10-725/36-725: Convex Optimization **Fall 2019**

Lecture 19: October 30th Coordinate Descent

Lecturer: Lecturer: Ryan Tibshirani *Scribes: Samuel Levy, Melda Korkut, Mingjie Sun*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

19.1 Recap from last time: numerical linear algebra

In $\mathbb{R}^n$, the 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

The operations with banded or sparse matrices are much cheaper.

Moreover, we have seen that there exists two classes of approaches for linear system solvers:

- **Direct**: QR decomposition, Cholesky decomposition.
  - These methods require a number of iterations that is independent from the desired level of accuracy. In other words, the accuracy of those methods do not depend on the conditioning.
  - Those methods are usually fast under sparsity but we need to worry special cases.
  - Update/downdate efficiently (e.g. we can recompute the QR decomposition of A after adding or deleting one row or column in $\mathcal{O}$(n) time).
- **Indirect**: Jacobi, Gauss-Seidl, gradient descent, conjugate gradients.
  - The accuracy of these methods vary. Furthermore, those methods are always faster under sparsity of the A matrix.

19.2 Coordinatewise optimality

We focus here on a very simple technique that can be surprisingly efficient and scalable: **coordinate descent** or more formally, coordinatewise minimization.

Coordinate descent answers the following question: given convex, differentiable $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, if we are at a point x such that $f(x)$ is minimized along each coordinate axis, have we found a *global minimizer*? That is, do we have:

$$f(x + \delta e_i) \geq f(x) \text{ for all } \delta, i \Rightarrow f(x) = \min_z f(z)$$

The answer is yes, because for convex functions, $f(x+\delta e_i) \geq f(x)$ for all δ, i is equivalent to $\left(\frac{\partial f}{\partial x_1}(x), ..., \frac{\partial f}{\partial x_n}(x)\right)$. Every partial derivative is zero, and each of these are also convex. An example of convex and differentiable function is given in Figure 19.1.

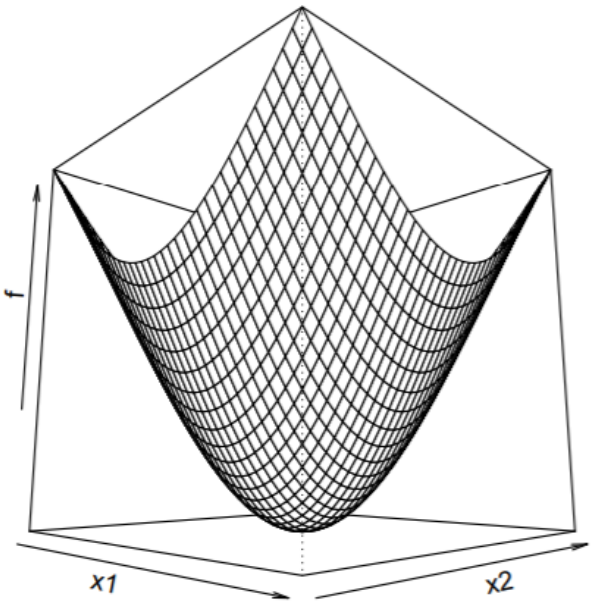

Figure 19.1: Example of convex and differentiable function

Now, for f convex and not differentiable, coordinate descent does not lead to a global minimizer in general. Figure 19.2 (right panel) illustrates why: since we try to change only one coordinate while fixing the other one, we see that moving horizontally and right from the intersection between the two red dotted lines, we observe an increase in the criterion; likewise, when we move vertically and up from the intersection between the two red dotted lines, we also observe that the criterion increases. However, moving simultaneously to the right hand upper corner, the criterion decreases (we get closer to the blue dot): coordinate descent fails.

Now, for $f(x) = g(x) + \sum_{i=1}^n h_i(x_i)$ with g convex, smooth, and each h_i convex (the nonsmooth part is called *separable*), coordinate descent leads to global minimization.

Proof: Using convexity of g and subgradient optimality, we have:

$$
\begin{aligned}
0 &\in \nabla_i g(x) + \partial h_i(x_i) && (19.1)\\
&\iff -\nabla_i g(x) \in \partial h_i(x_i) && (19.2)\\
&\iff h_i(y_i) \geq h_i(x_i) - \nabla_i g(x)(y_i - x_i) && (19.3)\\
&\iff \nabla_i g(x)(y_i - x_i) + h_i(y_i) - h_i(x_i) \geq 0 && (19.4)
\end{aligned}
$$

and by convexity of f, using the first-order characterization:

$$
f(y) - f(x) = g(x) - g(y) + \sum_{i=1}^n [h_i(y_i) - h_i(x_i)] \geq \sum_{i=1}^n [\nabla_i g(x)(y_i - x_i) + h_i(y_i) - h_i(x_i)] \geq 0 \qquad (19.5)
$$

■

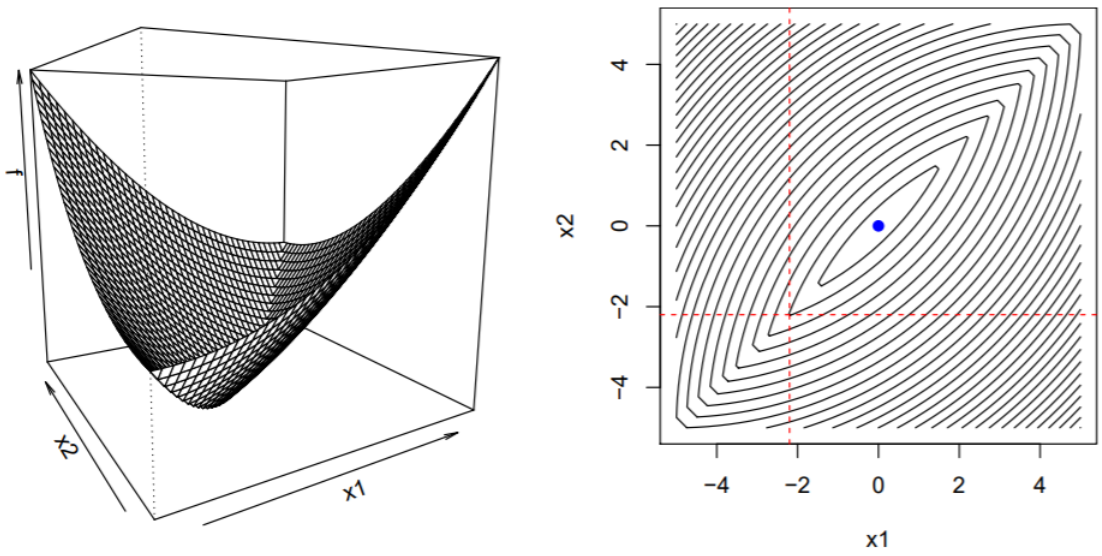

Figure 19.2: Example of convex and nondifferentiable function: 3D plot (left panel) and contour plot (right panel)

19.3 Coordinate descent

This suggests that for the problem

$$\min_x f(x) \tag{19.6}$$

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

$$x_i^{(k)} = \text{argmin}_{x_i} f(x_1^{(k)}, ..., x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, ..., x_n^{(k-1)}), \text{ for } i = 1, ..., n \text{ and } k = 1, 2, 3...$$

In other words, we minimize with respect to one element x_i, plug it back in f, and move to the next index. Important note: we always use the most recent information possible.

Tseng (2001) showed that 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 $x^{(k)}$, $k = 1, 2, 3...$ is a minimizer of f[1].

A few notes for coordinate descent:

- The order of cycle through coordinates is arbitrary, we can use any permutation of {1,...,n};
- we can replace everywhere individual coordinates with blocks of coordinates - even blocks of coordinates where there are repeated coordinates (e.g. 1,2,1,1,1,3,1,2) are acceptable, as long as we see them after a linear time;
- The "one-at-a-time" update scheme is critical, and "all-at-once" scheme **does not** necessarily converge;

[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)

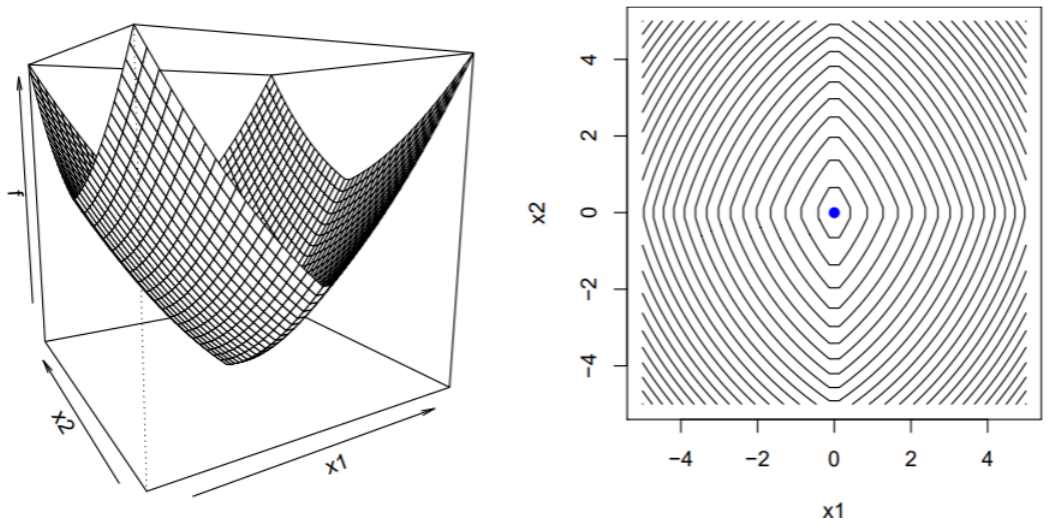

Figure 19.3: Example of a function $f(x) = g(x) + \sum_{i=1}^{n} h_i(x_i)$ with g convex, smooth, and each h_i convex.

- The analogy for solving linear systems: Gauss-Seidel versus Jacobi method.

19.4 Example: linear regression

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

$$\min_{\beta} \frac{1}{2} \| y - X\beta \|_2^2 \tag{19.7}$$

Minimizing over β_i, with all β_j, $j \neq i$ fixed:

$$0 = \nabla_i f(\beta) = X_i^T (X\beta - y) = X_i^T (X_i \beta_i + X_{-i}\beta_{-i} - y) \tag{19.8}$$

where X_{-i} and β_{-i} are original matrix or vector with i-th column or element removed respectively.

Equation 19.8 is equivalent to:

$$\beta_i = \frac{X_i^T (y - X_{-i}\beta_{-i})}{X_i^T X_i} \tag{19.9}$$

Coordinate descent repeats this update for $i = 1, 2, ..., p$,

Note that the computational cost for one cycle of coordinate descent is $\mathcal{O}(\text{np})$ where $\mathcal{O}(\text{n})$ to compute $X_i^T (y - X_{-i}\beta_{-i})$ for each update in a cycle (it is $\mathcal{O}(n)$ because we can precompute $X_i^T X_i \beta_i$), which is the same as gradient descent. Each coordinate costs $\mathcal{O}(n)$ to update r, $\mathcal{O}(n)$ to compute $X_i^T r$.

Equation 19.9 is equivalent to:

$$\beta_i = \frac{X_i^T r}{X_i^T X_i} = \frac{X_i^T r}{\| X_i \|_2^2} + \beta_i \tag{19.10}$$

where $r = y - X\beta$

We observe in Figure 19.5 how different methods converge, using 100 random instances with $n = 100$ and $p = 20$. In particular, coordinate descent's speed of convergence is comparable to the conjugate gradient for linear regression.

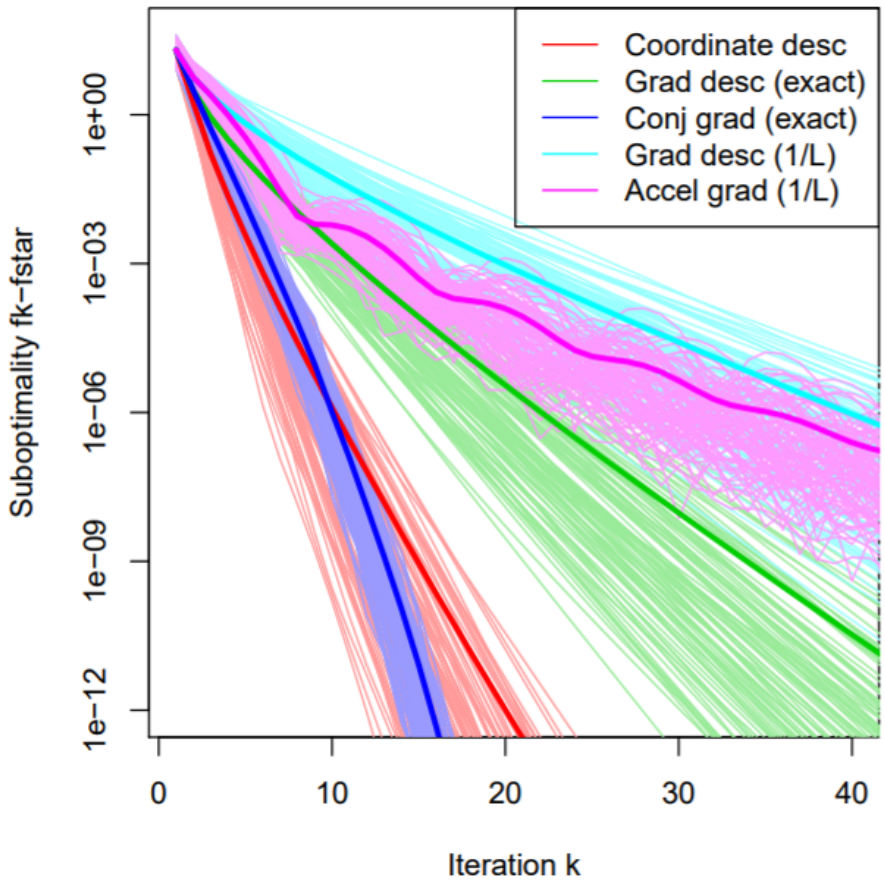

Figure 19.4: Comparison of convergence for several first-order methods

19.5 Example: Lasso Regression

Given $y \in \mathbb{R}^n$, and $X \in \mathbb{R}^{n \times p}$ whose columns are $X_1, \ldots, X_n$, consider lasso problem with:

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

Notice that here $\|\beta\|_1$ is convex, not differentiable but separable, since $\|\beta\|_1 = \sum_{i=1}^{p} |\beta_i|$. Minimizing over β_i we get

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

where $s_i \in \partial|\beta_i|$. By using soft-thresholding we get,

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

Figure below shows proximal gradient vs coordinate descent for lasso regression. The coordinate gradient descent here and gradient descent both share $O(np)$ flops in each iteration.

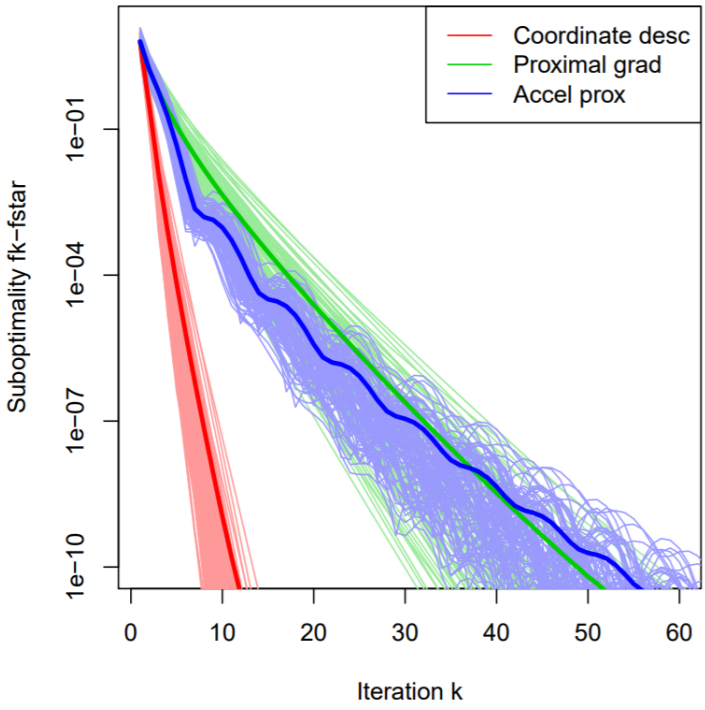

Figure 19.5: Coordinate descent vs proximal gradient for lasso

19.6 Example: Box-constrained QP

Let $b \in \mathbb{R}^n$, $Q \in \mathbb{S}_+^n$. A box-constrained QP is:

$$\min_x \frac{1}{2}x^T Q x + b^T x \text{ subject to } l \le x \le u$$

Notice that the box constraints are separable coordinate-wise.

$$I\{l \le x \le u\} = \sum_{i=1}^n I\{l_i \le x_i \le u_i\}$$

Minimizing over x_i gives

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

where $T_{[l_i,u_i]}$ is:

$$T_{[l_i,u_i]}(z) = \begin{cases} u_i & \text{if } z > u_i; \\ z & \text{if } l_i \le z \le u_i \\ l_i & \text{if } z < l_i \end{cases}$$

19.7 Example: Support Vector Machines

Coordinate descent can be applied to SVM dual problem:

$$\min_{\alpha} \frac{1}{2}\alpha^T \tilde{X}\tilde{X}^T\alpha - 1^T\alpha \text{ subject to } 0 \leq \alpha \leq C1, \alpha^T y = 0.$$

Notice that the equality constraint is not separable, so we cannot do the trick we did in the previous example. So, Platt proposed Sequential minimal optimization algorithm. SMO does basically coordinate descent in blocks of 2. Instead of cycling, it chooses the next block greedily. Using complementary slackness conditions

$$\alpha_i \left(1 - \psi_i - (\tilde{X}\beta)_i - y_i\beta_0\right) = 0$$

$$(C - \alpha_i)\,\psi_i = 0$$

where β, β_0, ψ are the primal coefficients, intercept and slacks respectively. $\beta = \tilde{X}^T\alpha$ and β_0 is calculated by using any i such that $0 < \alpha_i < C$, and ψ is calculated from the two conditions above.

SMO repeats the following:

- Choose α_i, α_j greedily such that they violate complementary slackness
- Minimize over α_i, α_j exactly, keeping all other variables fixed

The second step reduces to minimizing univariate quadratic over an interval, using the equality constraint.

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

19.8 Coordinate Descent in Statistics and ML

In history, the idea appeared in Fu (1998), then in Daubechies et al. (2004), but it was ignored then. Later in 2007 it gained popularity.

The advantages of the method is that it is very easy to implement and pretty simple. If carefully implemented, it can be state-of-the-art. It is also scalable since it does not need to keep full data in memory.

Examples of applications: lasso regression, lasso GLMs (under proximal Newton), SVMs, group lasso, graphical lasso (applied to the dual), additive modeling, matrix completion, regression with nonconvex penalties.

19.9 Pathwise Coordinate Descent for Lasso

Friedman et al. proposed the pathwise coordinate descent method for Lasso problem. The algorithm runs over two loops.

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 λ_k, initialize coordinate descent algorithm at the computed solution for λ_{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 coefficents to A, go back one step

Some important points:

- Even when the solution is desired at only one , the pathwise strategy (solving over $\lambda_1 > \lambda_2 > ...\lambda_r = \lambda$) is typically much more efficient than directly performing coordinate descent at λ, indeed this is what many existing packages will do.
- Active set strategy takes advantage of sparsity; e.g., for very 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 l_1 penalized minimization problems
- Fast Fortran implement glmnet, which can be linked to R or MATLAB

19.10 Coordinate Gradient Descent

For a smooth function f, the iterations

$$x_i^{(k)} = x_I^{(k-1)} - t_{ki} \nabla_i f \left(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)} \right), \quad i = 1, \ldots, n$$

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

$$x_i^{(k)} = prox_{h_i, t_{ki}} \left(x_i^{(k-1)} - t_{ki} \nabla_i g \left(x_1^{(k)}, \ldots, x_i^{(k-1)}, \ldots, x_n^{(k-1)} \right) \right), \quad i = 1, \ldots, n$$

for $k = 1, 2, 3 \ldots$ are called *coordinate proximal gradient descent*. When g is quadratic, these two updates are the SAME under proper step sizes.

19.11 Convergence analysis

There has been a lot of theory for. coordinate descent. Each combination of the following cases: coordinate descent/coordinate gradient descent, cyclic rule/permuted cyclic/greedy rule/randomized rule has been analyzed before. It is worth noting that the constants in the convergence rate matters and there are much recent work on improving those constant term. Last, it is generally believe that coordinate descent should perform better than first-order methods.

19.12 Screening rules

Screening rules works by pre-computing those variables that are bound to be zero in the optimal solution. Then discarding these variables makes it easier to solve the problem.

There has been a lot of research on designing screening rules for the lasso problem. Here we introduce the first and one of the simplest rule:

Theorem 19.1 ***SAFE*** *rule for the lasso problem: For any $i \in \{1, \cdots, p\}$, if the following is satisfied:*

$$|X_i^T y| < \lambda - \|X_i\|_2 \|y\|_2 \frac{\lambda_{max} - \lambda}{\lambda_{max}}$$

where $\lambda_{max} = \|X^T y\|_\infty$, then in the optimal solution β, we have $\beta_i = 0$.

Proof: Recall the dual of the Lasso problem:

$$\begin{aligned} \min_u \quad & g(u) \\ \text{subject to} \quad & \|X^T u\|_\infty \leq \lambda \end{aligned}$$

where $g(u) = \frac{1}{2}\|y\|_2^2 - \frac{1}{2}\|y - u\|_2^2$. We first find a feasible point of the dual problem, recall that:

$$\lambda_{max} = \|X^T y\|_\infty$$

Then if we let $u_0 = y \cdot \frac{\lambda}{\lambda_{max}}$, we have:

$$\|X^T u_0\|_\infty \leq \lambda$$

Now we have a feasible point u_0 of the lasso dual formulation. Then $\gamma = g(u_0)$ is a lower bound on the dual optimal value. So the dual problem is equivalent to:

$$\begin{aligned} \min_u \quad & g(u) \\ \text{subject to} \quad & \|X^T u\|_\infty \leq \lambda \\ & g(u) \geq \gamma \end{aligned}$$

The KKT condition (stationary condition) for lasso tells us that for the optimal point u_{opt} of dual problem, if $X_i^T u_{opt} < \lambda$, then β_i must be zero at the optimal solution of lasso.
Then for each i, consider the following problem:

$$\begin{aligned} \max_u \quad & |X_i^T u| \\ \text{subject to} \quad & g(u) \geq \gamma \end{aligned}$$

We denote the optimal criterion value of the above problem as m_i. Notice that this problem is not convex. But we can solve the following convex problem and compute m_i as the maximum of the criterion values over $\pm X_i$.

$$\begin{aligned} \max_u \quad & X_i^T u \\ \text{subject to} \quad & g(u) \geq \gamma \end{aligned}$$

The dual problem is:

$$\begin{aligned} \min_\mu \quad & -\gamma\mu + \frac{1}{2\mu}\|\mu y - X_i\|_2^2 \\ \text{subject to} \quad & \mu > 0 \end{aligned}$$

We now directly solve the dual:

$$\begin{aligned} -\gamma\mu + \frac{1}{\mu}\|\mu y - X_i\|_2^2 &= (\|y\|_2^2 - 2\gamma)\frac{\mu}{2} + \frac{1}{2\mu}\|X_i\|_2^2 - X_i^T y \\ &\geq \|X_i\|_2\sqrt{\|y\|_2^2 - 2\gamma} - X_i^T y \end{aligned}$$

where the equality is reached when $\mu = \sqrt{\frac{\|X_i\|_2^2}{\|y\|_2^2 - 2\gamma}}$.
Taking the maximum over $\pm X_i$: (recall $\gamma = g(u_0)$)

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

We want m_i to be smaller than λ, which guarantees that $\beta_i = 0$:

$$\begin{aligned} & m_i < \lambda \\ \iff & \|X_i\|_2\sqrt{\|y\|_2^2 - 2\gamma} + |X_i^T y| < \lambda \\ \iff & \|X_i\|_2 \cdot \|y - y \cdot \frac{\lambda}{\lambda_{max}}\|_2 + |X_i^T y| < \lambda \\ \iff & |X_i^T y| < \lambda - \|X_i\|_2\|y\|_2\frac{\lambda_{max} - \lambda}{\lambda_{max}} \end{aligned}$$

■

References

[T01] P. TSENG, "Convergence of a Block Coordinate Descent Method for Nondifferentiable Minimization", *Journal of Optimization Theory and Applications*, 2001, pp. 475–494.

[F98] W. J. FU "Penalized Regressions: The Bridge versus the Lasso", *Journal of Computational and Graphical Statistics*, 1998, 7:3, 397-416.