Scaling Up LASSO Solvers

• Another 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
• ADMM
Coordinate Descent

- Given a function $F$
  - Want to find minimum

- Often, hard to find minimum for all coordinates, but easy for one coordinate

- Coordinate descent:
  
  $$\text{while not converged}$$
  
  $$\text{pick coord } j \text{ randomly}$$

  $$\beta_j \leq \arg\min_{b} F(\beta_1, ..., \beta_{j-1}, b, \beta_{j+1}, ..., \beta_{p})$$

- How do we pick a coordinate?

- When does this converge to optimum?

Soft Thresholding

$$\min_{\beta_j} \text{obj. } (\beta_1, ..., \beta_j, \beta_{j+1}, ..., \beta_p)$$

$$\hat{\beta}_j = \begin{cases} 
(c_j + \lambda)/a_j & c_j < -\lambda \\
0 & c_j \in [-\lambda, \lambda] \\
(c_j - \lambda)/a_j & c_j > \lambda 
\end{cases}$$

$$c_j = \text{cov}(X_j, Y_j)$$

$$a_j = \text{var}(X_j)$$

From Kevin Murphy textbook
Stochastic Coordinate Descent for LASSO (aka Shooting Algorithm)

• Repeat until convergence
  – Pick a coordinate $j$ at random
  • Set:
    $$\hat{\beta}_j = \begin{cases} 
    (c_j + \lambda)/a_j & c_j < -\lambda \\
    0 & c_j \in [-\lambda, \lambda] \\
    (c_j - \lambda)/a_j & c_j > \lambda 
    \end{cases}$$
    $$\beta = (\hat{\beta}_1, \ldots, \hat{\beta}_k, \beta_{k+1}, \ldots, \beta_p)$$
  • Where:
    $$a_j = 2\sum_{i=1}^{N}(x_{ij})^2, \quad c_j = 2\sum_{i=1}^{N}x_{ij}(y_i - \beta'_{j}x_{ij})$$
    
    cost per iteration $\to O(N)$

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

• Analysis works for LASSO, L1 regularized logistic regression, and other objectives!

• For (coordinate-wise) strongly convex functions:
  $$F(\beta + \Delta \beta) \geq F(\beta) + \Delta \beta' (DF(\beta)) + \frac{\gamma (\Delta \beta)^2}{2}$$

• Theorem:
  – Starting from
  – After $T$ iterations
  $$E[ F(\beta_T) ] - F(\beta^*) \leq \frac{\gamma \| \beta^* \|^2 + F(\beta_0)}{T}$$
  
  – Where $E[ ]$ is with random coordinate choices of SCD

• Natural question: How does SCD & SGD convergence rates differ?
**Shooting: Sequential SCD**

**Lasso:** \( \min F(\beta) \) where \( F(\beta) = \| X \beta - y \|_2^2 + \lambda \| \beta \|_1 \)

**Stochastic Coordinate Descent (SCD)**
(e.g., Shalev-Shwartz & Tewari, 2009)
While not converged,
- Choose random coordinate \( j \),
- Update \( \beta_j \) (closed-form minimization)

** Shotgun: Parallel SCD ** [Bradley et al '11]

**Lasso:** \( \min F(\beta) \) where \( F(\beta) = \| X \beta - y \|_2^2 + \lambda \| \beta \|_1 \)

**Shotgun (Parallel SCD)**
While not converged,
- On each of \( P \) processors,
  - Choose random coordinate \( j \),
  - Update \( \beta_j \) (same as for Shooting)
Is SCD inherently sequential?

Lasso: \( \min F(\cdot) \) where \( F(\cdot) = \| X y \|_2^2 + \| \cdot \|_1 \)

Coordinate update:
\[
j \rightarrow j + j (\text{closed-form minimization})
\]

Collective update:
\[
\begin{pmatrix}
i \\
0 \\
0 \\
0 \\
j
\end{pmatrix}
\]

Convergence Analysis

Lasso: \( \min F(\cdot) \) where \( F(\cdot) = \| X y \|_2^2 + \| \cdot \|_1 \)

Theorem: Shotgun Convergence
Assume \( P < \frac{p}{r} + 1 \)
where \( r = \text{spectral radius of } XX^T \)

\[
E F(\cdot) = \frac{p}{TP} \left[ \frac{\frac{1}{2} \| * \|_2^2 + F(\cdot)}{TP} \right] + F(\cdot)
\]

Nice case: Unrelated features
= \_
\& \& P_{\text{max}} = \_

Bad case: Correlated features
= \_
\& \& P_{\text{max}} = \_ (at worst)
Stepping Back...

- Stochastic coordinate ascent
  - Optimization:
  - Parallel SCD:
  - Issue:
  - Solution:

- Natural counterpart:
  - Optimization:
  - Parallel
  - Issue:
  - Solution:

What you need to know

- Sparsistency
- Fused LASSO
- LASSO Solvers
  - LARS
  - A simple SCD for LASSO (Shooting)
    - Your HW, a more efficient implementation! 😊
    - Analysis of SCD
  - Parallel SCD (Shotgun)
Case Study 3: fMRI Prediction

“Scalable” LASSO Solvers:
Parallel SCD (Shotgun)
Parallel SGD
Averaging Solutions
ADMM

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Sham Kakade
May 5\textsuperscript{th}, 2016

©Sham Kakade 2016

Stepping Back...

• Stochastic coordinate ascent
  – Optimization:
  – Parallel SCD:
  – Issue:
  – Solution:

• Natural counterpart:
  – Optimization:
  – Parallel
  – Issue:
  – Solution:

©Sham Kakade 2016
Parallel SGD with No Locks
[e.g., Hogwild!, Niu et al. '11]

• Each processor in parallel:
  – Pick data point $i$ at random
  – For $j = 1...p$:

  \[
  \beta_j \leftarrow \beta_j - \eta \cdot (D^T (x_i, \beta))_j
  \]

• Assume atomicity of:

\[
\beta_j \leftarrow \beta_j + \eta \text{ considering } \nabla f(x, \beta)
\]

Addressing Interference in Parallel SGD

• Key issues:
  – Old gradients
  – Processors overwrite each other’s work

• Nonetheless:
  – Can achieve convergence and some parallel speedups
  – Proof uses weak interactions, but through sparsity of data points
Problem with Parallel SCD and SGD

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

- Works well on shared memory machines

- Very difficult to implement efficiently in distributed memory

- 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 & $P$ machines
- Stochastic optimization problem:
  \[ \min_{\beta} F(\beta) = \frac{1}{N} \sum_{i=1}^{N} f(x_i, \beta) \]
- Distribute data:
  \[ \text{randomly distribute data over } P \text{ machines} \]
- Solve problems independently
  \[ \beta_k = \arg\min_{\beta} \frac{1}{n_k} \sum_{x \in D_k} f(x, \beta) \]
- Merge solutions
  \[ \bar{\beta} = \frac{1}{P} \sum_{k=1}^{K} \beta_k \]
- Why should this work at all????

©Sham Kakade 2016
For Convex Functions...

- Convexity:

\[ F(\beta) \leq \frac{F(\beta_1) + F(\beta_2)}{2} \]

- Thus:

Hopefully...

- Convexity only guarantees:

- But, estimates from independent data!

Figure from John Duchi

©Sham Kakade 2016
Analysis of Distribute-then-Average [Zhang et al. ’12]

- Under some conditions, including strong convexity, lots of smoothness, etc.
- If all data were in one machine, converge at rate:
  \[ E[\|\beta - \beta^*\|^2] = O\left(\frac{1}{n}\right) \]

- With \( P \) machines, converge at a rate:
  \[ E[\|\beta - \beta^*\|^2] = O\left(\frac{1}{nP} + \frac{1}{n^2}\right) \]

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:

- 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
Alternating Directions Method of Multipliers (ADMM)

- A tool for solving convex problems with separable objectives:
  \[ \min_x \left\{ f(x) + g(x) \right\} \]

- LASSO example:
  \[ \min_{\beta} \left\{ \|y - x \beta\|_2^2 + \lambda \|\beta\|_1 \right\} \]
  
- Know how to minimize \( f(\beta) \) or \( g(\beta) \) separately

ADMM Insight

- Try this instead:
  \[ \min_{x, z} \left\{ f(x) + g(z) \right\} \quad \text{s.t.} \quad x = z \]

- Solve using method of multipliers
- Define the augmented Lagrangian:
  \[ \min_{x, z, y} \left\{ f(x) + g(z) + y^T (x - z) + \frac{\lambda}{2} \|x - z\|_2^2 \right\} \]
  
  - Issue: L2 penalty destroys separability of Lagrangian
  - Solution: Replace minimization over (x, z) by alternating minimization
ADMM Algorithm

- Augmented Lagrangian:

\[ L_\rho(x, z, y) = f(x) + g(z) + y^T(x - z) + \frac{\rho}{2}||x - z||_2^2 \]

- Alternate between:

1. \( x \leftarrow \arg \min_x \{ L_\rho(x, z, y) \} \)
2. \( z \leftarrow \arg \min_z \{ L_\rho(x, z, y) \} \)
3. \( y \leftarrow y + \frac{\rho}{2} (x - z) \)

ADMM for LASSO

- Objective:

\[ L_\rho(x, z, y) = f(x) + g(z) + y^T(x - z) + \frac{\rho}{2}||x - z||_2^2 \]

- Augmented Lagrangian:

\[ L_\rho(\beta, z, a) = \frac{1}{2} \|y - X\beta\|_2^2 + \frac{1}{2} \|z\|_1 + \frac{\rho}{2} \|\beta - z\|_2^2 \]

- Alternate between:

1. \( \beta \leftarrow \arg \min_\beta \{ L_\rho(\beta, z, a) \} \)
2. \( z \leftarrow \arg \min_z \{ L_\rho(\beta, z, a) \} \)
3. \( a \leftarrow a + \rho (\beta - z) \)
ADMM Wrap-Up

• When does ADMM converge?
  — Under very mild conditions
  — Basically, $f$ and $g$ must be convex

• ADMM is useful in cases where
  — $f(x) + g(x)$ is challenging to solve due to coupling
  — We can minimize
    • $f(x) + (x-a)^2$
    • $g(x) + (x-a)^2$

• Reference

What you need to know

• 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

• ADMM
  — General idea
  — Application to LASSO