Coordinate Descent

Ryan Tibshirani
Convex Optimization 10-725
In $\mathbb{R}^n$, rough flop counts for basic operations are as follows

- Vector-vector operations: $n$ flops
- Matrix-vector multiplication: $n^2$ flops
- Matrix-matrix multiplication: $n^3$ flops
- Linear system solve: $n^3$ flops

Operations with **banded or sparse matrices** are much cheaper

Two classes of approaches for linear system solvers:

- **Direct**: QR decomposition, Cholesky decomposition
- **Indirect**: Jacobi, Gauss-Seidel, gradient descent, conjugate gradient

Rough rule of thumb: if problem fits easily in memory, go direct, else go indirect
Outline

Today:

• Coordinate descent
• Examples
• Implementation tricks
• Graphical lasso
• Screening rules
Coordinatewise optimality

We’ve seen some pretty sophisticated methods thus far

We now focus on a very simple technique that can be surprisingly efficient, scalable: coordinate descent, or more appropriately called coordinatewise minimization

Q: Given convex, differentiable $f : \mathbb{R}^n \rightarrow \mathbb{R}$, if we are at a point $x$ such that $f(x)$ is minimized along each coordinate axis, then have we found a global minimizer?

That is, does $f(x + \delta e_i) \geq f(x)$ for all $\delta, i \Rightarrow f(x) = \min_z f(z)$?

(Here $e_i = (0, \ldots, 1, \ldots, 0) \in \mathbb{R}^n$ is the $i$th standard basis vector)
A: Yes! Proof:

$$0 = \nabla f(x) = \left( \frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x) \right)$$

Q: Same question, but now for $f$ convex, and not differentiable?
A: No! Look at the above counterexample

Q: Same, now $f(x) = g(x) + \sum_{i=1}^{n} h_i(x_i)$, with $g$ convex, smooth, and each $h_i$ convex? (Here the nonsmooth part is called separable)
A: Yes! Proof: using convexity of $g$ and subgradient optimality

\[ f(y) - f(x) = g(x) - g(y) + \sum_{i=1}^{n} [h_i(y_i) - h_i(x_i)] \]

\[ \geq \sum_{i=1}^{n} \left[ \nabla_i g(x)(y_i - x_i) + h_i(y_i) - h_i(x_i) \right] \geq 0 \]
Coordinate descent

This suggests that for the problem

$$\min_x f(x)$$

where $f(x) = g(x) + \sum_{i=1}^n h_i(x_i)$, with $g$ convex and differentiable and each $h_i$ convex, we can use coordinate descent: let $x^{(0)} \in \mathbb{R}^n$, and repeat

$$x_i^{(k)} = \arg\min_{x_i} f(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)})$$

for $i = 1, \ldots, n$

for $k = 1, 2, 3, \ldots$

Important note: we always use most recent information possible
Tseng (2001) showed that for such $f$ (provided $f$ is continuous on compact set $\{x : f(x) \leq f(x^{(0)})\}$ and $f$ attains its minimum), any limit point of $x^{(k)}$, $k = 1, 2, 3, \ldots$ is a minimizer of $f$\(^1\)

Notes:

- Order of cycle through coordinates is arbitrary, can use any permutation of $\{1, 2, \ldots, n\}$
- Can everywhere replace individual coordinates with blocks of coordinates
- “One-at-a-time” update scheme is critical, and “all-at-once” scheme does not necessarily converge
- The analogy for solving linear systems: Gauss-Seidel versus Jacobi method

\(^1\)Using basic real analysis, we know $x^{(k)}$ has subsequence converging to $x^*$ (Bolzano-Weierstrass), and $f(x^{(k)})$ converges to $f^*$ (monotone convergence)
Example: linear regression

Given \( y \in \mathbb{R}^n \), and \( X \in \mathbb{R}^{n \times p} \) with columns \( X_1, \ldots, X_p \), consider the linear regression problem:

\[
\min_{\beta} \frac{1}{2} \| y - X \beta \|_2^2
\]

Minimizing over \( \beta_i \), with all \( \beta_j, j \neq i \) fixed:

\[
0 = \nabla_i f(\beta) = X_i^T (X \beta - y) = X_i^T (X_i \beta_i + X_{-i} \beta_{-i} - y)
\]

i.e., we take

\[
\beta_i = \frac{X_i^T (y - X_{-i} \beta_{-i})}{X_i^T X_i}
\]

Coordinate descent repeats this update for \( i = 1, 2, \ldots, p, 1, 2, \ldots \). Note that this is exactly Gauss-Seidl for the system \( X^T X \beta = X^T y \).
Coordinate descent vs gradient descent for linear regression: 100 random instances with $n = 100$, $p = 20$
Is it fair to compare 1 cycle of coordinate descent to 1 iteration of gradient descent? Yes, if we’re clever

- Gradient descent: \( \beta \leftarrow \beta + tX^T(y - X\beta) \), costs \( O(np) \) flops
- Coordinate descent, one coordinate update:

\[
\beta_i \leftarrow \frac{X_i^T(y - X_{-i}\beta_{-i})}{X_i^TX_i} = \frac{X_i^Tr}{\|X_i\|^2} + \beta_i
\]

where \( r = y - X\beta \)

- Each coordinate costs \( O(n) \) flops: \( O(n) \) to update \( r \), \( O(n) \) to compute \( X_i^Tr \)
- One cycle of coordinate descent costs \( O(np) \) operations, same as gradient descent
Example: lasso regression

Consider the lasso problem:

$$\min_{\beta} \frac{1}{2} \| y - X \beta \|_2^2 + \lambda \| \beta \|_1$$

Note that nonsmooth part here is separable: $$\| \beta \|_1 = \sum_{i=1}^{p} |\beta_i|$$. Minimizing over $$\beta_i$$, with $$\beta_j, j \neq i$$ fixed:

$$0 = X_i^T X_i \beta_i + X_i^T (X_{-i} \beta_{-i} - y) + \lambda s_i$$

where $$s_i \in \partial |\beta_i|$$. Solution is simply given by soft-thresholding

$$\beta_i = S_{\lambda/\|x_i\|_2^2} \left( \frac{X_i^T (y - X_{-i} \beta_{-i})}{X_i^T X_i} \right)$$

Repeat this for $$i = 1, 2, \ldots, p, 1, 2, \ldots$$
Coordinate descent vs proximal gradient for lasso regression: 100 random instances with $n = 200$, $p = 50$ (all methods cost $O(np)$ per iter)
Example: box-constrained QP

Given \( b \in \mathbb{R}^n \), \( Q \in \mathbb{S}_+^n \), consider a box-constrained QP:

\[
\min_x \frac{1}{2} x^T Q x + b^T x \quad \text{subject to } l \leq x \leq u
\]

Fits into our framework, as \( I\{l \leq x \leq u\} = \sum_{i=1}^n I\{l_i \leq x_i \leq u_i\} \)

Minimizing over \( x_i \) with all \( x_j, j \neq i \) fixed: same basic steps give

\[
x_i = T_{[l_i, u_i]} \left( \frac{b_i - \sum_{j \neq i} Q_{ij} x_j}{Q_{ii}} \right)
\]

where \( T_{[l_i, u_i]} \) is the truncation (projection) operator onto \([l_i, u_i] \):

\[
T_{[l_i, u_i]}(z) = \begin{cases} 
  u_i & \text{if } z > u_i \\
  z & \text{if } l_i \leq z \leq u_i \\
  l_i & \text{if } z < l_i 
\end{cases}
\]
Example: support vector machines

A coordinate descent strategy can be applied to the SVM dual:

$$\min_{\alpha} \frac{1}{2} \alpha^T \tilde{X} \tilde{X}^T \alpha - 1^T \alpha \quad \text{subject to} \quad 0 \leq \alpha \leq C, \quad \alpha^T y = 0$$

Sequential minimal optimization or SMO (Platt 1998) is basically blockwise coordinate descent in blocks of 2. Instead of cycling, it chooses the next block greedily.

Recall the complementary slackness conditions

$$\alpha_i (1 - \xi_i - (\tilde{X} \beta)_i - y_i \beta_0) = 0, \quad i = 1, \ldots, n \quad (1)$$

$$\ (C - \alpha_i) \xi_i = 0, \quad i = 1, \ldots, n \quad (2)$$

where $\beta, \beta_0, \xi$ are the primal coefficients, intercept, and slacks. Recall that $\beta = \tilde{X}^T \alpha$, $\beta_0$ is computed from $(1)$ using any $i$ such that $0 < \alpha_i < C$, and $\xi$ is computed from $(1)$, $(2)$.
SMO repeats the following two steps:

- Choose $\alpha_i, \alpha_j$ that violate complementary slackness, greedily (using heuristics)
- Minimize over $\alpha_i, \alpha_j$ exactly, keeping all other variables fixed

Using equality constraint, reduces to minimizing univariate quadratic over an interval (From Platt 1998)

Note this does not meet separability assumptions for convergence from Tseng (2001), and a different treatment is required

Many further developments on coordinate descent for SVMs have been made; e.g., a recent one is Hsiesh et al. (2008)
Coordinate descent in statistics and ML

History in statistics/ML:

- Idea appeared in Fu (1998), and then again in Daubechies et al. (2004), but was inexplicably ignored
- Later, three papers in 2007, especially Friedman et al. (2007), really sparked interest in statistics and ML communities

Why is it used?

- Very simple and easy to implement
- Careful implementations can achieve state-of-the-art
- Scalable, e.g., don’t need to keep full data in memory

Examples: lasso regression, lasso GLMs (under proximal Newton), SVMs, group lasso, graphical lasso (applied to the dual), additive modeling, matrix completion, regression with nonconvex penalties
Pathwise coordinate descent for lasso

Structure for pathwise coordinate descent, Friedman et al. (2009):

Outer loop (pathwise strategy):
- Compute the solution over a sequence $\lambda_1 > \lambda_2 > \cdots > \lambda_r$ of tuning parameter values
- For tuning parameter value $\lambda_k$, initialize coordinate descent algorithm at the computed solution for $\lambda_{k+1}$ (warm start)

Inner loop (active set strategy):
- Perform one coordinate cycle (or small number of cycles), and record active set $A$ of coefficients that are nonzero
- Cycle over only the coefficients in $A$ until convergence
- Check KKT conditions over all coefficients; if not all satisfied, add offending coefficients to $A$, go back one step
Notes:

- Even when the solution is desired at only one $\lambda$, the pathwise strategy (solving over $\lambda_1 > \cdots > \lambda_r = \lambda$) is typically much more efficient than directly performing coordinate descent at $\lambda$.
- Active set strategy takes advantage of sparsity; e.g., for large problems, coordinate descent for lasso is much faster than it is for ridge regression.
- With these strategies in place (and a few more clever tricks), coordinate descent can be competitive with fastest algorithms for $\ell_1$ penalized minimization problems.
- Fortran implementation *glmnet*, linked to R, MATLAB, etc.
Coordinate gradient descent

For a smooth function $f$, the iterations

$$x_i^{(k)} = x_i^{(k-1)} - t_{ki} \cdot \nabla_i f(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)}),$$

$i = 1, \ldots, n$

for $k = 1, 2, 3, \ldots$ are called coordinate gradient descent, and when $f = g + h$, with $g$ smooth and $h = \sum_{i=1}^n h_i$, the iterations

$$x_i^{(k)} = \text{prox}_{h_i, t_{ki}} \left( x_i^{(k-1)} - t_{ki} \cdot \nabla_i g(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)}) \right),$$

$i = 1, \ldots, n$

for $k = 1, 2, 3, \ldots$ are called coordinate proximal gradient descent

When $g$ is quadratic, (proximal) coordinate gradient descent is the same as coordinate descent under proper step sizes
Convergence analyses

Theory for coordinate descent moves quickly. Each combination of the following cases has (probably) been analyzed:

- Coordinate descent or (proximal) coordinate gradient descent?
- Cyclic rule, permuted cyclic, or greedy rule, randomized rule?

Roughly speaking, results are similar to those for proximal gradient descent: under standard conditions, get standard rates

But constants differ and this matters! Much recent work is focused on improving them

Also, it is generally believe that coordinate descent should perform better than first-order methods (when implementable)

Some references are Beck and Tetruashvili (2013), Wright (2015), Sun and Hong (2015), Li et al. (2016)
Graphical lasso

Consider $X \in \mathbb{R}^{n \times p}$, with rows $x_i \sim N(0, \Sigma), i = 1, \ldots, n$, drawn independently. Suppose $\Sigma$ is unknown. It is often reasonable (for large $p$) to seek a sparse estimate of $\Sigma^{-1}$

Why? For $z \sim N(0, \Sigma)$, normality theory tells us

$$\Sigma_{ij}^{-1} = 0 \iff z_i, z_j \text{ conditionally independent given } z_\ell, \ell \neq i, j$$

**Graphical lasso** (Banerjee et al. 2007, Friedman et al. 2007):

$$\min_{\Theta \in S_p^+} - \log \det \Theta + \text{tr}(S \Theta) + \lambda \|\Theta\|_1$$

where $S = X^T X / n$, and $\|\Theta\|_1 = \sum_{i,j=1}^p |\Theta_{i,j}|$. Observe that this is a convex problem. Solution $\hat{\Theta}$ serves as estimate for $\Sigma^{-1}$
Glasso algorithm

Graphical lasso KKT conditions (stationarity):

\[
-\Theta^{-1} + S + \lambda \Gamma = 0
\]

where \( \Gamma_{ij} \in \partial|\Theta_{ij}| \). Let \( W = \Theta^{-1} \). Note \( W_{ii} = S_{ii} + \lambda \), because \( \Theta_{ii} > 0 \) at solution. Now partition:

\[
W = \begin{bmatrix} W_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}, \quad \Theta = \begin{bmatrix} \Theta_{11} & \theta_{12} \\ \theta_{21} & \theta_{22} \end{bmatrix}, \quad S = \begin{bmatrix} S_{11} & s_{12} \\ s_{21} & s_{22} \end{bmatrix}, \quad \Gamma = \begin{bmatrix} \Gamma_{11} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} \end{bmatrix}
\]

where \( W_{11} \in \mathbb{R}^{(p-1) \times (p-1)} \), \( w_{12} \in \mathbb{R}^{(p-1) \times 1} \), and \( w_{21} \in \mathbb{R}^{1 \times (p-1)} \), \( w_{22} \in \mathbb{R} \); same with others

Glasso algorithm (Friedman et al., 2007): solve for \( w_{12} \) (recall that \( w_{22} \) is known), with all other columns fixed; then solve for second-to-last column, etc., and cycle around until convergence
Screening rules

In some problems, screening rules can be used in combination with coordinate descent to further wittle down the active set. Screening rules themselves have amassed a huge literature recently.

Originated with El Ghaoui et al. (2010), SAFE rule for the lasso:

\[ |X^T_i y| < \lambda - \|X_i\|_2 \|y\|_2 \frac{\lambda_{\text{max}} - \lambda}{\lambda_{\text{max}}} \Rightarrow \hat{\beta}_i = 0, \quad \text{all } i = 1, \ldots, p \]

where \( \lambda_{\text{max}} = \|X^T y\|_{\infty} \) (the smallest value of \( \lambda \) such that \( \hat{\beta} = 0 \))

Note: this is not an if and only if statement! But it does give us a way of eliminating features apriori, without solving the lasso.

(There have been many advances in screening rules for the lasso ... but this was the first, and one of the simplest)
SAFE rule derivation

Why is the SAFE rule true? Construction comes from lasso dual:

\[
\max_u g(u) \text{ subject to } \|X^T u\|_\infty \leq \lambda
\]

where \( g(u) = \frac{1}{2} \|y\|_2^2 - \frac{1}{2} \|y - u\|_2^2 \). Suppose that \( u_0 \) is dual feasible (e.g., take \( u_0 = y \cdot \lambda/\lambda_{\max} \)). Then \( \gamma = g(u_0) \) is a lower bound on the dual optimal value, so dual problem is equivalent to

\[
\max_u g(u) \text{ subject to } \|X^T u\|_\infty \leq \lambda, \quad g(u) \geq \gamma
\]

Now let \( m_i = \max_u |X_i^T u| \) subject to \( g(u) \geq \gamma \), for \( i = 1, \ldots, p \). Then

\[
m_i < \lambda \implies |X_i^T \hat{u}| < \lambda \implies \hat{\beta}_i = 0, \quad i = 1, \ldots, p
\]

The last implication comes from the KKT conditions.
(From El Ghaoui et al. 2010)
Another dual argument shows that

$$\max_u X_i^T u \text{ subject to } g(u) \geq \gamma$$

$$= \min_{\mu > 0} \gamma \mu + \frac{1}{\mu} \|\mu y - X_i\|_2^2$$

$$= \|X_i\|_2 \sqrt{\|y\|_2 - 2\gamma - X_i^T y}$$

where the last equality comes from direct calculation.

Thus $m_i$ is given the maximum of the above quantity over $\pm X_i$,

$$m_i = \|X_i\|_2 \sqrt{\|y\|_2 - 2\gamma + |X_i^T y|}, \quad i = 1, \ldots, p$$

Lastly, substitute $\gamma = g(y \cdot \lambda / \lambda_{\text{max}})$. Then $m_i < \lambda$ is precisely the SAFE rule given on previous slide.
Early coordinate descent references in optimization:

Early coordinate descent references in statistics and ML:

- W. Fu (1998), “Penalized regressions: the bridge versus the lasso”
Applications of coordinate descent:

Recent theory for coordinate descent:

- Y. Nesterov (2010), “Efficiency of coordinate descent methods on huge-scale optimization problems”
Screening rules and graphical lasso:

- R. Mazumder and T. Hastie (2011), “Exact covariance thresholding into connected components for large-scale graphical lasso”