$. 
\( \ell_1 \)-regularized convex minimization problem

1. \( \ell_1 \)-regularized linear least squares problem

Find \( x \) so that \( Ax - b \approx 0 \) and \( x \) has “few” nonzeros.

Formulate this as an unconstrained convex optimization problem:

\[
\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 + c\|x\|_1 \quad (c > 0)
\]

2. \( \ell_1 \)-regularized logistic regression problem

\[
\min_{w \in \mathbb{R}^{n-1}, v \in \mathbb{R}} \frac{1}{m} \sum_{i=1}^{m} \log(1 + \exp(-(w^T a_i + v b_i))) + c\|w\|_1,
\]

where \( a_i = b_i z_i \) and \((z_i, b_i) \in \mathbb{R}^{n-1} \times \{-1, 1\}, \ i = 1, ..., m \) are a given set of (observed or training) examples.
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION

General Problem Model: Nonsmooth Separable Optimization

\[
P \min_x F_c(x) := f(x) + cP(x) \quad (c \geq 0)
\]

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

\(P : \mathbb{R}^n \rightarrow (-\infty, \infty]\) is proper, convex, lsc, and \(P(x) = \sum_{j=1}^n P_j(x_j)\)

\((x = (x_1, \ldots, x_n)^T)\).

- \(P(x) = \begin{cases} 0 & \text{if } l \leq x \leq u \\ \infty & \text{else} \end{cases}\)

- \(P(x) = \|x\|_1\)
Previous methods

- Fukushima and Mine (81) proposed a proximal gradient descent method which computes a direction $\bar{d}$ as the solution of the subproblem

$$\min_{d} \nabla f(x)^T d + \frac{1}{2}\rho \|d\|^2 + cP(x + d) \quad (\rho > 0)$$

and showed local linear convergence to a stationary point $x^*$ under the assumption that $\nabla^2 f(x^*)$ is positive definite.

- Auslender (78)
- Mine and Fukushima (81)
- Kiwiel (86)
- Fukushima (91)

The above studies did not present numerical results.
Coord. Gradient Descent Method

Descent direction.

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

\[
\min_{d|d_j=0 \ \forall j \notin \mathcal{J}} \{ \nabla f(x)^T d + \frac{1}{2} d^T H d + cP(x + d) - cP(x) \}
\]

direc. subprob

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

Facts:

- \( d_H(x; \mathcal{N}) = 0 \iff F'_c(x; d) \geq 0 \ \forall d \in \mathbb{R}^n \). \quad \text{stationarity}

- \( H \) is diagonal \( \Rightarrow d_H(x; \mathcal{J}) = \sum_{j \in \mathcal{J}} d_H(x; j), q_H(x; \mathcal{J}) = \sum_{j \in \mathcal{J}} q_H(x; j) \). \quad \text{separab.}

- \( q_H(x; \mathcal{J}) \leq -\frac{1}{2} d^T H d \) \quad \text{where} \ d = d_H(x; \mathcal{J}).
This coord. grad. descent approach may be viewed as a hybrid of gradient-projection and coordinate descent. In particular,

- if $\mathcal{J} = \mathcal{N}$ and $P(x) = \begin{cases} 0 & \text{if } l \leq x \leq u \\ \infty & \text{else} \end{cases}$, then $d_H(x; \mathcal{N})$ is a scaled gradient-projection direction for bound-constrained optimization.

- if $f$ is quadratic and we choose $H = \nabla^2 f(x)$, then $d_H(x; \mathcal{J})$ is a (block) coordinate descent direction.

If $H$ is diagonal, then subproblems can be solved in parallel.

- If $P \equiv 0$, then $d_H(x)_j = -\nabla f(x)_j / H_{jj}$.

- If $P(x) = \begin{cases} 0 & \text{if } l \leq x \leq u \\ \infty & \text{else} \end{cases}$, then
  \[ d_H(x)_j = \text{median}\{l_j - x_j, -\nabla f(x)_j / H_{jj}, u_j - x_j\} \]

- If $P$ is the 1-norm, then
  \[ d_H(x)_j = -\text{median}\{(\nabla f(x)_j - c) / H_{jj}, x_j, (\nabla f(x)_j + c) / H_{jj}\} \]
**Steps**: Armijo rule

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

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

For the $\ell_1$-regularized linear least squares problem, the minimization rule

$$\alpha \in \arg \min \{F_c(x + td) \mid t \geq 0\}$$

or the limited minimization rule

$$\alpha \in \arg \min \{F_c(x + td) \mid 0 \leq t \leq s\},$$

where $0 < s < \infty$, can also be used.
Choose $\mathcal{J}$:

- Gauss-Seidel rule:
  
  $\mathcal{J}$ cycles through $\{1\}, \{2\}, \ldots, \{n\}$.

- Gauss-Southwell-$r$ rule:

  $$
  \|d_D(x; \mathcal{J})\|_\infty \geq \nu \|d_D(x; \mathcal{N})\|_\infty
  $$

  where $0 < \nu \leq 1$, $D > 0_n$ is diagonal (e.g., $D = \text{diag}(H)$).

- Gauss-Southwell-$q$ rule:

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

  Where $0 < \nu \leq 1$, $D > 0_n$ is diagonal (e.g., $D = \text{diag}(H)$).
Coordinate Descent Method

When $P \equiv 0$. Given $x \in \mathbb{R}^n$, Choose $i \in \mathcal{N}$. Update

$$x^{\text{new}} = \arg\min_{u | u_j = x_j \forall j \neq i} f(u).$$

Repeat until convergence.

- Gauss-Seidel rule: Choose $i$ cyclically, $1, 2, \ldots, n$, $1, 2, \ldots$
- Gauss-Southwell rule: Choose $i$ with $|\frac{\partial f}{\partial x_i}(x)|$ maximum.

Properties:

- If $f$ convex, then every cluster point of the $x$-sequence is a minimizer.
- If $f$ nonconvex, then G-Seidel can cycle (Powell ’73) but G-Southwell still converges.
- Convergence is possible when $P \neq 0$ (Tseng ’01).
Advantage of CGD

- CGD method is simple, highly parallelizable, and is suited for solving large-scale problems.

- CGD not only has cheaper iterations than exact coordinate descent, it also has stronger global convergence properties.
Global convergence  If

- $0 \prec \lambda I \preceq D$, $H \preceq \bar{\lambda} I$,

- $\mathcal{J}$ is chosen by G-Seidel, G-Southwell-$r$, G-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 $F_c$. 
Local convergence rate  If

- $0 < \lambda I \preceq D$, $H \preceq \bar{\lambda} I$,
- $J$ is chosen by G-Seidel or Gauss-Southwell-$q$ rule,
- $\alpha$ is chosen by Armijo rule,

in addition, if $P$ and $f$ satisfy 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 $\text{dom} P$.

**C2** $f$ is (nonconvex) quadratic. $P$ is polyhedral.

**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 $\mathbb{R}^m$. $P$ is polyhedral.
A Coordinate Gradient Descent Method for Structured Nonsmooth Optimization

\[ f(x) = \max_{y \in Y} \{(Ex)^T y - g(y)\} + q^T x, \]  
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 \( \mathbb{R}^m \). \( P \) is polyhedral.

Notes:

Proof of convergence rate uses a local Lipschitzian 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 \( F_c \) and \( \text{dist}(x, X^*) = \min_{x^* \in X^*} \|x - x^*\|_2 \).
A Coordinate Gradient Descent Method for Structured Nonsmooth Optimization

Numerical Experience on $\ell_1$-regularized Convex Minimization

1. $\ell_1$-regularized linear least squares problem (Compressed Sensing):

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|^2_2 + c\|x\|_1 \quad (c > 0)$$

- Implement CGD method in Matlab.
- Choose $H$ as a constant multiple of the identity matrix

\[ H = \theta I, \]

Initially, we set $\theta^{\text{init}} = \|Au\|_2^2$ where $u$ is a random unit vector. Then updated as follows:

\[ \theta^{\text{new}} = \begin{cases} 
\max\{\theta/\alpha, 1\} & \text{if } \alpha > 10 \\
\min\{\theta/\alpha, 1\} & \text{if } \alpha < 10^{-1} \\
\theta & \text{otherwise.}
\end{cases} \]
• Choose $\mathcal{J}$ by Gauss-Southwell-$r$ rule,

$$\mathcal{J} = \{ j \mid \|d_H(x; j)\|_2 \geq \nu \|d_H(x; i)\|_\infty \}. $$

Or by Gauss-Southwell-$q$ rule,

$$\mathcal{J} = \{ j \mid q_H(x; j) \leq \nu \min_i q_H(x; i) \}. $$

• Choose $\alpha$ by the minimization rule.

$$\alpha \in \arg\min \left\{ \frac{a_1}{2} t^2 - a_2 t + c \sum_{j \in \mathcal{J}} |x_j + td_j| + a_3 \mid t > 0 \right\},$$

where $a_1 = \|Ad\|_2^2$, $a_2 = (Ad)^T(b - Ax)$, $a_3 = \frac{1}{2}\|b - Ax\|_2^2$, and $d = d_H(x; \mathcal{J})$.

• The CGD method is terminated when

$$\|Hd_H(x; \mathcal{N})\|_\infty \leq 10^{-3}. $$
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION

- $A \in \mathbb{R}^{m \times n}$ is obtained by first filling it with independent samples of the standard Gaussian distribution and then orthonormalizing the rows $(m = 2048, n = 8192)$.

the original signal $x$ contains 320 randomly placed $\pm 1$ spikes.

the measurement $Ax$ is corrupted by the noise $\xi$, $b = Ax + \xi$, where $\xi$ is Gaussian white noise with variance $(0.01 \|Ax\|^2$).

- Comparison with l1-ls (Kim et al ’07, interior-point), GPSR-BB (Figueiredo, Nowak and Wright ’07, gradient projection), and FPC (Hale, Yin and Zhang ’07, fixed-point continuation).

- Require only matrix-vector mults. involving $A$ and $A^T$ (dominant computations) at each iteration.

Matrix-vector mults. involving $A^T$ cost $O(mn)$ opers. (same as the computational cost of other algorithms).
But matrix-vector mults. involving $A$ cost $O(m|\mathcal{J}|)$ opers. (only need to evaluate $Ad$).

Sorting is needed to find the stepsize, the cost is $O(|\mathcal{J}| \ln |\mathcal{J}|)$. 
To perform this comparison, first run l1-ls and then each of the others until each reaches the same objective value reached by l1-ls (10 random instances).

<table>
<thead>
<tr>
<th></th>
<th>l1-ls</th>
<th>CGD-GS-q</th>
<th>CGD-GS-r</th>
<th>GPSR-BB</th>
<th>FPC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>n = 8192, m = 2048, c = 0.05∥ATb∥∞</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean iterations</td>
<td>12</td>
<td>17</td>
<td>20</td>
<td>28</td>
<td>83</td>
</tr>
<tr>
<td>mean nnz(x)</td>
<td>846</td>
<td>402</td>
<td>442</td>
<td>534</td>
<td>574</td>
</tr>
<tr>
<td>mean CPU time</td>
<td>1.9e+01</td>
<td>8.3e-01</td>
<td>1.0e+00</td>
<td>2.7e+00</td>
<td>7.6e+00</td>
</tr>
<tr>
<td>mean error</td>
<td>1.4e-01</td>
<td>1.5e-01</td>
<td>1.5e-01</td>
<td>1.5e-01</td>
<td>1.5e-01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>l1-ls</th>
<th>CGD-GS-q</th>
<th>CGD-GS-r</th>
<th>GPSR-BB</th>
<th>FPC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>n = 8192, m = 2048, c = 0.01∥ATb∥∞</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean iterations</td>
<td>13</td>
<td>36</td>
<td>44</td>
<td>104</td>
<td>120</td>
</tr>
<tr>
<td>mean nnz(x)</td>
<td>1043</td>
<td>779</td>
<td>1110</td>
<td>968</td>
<td>954</td>
</tr>
<tr>
<td>mean CPU time</td>
<td>2.6e+01</td>
<td>1.7e+00</td>
<td>2.0e+00</td>
<td>9.0e+00</td>
<td>1.0e+01</td>
</tr>
<tr>
<td>mean error</td>
<td>3.4e-02</td>
<td>4.6e-02</td>
<td>4.6e-02</td>
<td>4.2e-02</td>
<td>3.6e-02</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>l1-ls</th>
<th>CGD-GS-q</th>
<th>CGD-GS-r</th>
<th>GPSR-BB</th>
<th>FPC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>n = 8192, m = 2048, c = 0.005∥ATb∥∞</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean iterations</td>
<td>13</td>
<td>63</td>
<td>73</td>
<td>1001</td>
<td>118</td>
</tr>
<tr>
<td>mean nnz(x)</td>
<td>1234</td>
<td>1383</td>
<td>1947</td>
<td>8178</td>
<td>1567</td>
</tr>
<tr>
<td>mean CPU time</td>
<td>2.9e+01</td>
<td>3.0e+00</td>
<td>3.5e+00</td>
<td>8.7e+01</td>
<td>1.0e+01</td>
</tr>
<tr>
<td>mean error</td>
<td>2.2e-02</td>
<td>3.4e-02</td>
<td>3.3e-02</td>
<td>7.1e-01</td>
<td>3.0e-02</td>
</tr>
</tbody>
</table>
A Coordinate Gradient Descent Method for Structured Nonsmooth Optimization

2. $\ell_1$-regularized logistic regression problem:

$$\min_{w \in \mathbb{R}^{n-1}, v \in \mathbb{R}} \frac{1}{m} \sum_{i=1}^{m} \log(1 + \exp(-(w^T a_i + v b_i))) + c \|w\|_1,$$

- Implement CGD method in Matlab.
- Choose $H$ as a diagonal Hessian approximation

$$H = \text{diag} \left[ \min \{ \max \{ \nabla^2 f(x)_{jj}, 10^{-10} \}, 10^{10} \} \right]_{j=1, \ldots, n},$$

where $x = ((w)^T, v)^T$.
- Choose $J$ by Gauss-Southwell-$r$ rule, or by Gauss-Southwell-$q$ rule.
- Choose $\alpha$ by the Armijo rule.
The CGD method is terminated when
\[ \|H d_H(x; \mathcal{N})\|_\infty \leq 10^{-6}. \]

Numerical tests on some large two-class data classification problems (sparse) and on randomly generated problems (dense).

Comparison with l1-logreg (Koh, Kim and Boyd ’07, interior-point) and SpaRSA (Wright, Nowak and Figueiredo ’07, iterative method).
Test Results

- To perform this comparison, first run l1-logreg and then each of the others until each reaches the same objective value reached by l1-logreg.

<table>
<thead>
<tr>
<th></th>
<th>l1-logreg ((10^{-4}))</th>
<th>CGD-GS-q</th>
<th>CGD-GS-r</th>
<th>SpaRSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>leu iterations</td>
<td>25</td>
<td>92</td>
<td>146</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td>3.14477e-02</td>
<td>3.14458e-02</td>
<td>3.14212e-02</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.1e+00</td>
<td>6.6e-01</td>
<td>1.0e+00</td>
<td></td>
</tr>
<tr>
<td>n = 7130, (m = 38), (\mu = 0.01\mu_{max})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rcv1 iterations</td>
<td>28</td>
<td>118</td>
<td>105</td>
<td>191</td>
</tr>
<tr>
<td></td>
<td>2.12420e-01</td>
<td>2.12420e-01</td>
<td>2.12420e-01</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.1e+01</td>
<td>7.5e+00</td>
<td>6.9e+00</td>
<td>7.5e+00</td>
</tr>
<tr>
<td>n = 47237, (m = 20242), (\mu = 0.01\mu_{max})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>real-sim iterations</td>
<td>22</td>
<td>71</td>
<td>72</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>2.14289e-01</td>
<td>2.14289e-01</td>
<td>2.14287e-01</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.4e+01</td>
<td>7.5e+00</td>
<td>7.9e+00</td>
<td>7.6e+00</td>
</tr>
<tr>
<td>n = 20959, (m = 72309), (\mu = 0.01\mu_{max})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Randomly generated problem (Features of positive (negative) examples are independent and identically distributed, dense).

<table>
<thead>
<tr>
<th></th>
<th>l1-logreg $10^{-4}$</th>
<th>CGD-GS-q $10^{-6}$</th>
<th>CGD-GS-r $10^{-6}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 random</td>
<td>$n = 10001, m = 1000, c = 0.01c_{\text{max}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean iterations</td>
<td>20</td>
<td>230</td>
<td>260</td>
</tr>
<tr>
<td>mean CPU time</td>
<td>2.4e+02</td>
<td>1.2e+01</td>
<td>1.4e+01</td>
</tr>
<tr>
<td></td>
<td>$n = 1001, m = 100, c = 0.01c_{\text{max}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean iterations</td>
<td>17</td>
<td>187</td>
<td>220</td>
</tr>
<tr>
<td>mean CPU time</td>
<td>1.9e-01</td>
<td>2.4e-01</td>
<td>2.9e-01</td>
</tr>
<tr>
<td></td>
<td>$n = 1001, m = 10000, c = 0.01c_{\text{max}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean iterations</td>
<td>16</td>
<td>82</td>
<td>88</td>
</tr>
<tr>
<td>mean CPU time</td>
<td>2.3e+02</td>
<td>4.8e+00</td>
<td>5.2e+00</td>
</tr>
<tr>
<td></td>
<td>$n = 101, m = 1000, c = 0.01c_{\text{max}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean iterations</td>
<td>14</td>
<td>60</td>
<td>66</td>
</tr>
<tr>
<td>mean CPU time</td>
<td>1.9e-01</td>
<td>6.8e-02</td>
<td>9.0e-02</td>
</tr>
</tbody>
</table>

• The computational cost for the search direction of l1-logreg is $O(\min(n - 1, m)^2 \max(n - 1, m))$ opers. per iteration. In contrast, the computational cost of CGD is $O(mn)$ opers. per iteration.
II. Linearly Constrained Smooth Minimization
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION

Outline

• Support Vector Machine (Primal and Dual Optimization Problem)

• General Problem Model: Linearly Constrained Smooth Minimization

• Coordinate Gradient Descent Method

• Convergence Results

• Complexity Bound

• Index Subset Selection

• Numerical Experience on SVM QP
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION

Support Vector Machines

Support Vector Classification

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

- 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 STRUCTURED NONSMOOTH OPTIMIZATION

Positive Examples

Maximize distances to nearest points

Negative Examples

Space of possible inputs
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSmoOTH OPTIMIZATION

SVM Optimization Problem

• The (original) Optimization Problem

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

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

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2_2 + C \sum_{i=1}^{n} \xi_i
\]
\[
\text{s.t. } a_i \left( w^T z_i - b \right) \geq 1 - \xi_i, \; \xi_i \geq 0, \; i = 1, \ldots, n.
\]
**A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION**

**SVM (Dual) Optimization Problem (Convex 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 \( K(z_i, z_j) = \exp(-\gamma \|z_i - z_j\|_2^2) \)
- 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.
A Coordinate Gradient Descent Method for Structured Nonsmooth Optimization

General Problem Model: Linearly Constrained Smooth Minimization

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

\( f : \mathbb{R}^n \rightarrow \mathbb{R} \) 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

\[
\min_{x+d \in X, \ d_j=0 \ \forall \ j \notin \mathcal{J}} \{ \nabla f(x)^T d + \frac{1}{2} d^T H d \}.
\]

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

Facts:

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

- \( d_H(x; \mathcal{J}) = 0 \iff q_H(x; \mathcal{J}) = 0 \).
A Coordinate Gradient Descent Method for Structured Nonsmooth Optimization

**Stepsize:** Armijo rule

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

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

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

Choose $J$ (Gauss-Southwell-$q$ rule): $J$ satisfies

$$q_D(x; J) \leq \nu q_D(x; N),$$

Where $D \succ 0_n$ is diagonal, $0 < \nu \leq 1$. 
Convergence Results

Global Convergence  If

• $0 \prec \lambda I \preceq D$, $H \preceq \bar{\lambda} I$,

• $\mathcal{J}$ 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 $X$. 

Local Convergence Rate  If

- $0 < \lambda I \preceq D$, $H \preceq \bar{\lambda} I$,
- $J$ 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 \( \mathbb{R}^m \).

Notes:

Proof of convergence rate uses a local Lipschitzian error bound

- Error Bound

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

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

If $0 < \lambda I \preceq D$, $H \preceq \bar{\lambda} I$ and $f$ is convex with Lipschitz cont. grad., then the number of iterations for achieving $\epsilon$-optimality is

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

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

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 (Hush and Scovel ’03, List and Simon ’05).
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 $d'_jd_j \geq 0 \ \forall j \in \mathcal{N}$.

- 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 Structured Nonsmooth 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:

$$\begin{align*}
\min_d & \quad \frac{1}{2} d^T D d + g^T d \\
\text{s.t.} & \quad l \leq x + d \leq u, \\
& \quad A d = 0,
\end{align*}$$

Where $D \succ 0_n$ is diagonal, and $g = \nabla f(x)$.

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

$$d_D(x; \mathcal{N}) = \sum_{i=1}^{r} d^i$$

where $d^i$ is an elementary vector of $\text{Null}(A)$ and $r \leq n - 1$.

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

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
  \[
  x^{\text{new}} = x + d_\mathcal{J}(x; \mathcal{J}),
  \]
  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 klvfo provided by Krzysztof Kiwiel.
A Coordinate Gradient Descent Method for Structured Nonsmooth Optimization

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

- $O(n)$ opers. per iteration to update gradient $Qx - e$.

- 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 $\mathcal{J}$ differently, but with the same cardinality of 2.
## 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>713288/-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/-88898.40/6.4</td>
<td>12420/-88898.39/6.5</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>
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION

- 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.
III. Extensions
Outline

• Linearly Constrained Nonsmooth Separable Minimization

• Sparse Covariance Selection
The CGD method ($J$ is chosen by Gauss-Southwell-$q$ rule) can be extended to solve a linearly constrained nonsmooth separable minimization:

\[
\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad f(x) + cP(x) \\
\text{s.t.} & \quad Ax = b.
\end{align*}
\]

$P$ is not necessarily separable for the convergence (and the complexity bound).
Sparse Covariance Selection

d’Aspremont, Banerjee, El Ghaoui ’07
Zhaosong ’07
Friedman, Hastie, Tibshirani ’07

\[
\min_{X \in S^n_+} f(X) + c\|X\|_1
\]

\[f(X) = -\log \det(X) + \text{tr}(XS) \quad (S \in S^n_+ \text{ is empirical covariance matrix}),\]
\[\|X\|_1 = \sum_{i,j} |X_{ij}|, \quad c > 0.\]

- \(f\) is strictly convex, cont. diff. on its domain, \(O(n^3)\) opers. to evaluate. \(\|\cdot\|_1\) is convex, nonsmooth. In applications, \(n\) can exceed 6000.

The dual problem is a bound-constrained convex program:

\[
\min_{W \in S^n_+, \|W - S\|_\infty \leq c} -\log \det(W) - n
\]

\[\|Y\|_\infty = \max_{i,j} |Y_{ij}|.\]
• IP method requires $O(n^7 \log(1/\epsilon))$ oper. to find $\epsilon$-optimal soln. Impractical! Nesterov’s first-order smoothing method requires $O(n^4/\epsilon)$ oper. Zhaosong ’07.

• Use CD (GSeidel) to solve the dual problem, cycling thru columns $(i = 1, \ldots, n)$ of $W$. Each iteration reduces (via determinant property) to

$$\min_{y \in \mathbb{R}^{n-1}} y^T (W_{i-i}-)^{-1} y$$

s.t. $|y - S_{ij}| \leq c \quad i = 1, \ldots, n - 1$.

or (via duality)

$$\min_{\zeta \in \mathbb{R}^{n-1}} \frac{1}{2} \zeta^T W_{i-i-}\zeta - S_{i-i}^T \zeta + c \|\zeta\|_1.$$ 

Solve this using IP method Banerjee et al ’07 or CD (GSeidel) Friedman et al ’07.
Can apply CGD (GSeidel) to either primal or dual problem. When applied to the primal, each iteration entails

\[
\min_{u \in \mathbb{R}^n} \left\{ \text{tr}((-X^{-1} + S)D) + \frac{1}{2}u^T Hu + c\|X + D\|_1 \right\}_{D = u e_i^T + e_i u^T}.
\]

For diagonal \( H \succ 0_n \), the minimizing \( D \) has closed form. For each trial \( \alpha \) in the Armijo LS, \( \det(X + \alpha D) \) can be evaluated from \( \det(X) \) and \( X^{-1} \) in \( O(n^2) \) oper. Update \( X^{-1} \) in \( O(n^2) \) opers.

Similar application to the dual.

Global convergence, local convergence rate, complexity analysis.

Numerical tests (ongoing).
Conclusions

1. Numerical results shows the practical efficiency of the method for a large-scale $\ell_1$-regularized convex minimization.

2. The CGD method is the first globally convergent block-coordinate update method for general linearly constrained smooth minimization.

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. How would the CGD method perform on bound-constrained problems?

5. Can CGD method be extended to handle the following generalized problem:

$$\min_{x \in \mathbb{R}^n} f(x) + cP(x)$$
$$\text{s.t. } f_1(x) = 0, \ldots, f_m(x) = 0,$$
where $f_1(x), \ldots, f_m(x)$ are twice continuously differentiable functions.

6. Can CGD method be extended to a nonconvex nonsmooth regularization problem (e.g. \(\ell_p\)-regularization, \(0 < p < 1\))?
Thank you!

Tseng, P. and Yun S., A coordinate gradient descent method for nonsmooth separable minimization.
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 block-coordinate gradient descent method for linearly constrained nonsmooth separable minimization.
Yun S. and Toh K.-C., A coordinate gradient descent method for $\ell_1$-regularized convex minimization.
Bi-level Optimization
Problem Model

\[
\min_{x \in S_f} P(x)
\]

\(P : \mathbb{R}^n \to (-\infty, \infty] \) is a proper, convex, lsc function.

\(S_f\) denotes the set of stationary points of a smooth convex function \(f\) over 
\(\text{dom} P = \{x \mid P(x) < \infty\}\).

Sparse solution of an underdetermined system of linear equations
A COORDINATE GRADIENT DESCENT METHOD FOR STRUCTURED NONSMOOTH OPTIMIZATION

\[ P(x) = \|x\|_1. \]

\[ f(x) = \|Ax - b\|_2^2. \]
Algorithm (Regularization Strategy)

Choose $x^0 \in \text{dom} P$, $c^0 > 0$, $\epsilon^0 > 0$. For $k = 1, 2, \ldots$, generate $x^k$ from $x^{k-1}$ according to the iteration:

1. Choose $c^k > 0$ and $\epsilon^k > 0$.

2. Compute $x^k$ as a point satisfying

- $\|d_{D^k}(x^k; \mathcal{N})\|_2 \leq \epsilon^k$,
- $\|D^k d_{D^k}(x^k; \mathcal{N})\|_2 \leq \epsilon^k$
- $-(D^k x^k + \nabla f(x^k))^T d_{D^k}(x^k; \mathcal{N}) \leq \epsilon^k$

by applying the CGD method to the problem:

$$\min_x f(x) + cP(x) \quad (c \geq 0)$$
with $c = c^k$ and an initial point $x = x^{k-1}$. 
Convergence Results

Assume $P$ is level-bounded and \( \text{dom} P \cap S_f \neq \emptyset \).

If we choose \( c^k \) and \( \epsilon^k \) to tend to zero so that

\[
\lim_{k \to \infty} \frac{\epsilon^k}{c^k} = 0,
\]

then every cluster point of \( \{x^k\} \) is an optimal solution of the bi-level problem.