Case Study 3: fMRI Prediction

Stochastic Coordinate Descent (SCD) for LASSO (Shooting)
Parallel SCD (Shotgun)
Parallel SGD
Averaging Solutions

Machine Learning/Statistics for Big Data
CSE599C1/STAT592, University of Washington
Carlos Guestrin

February 21st, 2013

Today

- One way to solve LASSO problem
- Stochastic Coordinate Descent (SCD)
- Minimizing a coordinate in LASSO
- A simple SCD for LASSO (Shooting)
  - Your HW, a more efficient implementation! ☺
- Analysis of SCD
- Parallel SCD (Shotgun)
- Other parallel learning approaches for linear models
  - Parallel stochastic gradient descent (SGD)
  - Parallel independent solutions then averaging

Coordinate Descent

$F(\beta_1, \ldots, \beta_d)$

- Given a function $F(\beta)$
  - Want to find minimum $\beta^* \leftarrow \min_\beta F(\beta)$
- Often, hard to find minimum for all coordinates, but easy for one coordinate

  1-d optimization problem
- Coordinate descent:

  while not converged

  pick coordinate j

  $$\beta_j \leftarrow \min_b F(\beta_1, \beta_2, \ldots, \beta_{j-1}, b, \beta_{j+1}, \ldots, \beta_d)$$
- How do we pick a coordinate?

  Round robin, random, smartly, ...
- When does this converge to optimum? e.g., strongly convex (separability)

LASSO Regression

- **LASSO:** least absolute shrinkage and selection operator
- New objective:

$$\min_\beta \underbrace{\sum_{i=1}^{N} (y^i - (\beta_0 + \beta^T x^i))^2}_{RSS(\beta)} + \lambda \|\beta\|_1$$

$$\Updownarrow$$

$$\min_\beta RSS(\beta) \quad \text{s.t.} \quad \|\beta\|_1 \le B$$

Soft Threshholding

$F(\beta) = RSS(\beta) + \lambda\|\beta\|_1$

- Gradient of RSS term:

$$\frac{\partial}{\partial \beta_j} RSS(\beta) = a_j \beta_j - c_j$$

$a_j = 2\sum_{i=1}^{N}(x_j^i)^2$

$c_j = 2\sum_{i=1}^{N} x_j^i (y^i - \beta_{-j}^T x_{-j}^i)$

β_{-j}, x_{-j}: all but the j^{th} coeff.

$c_j \propto corr(x_j, r_{-j})$

msr how relevant x_j is for pred y beyond what the others can

r_{-j}: residual from model w/o j^{th} cov.

- Subgradient of full objective:

$$\partial_{\beta_j} F(\beta) = (a_j\beta_j - c_j) + \lambda\, \partial_{\beta_j}\|\beta\|_1$$

$$= \begin{cases} a_j\beta_j - c_j - \lambda & \beta_j < 0 \\ [-c_j - \lambda, -c_j + \lambda] & \beta_j = 0 \\ a_j\beta_j - c_j + \lambda & \beta_j > 0 \end{cases}$$

Soft Threshholding

$\partial_{\beta_j} F = 0 \Rightarrow \min_{\beta_j} F(\beta_1, \ldots, \beta_{j-1}, \beta_j, \beta_{j+1}, \ldots, \beta_p)$

- Set subgradient = 0:

$$\partial_{\beta_j} F(\beta) = \left\{ \begin{array}{ll} a_j\beta_j - c_j - \lambda & \beta_j < 0 \\ [-c_j - \lambda, -c_j + \lambda] & \beta_j = 0 \\ a_j\beta_j - c_j + \lambda & \beta_j > 0 \end{array} \right. = 0$$

If $\beta_j < 0$

$a_j\beta_j - c_j - \lambda = 0$

$\Rightarrow \beta_j = \frac{c_j + \lambda}{a_j} < 0 \Rightarrow c_j < -\lambda$ strong neg. corr., then $\beta_j < 0$

If $\beta_j > 0$

$a_j\beta_j - c_j + \lambda = 0 \Rightarrow \beta_j = \frac{c_j - \lambda}{a_j} > 0 \Rightarrow c_j > \lambda$ strong pos. corr., then $\beta_j > 0$

If $\beta_j = 0$ $-\lambda < c_j < \lambda$ otherwise, $\beta_j = 0$

- The value of $c_j = 2\sum_{i=1}^{N} x_j^i (y^i - \beta'_{-j} x_{-j}^i)$ constrains β_j

Soft Threshholding

$a_j \geq 0$

$\min_b f(\ldots \beta_{j-1}, b, \beta_{j+1}, \ldots)$

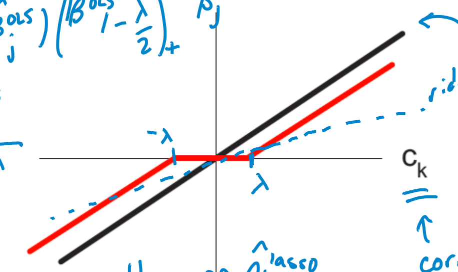
$$\hat{\beta}_j = \begin{cases} (c_j+\lambda)/a_j & c_j < -\lambda \\ 0 & c_j \in [-\lambda, \lambda] \\ (c_j-\lambda)/a_j & c_j > \lambda \end{cases} = \text{sign}\left(\frac{c_j}{a_j}\right)\left(\frac{|c_j|}{a_j} - \frac{\lambda}{a_j}\right)_+$$

$$\hat{\beta}_j = \text{sign}(c_j)\,\frac{(|c_j|-\lambda)_+}{a_j}$$

If $X^T X = I$

$$\hat{\beta}_j^{lasso} = \text{sign}(\hat{\beta}_j^{OLS})\left(|\hat{\beta}_j^{OLS}| - \frac{\lambda}{2}\right)_+$$

$$\hat{\beta}_j^{ridge} = \frac{\hat{\beta}_j^{OLS}}{1+\lambda}$$



$= \text{corr}(x_j, r_{-j})$

From Kevin Murphy textbook

In LASSO, all coeff $\hat{\beta}_j^{lasso}$ are shrunk relative to $\hat{\beta}_j^{OLS}$

Stochastic Coordinate Descent for LASSO (aka Shooting Algorithm)

- Repeat until convergence
  - Pick a coordinate *j* at random
    - Set:
$$\hat{\beta}_j = \begin{cases} (c_j+\lambda)/a_j & c_j < -\lambda \\ 0 & c_j \in [-\lambda, \lambda] \\ (c_j-\lambda)/a_j & c_j > \lambda \end{cases} = \text{sign}(c_j)\,\frac{(|c_j|-\lambda)_+}{a_j}$$
    - Where:

cache

$$a_j = 2\sum_{i=1}^{N}(x_j^i)^2 \qquad c_j = 2\sum_{i=1}^{N} x_j^i (y^i - \beta'_{-j} x_{-j}^i)$$

Cost per iteration $O(N)$

can be done more smartly...

proof: your HW!! :)

Analysis of SCD [Shalev-Shwartz, Tewari '09/'11]

$e_j = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}_j$

- Analysis works for LASSO, L1 regularized logistic regression, and other objectives!
- For (coordinate-wise) strongly convex functions:

$$F(\beta + \Delta\beta) \le F(\beta) + \partial\beta_j (\nabla F(\beta))_j + \gamma (\partial\beta_j)^2$$

$$\Delta\beta = \partial\beta_j e_j$$

Lasso $\gamma = 1$

Logistic Regression $\gamma = \frac{1}{4}$

- Theorem:
  - Starting from $\beta^{(0)}$
  - After T iterations

dim, how hard, where start

$$E\Big[\underbrace{F(\beta^{(T)})] - F(\beta^*)}_{\text{dist from opt}} \le \frac{d\left(\gamma \|\beta^*\|_2^2 + 2F(\beta^{(0)})\right)}{T+1}$$

$T+1$ ← gets linearly better with iters

  - Where E[ ] is wrt random coordinate choices of SCD

- Natural question: How does SCD & SGD convergence rates differ?

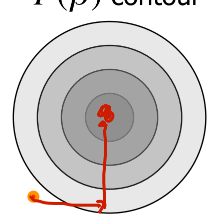
See paper: SCD → faster w. larger d ← no params to tune

SGD → faster w. larger N ← needs η

Shooting: Sequential SCD

Lasso: $\min_\beta F(\beta)$ where $F(\beta) = \|X\beta - \mathbf{y}\|_2^2 + \lambda \|\beta\|_1$

$F(\beta)$ contour



Stochastic Coordinate Descent (SCD)
(e.g., Shalev-Shwartz & Tewari, 2009)

While not converged,
- Choose random coordinate j,
- Update β_j (closed-form minimization)

how do we measure?
→ annoying: over a time window? has anything changed?
↳ do a round robin iter to measure convergence

Shotgun: Parallel SCD [Bradley et al '11]

Lasso: $\min_{\beta} F(\beta)$ where $F(\beta) = \|X\beta - \mathbf{y}\|_2^2 + \lambda \|\beta\|_1$

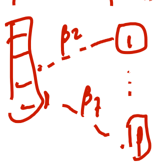
Shotgun (Parallel SCD)

While not converged,
- On each of P processors,
  - Choose random coordinate j,
  - Update β_j (same as for Shooting)

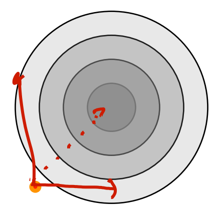
independently

procs.

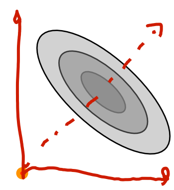
β



act as if they were only children



yes!! features are uncorrelated



no!! features are highly correlated

Is SCD inherently sequential?

Lasso: $\min_{\beta} F(\beta)$ where $F(\beta) = \|X\beta - \mathbf{y}\|_2^2 + \lambda \|\beta\|_1$

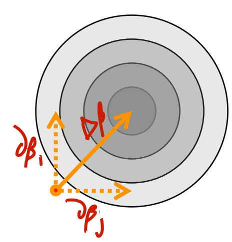
Coordinate update:

$$\beta_j \leftarrow \beta_j + \delta\beta_j$$

(closed-form minimization)

Collective update:

$$\Delta\beta = \begin{pmatrix} \delta\beta_i \\ 0 \\ 0 \\ \delta\beta_j \\ 0 \end{pmatrix}$$



Is SCD inherently sequential?

Lasso: $\min_{\beta} F(\beta)$ where $F(\beta) = \| X\beta - \mathbf{y} \|_2^2 + \lambda \| \beta \|_1$

Theorem: If X is normalized s.t. diag($X^T X$)=1,

$F(\beta + \Delta\beta) - F(\beta)$] decrease in objective

$$\leq \underbrace{-\sum_{i_j \in \mathcal{P}} \left(\delta\beta_{i_j}\right)^2}_{\text{"positive" progress}} + \underbrace{\sum_{\substack{i_j, i_k \in \mathcal{P}, \\ j \neq k}} \left(X^T X\right)_{i_j, i_k} \delta\beta_{i_j} \delta\beta_{i_k}}_{\text{could be positive or negative}}$$

"interference" or "bias" from parallelism

Is SCD inherently sequential?

Theorem: If X is normalized s.t. diag($X^T X$)=1,

$F(\beta + \Delta\beta) - F(\beta)$

Key term ← measures magnitude of interference

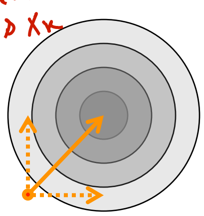
$$\leq -\sum_{i_j \in \mathcal{P}} \left(\delta\beta_{i_j}\right)^2 + \sum_{\substack{i_j, i_k \in \mathcal{P}, \\ j \neq k}} \left(X^T X\right)_{i_j, i_k} \delta\beta_{i_j} \delta\beta_{i_k}$$

$(X^T X)_{jk} = 0$

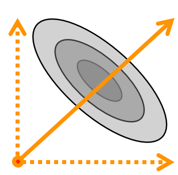
"correlation" x_j & x_k

$(X^T X)_{jk} \neq 0$

"interference"



Nice case:
Uncorrelated
features



Bad case:
Correlated
features

Shotgun: Convergence Analysis

<u>Lasso</u>: $\min_\beta F(\beta)$ where $F(\beta) = \| X\beta - \mathbf{y} \|_2^2 + \lambda \| \beta \|_1$

dim

Assume # parallel updates $P < d/\rho + 1$ spectral radius of $X^T X$

$$\underbrace{E[F(\beta^{(t)})]}_{\text{where we are}} - \underbrace{F(\beta^*)}_{\text{OPT}} \le \frac{d\left(\|\beta^*\|_2^2 + 2F(\beta^0)\right)}{TP}$$

iters, # procs

linear speed ups up to P processors

Generalizes bounds for Shooting (Shalev-Shwartz & Tewari, 2009)

Convergence Analysis

<u>Lasso</u>: $\min_\beta F(\beta)$ where $F(\beta) = \| X\beta - \mathbf{y} \|_2^2 + \lambda \| \beta \|_1$

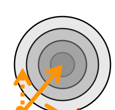
<u>Theorem: Shotgun Convergence</u>

Assume $P < d/\rho + 1$

where ρ = spectral radius of $\mathbf{X}^T\mathbf{X}$

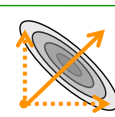
$$E\left[F(\beta^{(T)})\right] - F(\beta^*) \le \frac{d\left(\frac{1}{2}\|\beta^*\|_2^2 + F(\beta^{(0)})\right)}{TP}$$

Nice case: Uncorrelated features



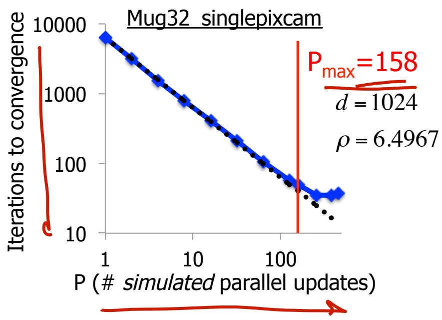
$\rho = \underline{1} \Rightarrow P_{\max} = \underline{d}$

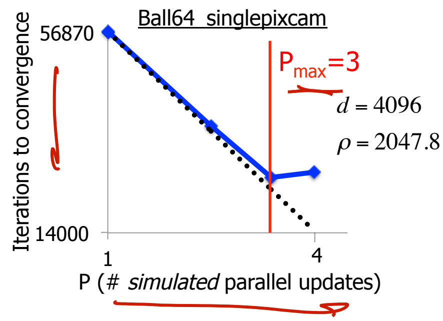
Bad case: Correlated features



$\rho = \underline{d} \Rightarrow P_{\max} = \underline{1}$ (at worst)

Empirical Evaluation





Stepping Back…

- Stochastic coordinate ascent SCD
  - Optimization: Pick a coordinate j; find $\min_{\beta_j}$
  - Parallel SCD: Pick P coordinates
  - Issue: coordinates may interfere P coordinates
  - Solution: bound possible interference based on ρ
- Natural counterpart: SGD
  - Optimization: Pick a data point i $\beta \leftarrow \beta - \eta \nabla F(x^i, \beta)$
  - Parallel Pick P data points & indep update β
  - Issue: can interfere in all coordinates
  - Solution: bound interfence

Parallel SGD with No Locks

[e.g., Hogwild!, Niu et al. '11]

- Each processor in parallel:
  - Pick data point i at random
  - For j = 1…d:

$$\beta_j \leftarrow \beta_j - \eta\,(\nabla F(x^i, \beta))_j$$

- Assume atomicity of: $\beta_j \leftarrow \beta_j + a$

other interferences

Addressing Interference in Parallel SGD

- Key issues:
  - Old gradients $\nabla F(x^{12}, \beta^{(7)}) \to \Delta_\beta^{(1)}$

Processor (1) $\beta^{(7)}$ $\beta^{(8)}$

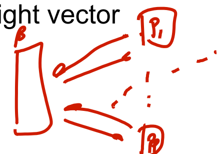
(2) $\Delta_\beta^{(12)}$

  - Processors overwrite each other's work
- Nonetheless:
  - Can achieve convergence and some parallel speedups
  - Proof uses weak interactions, but through sparsity of data points

sparsity X is key to analysis

Problem with Parallel SCD and SGD

- Both Parallel SCD & SGD assume access to current estimate of weight vector



fast access to β vector by all processors

- Works well on shared memory machines multicore

- Very difficult to implement efficiently in distributed memory, cloud

slow connection

- ~~Open problem~~: Good parallel SGD and SCD for distributed setting…
  - Let's look at a trivial approach

Simplest Distributed Optimization Algorithm Ever Made

- Given *N* data points & ~~*m*~~ p machines
- Stochastic optimization problem: $\min_\beta F(\beta) = \frac{1}{N}\sum_{i=1}^{N} F(x^i;\beta)$
- Distribute data: p machines

randomly assign data $\boxed{P_1}$ . . . . $\boxed{P_p}$

solves a problem D_k

$|D_k| = \frac{N}{p} := n$

- Solve problems independently

machine k independently estimates $\beta^{(k)} = \min_\beta \frac{1}{n} \sum_{x^i \in D_k} F(x^i;\beta)$

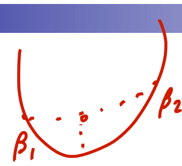
- Merge solutions

$$\tilde{\beta} = \frac{1}{p} \sum_k \beta^{(k)}$$

- Why should this work at all????

For Convex Functions…

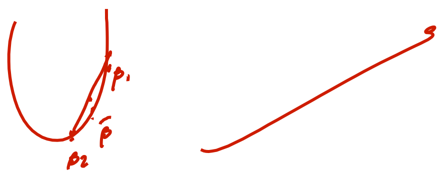
- Convexity:



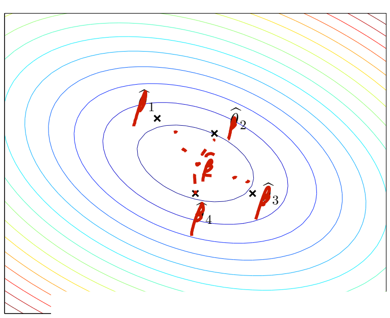
$$\frac{F(\beta_1)+F(\beta_2)}{2} \geq F(\bar{\beta})$$

- Thus:

$$\max(F(\beta_1), F(\beta_2)) \geq F(\bar{\beta})$$



Hopefully…



- Convexity only guarantees:

$$F(\bar{\beta}) \leq \max_{k} F(\beta^{(k)})$$

- But, estimates from independent data!

Figure from John Duchi

Analysis of Distribute-then-Average

[Zhang et al. '12]

- Under some conditions, including strong convexity, lots of smoothness, and more…
- If all data were in one machine, converge at rate: $\hat{\beta}_N \in \arg\min_{\beta} \frac{1}{N}\sum_{i=1}^{N} F(x^i, \beta)$

$$E[\|\hat{\beta}_N - \beta^*\|_2^2] = O\left(\frac{1}{N}\right)$$

- With p machines converge at a rate: $n = \frac{N}{p}$

$$E[\|\bar{\beta} - \beta^*\|_2^2] = O\Big(\underbrace{\frac{1}{N}}_{\text{unavoidable}} + \frac{1}{n^2}\Big) \quad \text{"bias" from parallelism}$$

e.g. 1T data points, 1000 machines $p \approx N^{\frac{1}{4}}$

$\Rightarrow$ plug in $\frac{1}{n^2} \rightarrow$ negligible when compared to $\frac{1}{N}$ great parallelism

Tradeoffs, tradeoffs, tradeoffs,…

- Distribute-then-Average:
  - "Minimum possible" communication
  - Bias term can be a killer with finite data
    - Issue definitely observed in practice
  - Significant issues for L1 problems:

    Sparsity patterns in machine i can be very different then those in machine j

    $\rightarrow$ average β $\Rightarrow$ lose sparsity
- Parallel SCD or SGD
  - Can have much better convergence in practice for multicore setting
  - Preserves sparsity (especially SCD)
  - But, hard to implement in distributed setting

What you need to know

- One way to solve LASSO problem
- Stochastic Coordinate Descent (SCD)
- Minimizing a coordinate in LASSO
- A simple SCD for LASSO (Shooting)
  - Your HW, a more efficient implementation! ☺
- Analysis of SCD
- Parallel SCD (Shotgun)
- Other parallel learning approaches for linear models
  - Parallel stochastic gradient descent (SGD)
  - Parallel independent solutions then averaging