Parallel Block Coordinate Descent Methods for Huge-Scale Partially Separable Optimization Problems

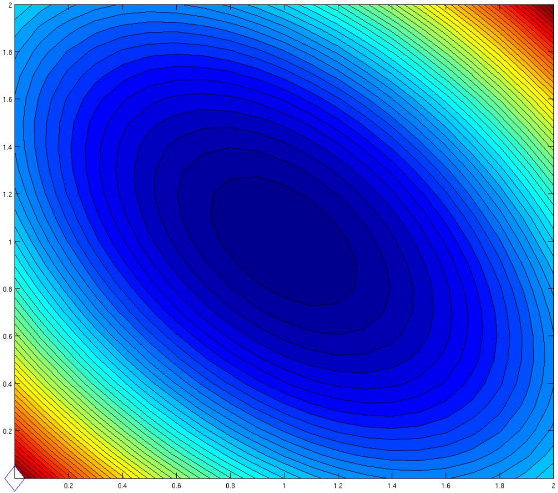
Peter Richtárik

School of Mathematics
The University of Edinburgh

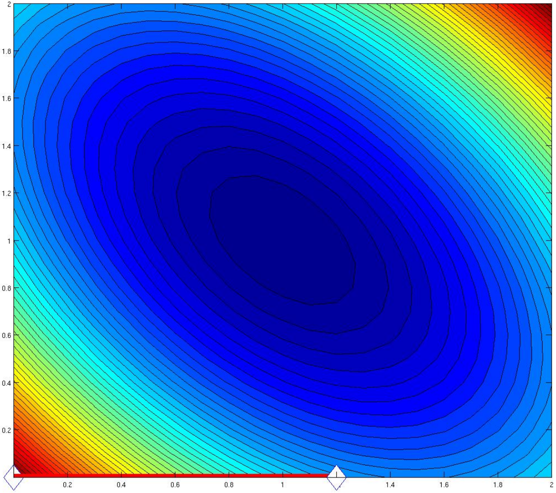
Joint work with Martin Takáč (Edinburgh)

INFORMS Phoenix ⋄ Oct 15, 2012

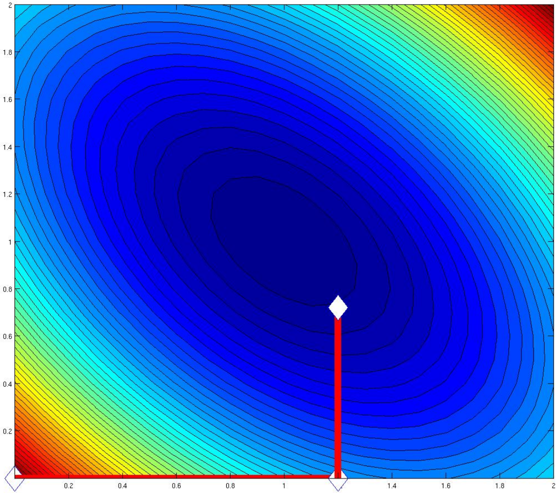
Serial Randomized Coordinate Descent: a 2D example



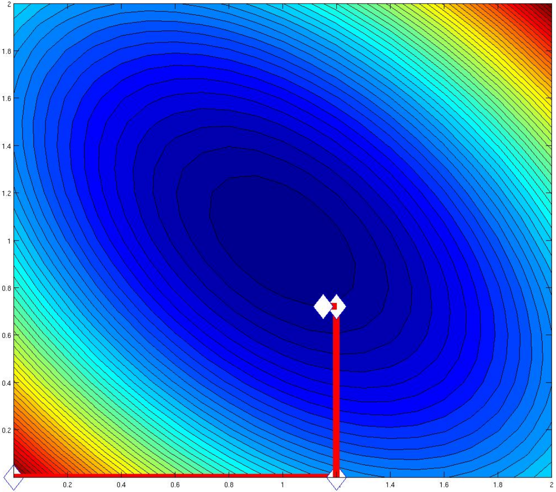
Serial Randomized Coordinate Descent: a 2D example



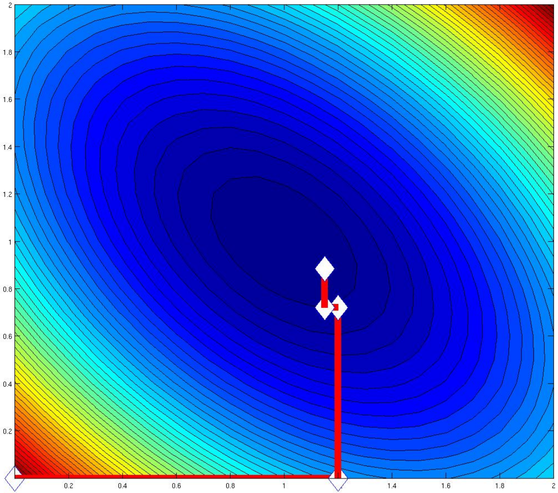
Serial Randomized Coordinate Descent: a 2D example



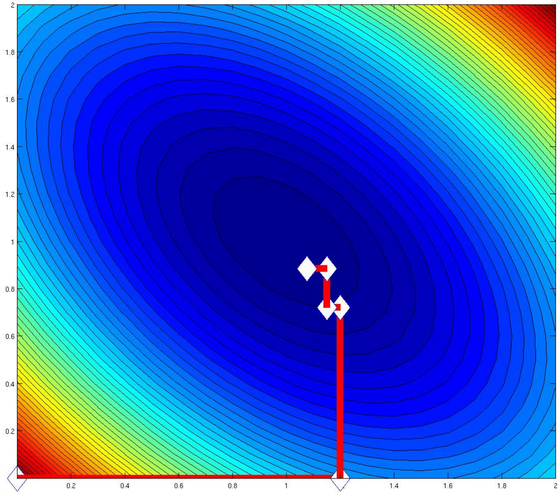
Serial Randomized Coordinate Descent: a 2D example



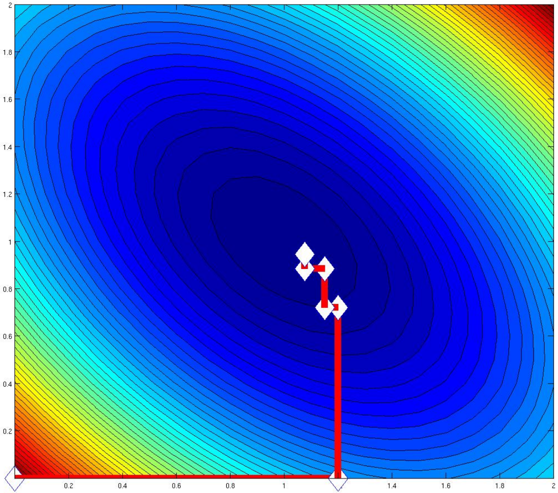
Serial Randomized Coordinate Descent: a 2D example



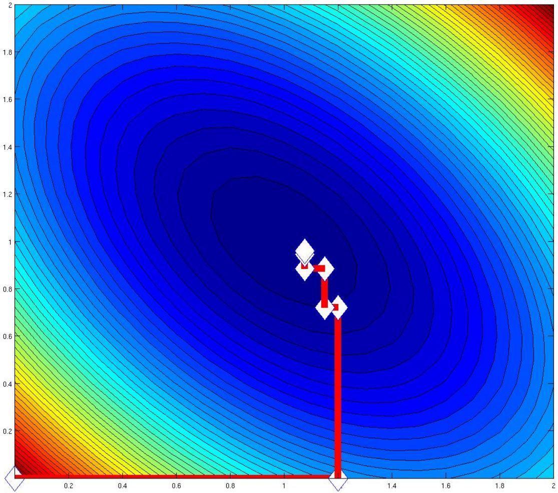
Serial Randomized Coordinate Descent: a 2D example



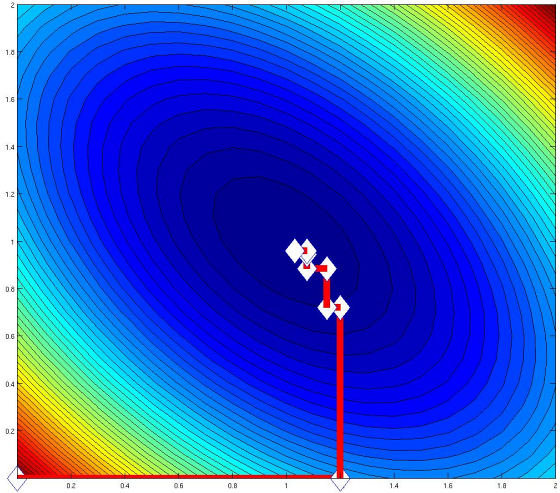
Serial Randomized Coordinate Descent: a 2D example



Serial Randomized Coordinate Descent: a 2D example



Serial Randomized Coordinate Descent: a 2D example



Part I.

4 Boring Slides

BS1: Blocks

$x \in \mathbf{R}^N$ is partitioned into n blocks/groups of variables:

$$x^{(1)}, \ldots, x^{(n)}, \qquad x^{(i)} \in \mathbf{R}^{N_i}, \qquad \sum_{i=1}^{n} N_i = N$$

$$x = \sum_{i=1}^{n} U_i x^{(i)}$$

Example: $N = 5, n = 2$

$$x = (x_1, x_2, x_3, x_4, x_5)^T = (\underbrace{(x_1, x_3)}_{x^{(1)} \in \mathbf{R}^2}, \underbrace{(x_2, x_4, x_5)}_{x^{(2)} \in \mathbf{R}^3})$$

$$U_1 \underbrace{\begin{pmatrix} x_1 \\ x_3 \end{pmatrix}}_{x^{(1)} \in \mathbf{R}^2} = \begin{pmatrix} x_1 \\ 0 \\ x_3 \\ 0 \\ 0 \end{pmatrix}, \qquad U_2 \underbrace{\begin{pmatrix} x_2 \\ x_4 \\ x_5 \end{pmatrix}}_{x^{(2)} \in \mathbf{R}^3} = \begin{pmatrix} 0 \\ x_2 \\ 0 \\ x_4 \\ x_5 \end{pmatrix}$$

$$x = U_1 x^{(1)} + U_2 x^{(2)}, \qquad x^{(i)} = U_i^T x$$

BS2: The Problem

$$\min_{x \in \mathbf{R}^N} \{F(x) \stackrel{\text{def}}{=} f(x) + \Psi(x)\}, \qquad \text{(P)}$$

Assumptions:

- f is convex and smooth
- f is (block/group) partially separable:

$$f = \sum_J f_J, \text{ each } f_J \text{ depends on a few blocks } x^{(i)} \text{ only}$$

- Ψ is convex, nonsmooth and simple
- Ψ is (block/group) separable: $\Psi(x) = \sum_{i=1}^n \Psi_i(x^{(i)})$

$\Psi(x)$	meaning
0	smooth minimization
$\lambda \\|x\\|_1$ $(N = n)$	term inducing sparsity
$\lambda \sum_{i=1}^n \\|x^{(i)}\\|_2$	group-sparsity term LASSO
indicator function of a set	(block) separable constraints

- n is huge ($n \approx 10^9$)

BS3: Assumption: Smoothness of f

Definition (Block norms and norm in $\mathbf{R}^N$)

We measure the size of block $h^{(i)} \in \mathbf{R}^{N_i}$ of $h \in \mathbf{R}^N$, using a dedicated norm: $\|h^{(i)}\|_{(i)}$, $i = 1, \dots, n$, and measure $x \in \mathbf{R}^N$ via

$$\|x\|_W^2 \stackrel{\text{def}}{=} \sum_{i=1}^{n} w_i \|x^{(i)}\|_{(i)}^2.$$

Assumption (Smoothness of f)

The gradient of f is block coordinate-wise Lipschitz with positive constants $L_1, L_2, \dots, L_n$:

$$\|\nabla_i f(x + U_i t) - \nabla_i f(x)\|_{(i)}^* \leq L_i \|t\|_{(i)}, \; x \in \mathbf{R}^N, \; t \in \mathbf{R}^{N_i}, \; i = 1, \dots, n,$$

where

$$\nabla_i f(x) \stackrel{\text{def}}{=} (\nabla f(x))^{(i)} = U_i^T \nabla f(x) \in \mathbf{R}^{N_i}.$$

BS4: Applications

- **Machine Learning:** L_1 regularized linear support vector machines (SVM) with various (smooth) loss functions
- **Ranking:** Google problem
- **Engineering:** truss topology design
- **Statistics:** least squares, ridge regression, sparse least squares (LASSO), elastic net, group lasso, (sparse) group LASSO
- **Optimal Design:** c and A optimality

Part II.

PR-BCD: Parallel Randomized Block Coordinate Descent

[RT2012b] P. R. and Martin Takáč
Parallel coordinate descent methods for big data problems

Parallel Randomized Block Coordinate Descent

Algorithm PR-BCD

Input: Choose initial point $x_0 \in \mathbf{R}^N$

for $k = 0, 1, 2, \ldots$ **do**

1. Choose random subset of blocks (sampling) $\hat{S} \subset \{1, 2, \ldots, n\}$
2. In parallel for $i \in \hat{S}$ do:
   (a) Compute block update: $h^{(i)}(x_k) \in \mathbf{R}^{N_i}$
   (b) Update the block: $x_k^{(i)} = x_k^{(i)} + h^{(i)}(x_k)$
   (c) $x_{k+1} = x_k$

end for

Leads to different algorithms for different samplings $\hat{S}$:

- $\hat{S} = \{i\}, \; i = 1, 2, \ldots, n$, with probability $\frac{1}{n}$ (serial uniform)
- $\hat{S} = \{i\}, \; i = 1, 2, \ldots, n$, with probability p_i (serial non-uniform)
- $\hat{S} = \{1, 2, \ldots, n\}$ with probability 1 (fully parallel)
- $\hat{S}$ = **many interesting choices in between!**

ESO: Expected Separable Overapproximation

Definition (ESO)

f admits an (α, β)-ESO with respect to sampling $\hat{S}$ if

$$\mathbf{E}[f(x + h_{[\hat{S}]})] \leq f(x) + \alpha \left(\langle \nabla f(x), h \rangle + \frac{\beta}{2} \|h\|_L^2 \right).$$

- The overapproximation is (block) separable in h:

$$\langle \nabla f(x), h \rangle + \frac{\beta}{2} \|h\|_L^2 = \sum_{i=1}^{n} \left(\langle \nabla_i f(x), h^{(i)} \rangle + \frac{\beta L_i}{2} \|h^{(i)}\|_{(i)}^2 \right).$$

Main Result: Iteration Complexity

Theorem

Assume f admits an (α, β)-ESO with respect to sampling $\hat{S}$. Choose $x_0 \in \mathbf{R}^N$, confidence level $0 < \rho < 1$, target accuracy

$$0 < \epsilon < \min\{2\left(\frac{\beta}{\alpha}\right)\mathcal{R}_L^2(x_0, x^*), F(x_0) - F^*\}$$

and iteration counter

$$k \geq \left(\frac{\beta}{\alpha}\right)\left(\frac{2\mathcal{R}_L^2(x_0, x^*)}{\epsilon} \log \frac{F(x_0) - F^*}{\epsilon\rho}\right).$$

If x_k is the random point generated by PR-BCD as applied to F, then

$$\mathbf{P}(F(x_k) - F^* \leq \epsilon) \geq 1 - \rho$$

Remarks:

- Specialized results for smooth functions, strongly convex functions, ...

ESO: Computing α, β

Constants α, β, and hence the complexity factor $\frac{\beta}{\alpha}$, depend on

- sampling $\hat{S}$
- function f

$$\frac{\beta}{\alpha} \quad \begin{cases} = n, & \text{serial uniform } \hat{S} \text{ [RT2011a]} \\ \ll n, & \text{doubly uniform } \hat{S}, \text{ (block) partially separable } f \text{ [RT2012b]} \end{cases}$$

[RT2011a] P. R. and Martin Takáč
Iteration complexity of randomized block coordinate descent methods for minimizing a composite function

[RT2012b] P. R. and Martin Takáč
Parallel block coordinate descent methods for huge-scale partially separable optimization problems

Doubly Uniform Samplings

Definition

Sampling $\hat{S}$ is doubly uniform if for all $S_1, S_2 \subset \{1, 2, \ldots, n\}$:

$$|S_1| = |S_2| \quad \Longrightarrow \quad \mathbf{P}(\hat{S} = S_1) = \mathbf{P}(\hat{S} = S_2).$$

Doubly Uniform Samplings

Definition

Sampling $\hat{S}$ is doubly uniform if for all $S_1, S_2 \subset \{1, 2, \ldots, n\}$:

$$|S_1| = |S_2| \quad \Longrightarrow \quad \mathbf{P}(\hat{S} = S_1) = \mathbf{P}(\hat{S} = S_2).$$

Remark: A doubly uniform sampling is uniquely characterized by

$$q_k \stackrel{\text{def}}{=} \mathbf{P}(|\hat{S}| = k), \qquad k = 0, 1, 2, \ldots, n.$$

Indeed, we have

$$\mathbf{P}(\hat{S} = S) = \frac{q_{|S|}}{\binom{n}{|S|}}$$

Doubly Uniform Samplings

Definition

Sampling $\hat{S}$ is doubly uniform if for all $S_1, S_2 \subset \{1, 2, \ldots, n\}$:

$$|S_1| = |S_2| \quad \Longrightarrow \quad \mathbf{P}(\hat{S} = S_1) = \mathbf{P}(\hat{S} = S_2).$$

Remark: A doubly uniform sampling is uniquely characterized by

$$q_k \overset{\text{def}}{=} \mathbf{P}(|\hat{S}| = k), \qquad k = 0, 1, 2, \ldots, n.$$

Indeed, we have

$$\mathbf{P}(\hat{S} = S) = \frac{q_{|S|}}{\binom{n}{|S|}}$$

Example: $n = 4$

- $\mathbf{P}(\emptyset) = q_0$
- $\mathbf{P}(\{1\}) = \mathbf{P}(\{2\}) = \mathbf{P}(\{3\}) = \mathbf{P}(\{4\}) = \frac{q_1}{4}$
- $\mathbf{P}(\{1, 2\}) = \mathbf{P}(\{1, 3\}) = \mathbf{P}(\{1, 4\}) = \mathbf{P}(\{2, 3\}) = \mathbf{P}(\{2, 4\}) = \mathbf{P}(\{3, 4\}) = \frac{q_2}{6}$
- $\mathbf{P}(\{1, 2, 3\}) = \mathbf{P}(\{1, 2, 4\}) = \mathbf{P}(\{1, 3, 4\}) = \mathbf{P}(\{2, 3, 4\}) = \frac{q_3}{4}$
- $\mathbf{P}(\{1, 2, 3, 4\}) = q_4$

What Can We Model with Doubly Uniform Samplings?

Recall that

$$q_k = \mathbf{P}(|\hat{S}| = k)$$

Two Examples of Computing Environments:

- τ reliable processors: $q_\tau = 1$
- τ unreliable/busy processors with each giving an answer, independently, with probability p:

$$q_k = \binom{\tau}{k} p^k (1-p)^k, \qquad k = 0, 1, \ldots, \tau$$

Partial Separability of f

Definition

Function f is (block) partially separable of degree ω if

$$f(x) = \sum_{J \in \mathcal{J}} f_J(x), \qquad \mathcal{J} \text{ consists of (some) subsets of } \{1, 2, \dots, n\}$$

- f_J depends on blocks $x^{(i)}$ for $i \in J$ only
- $\max_{J \in \mathcal{J}} |J| \leq \omega$

Partial Separability of f

Definition

Function f is (block) partially separable of degree ω if

$$f(x) = \sum_{J \in \mathcal{J}} f_J(x), \qquad \mathcal{J} \text{ consists of (some) subsets of } \{1, 2, \ldots, n\}$$

- f_J depends on blocks $x^{(i)}$ for $i \in J$ only
- $\max_{J \in \mathcal{J}} |J| \leq \omega$

Observe that:

- $1 \leq \omega \leq n$

Partial Separability of f

Definition

Function f is (block) partially separable of degree ω if

$$f(x) = \sum_{J \in \mathcal{J}} f_J(x), \qquad \mathcal{J} \text{ consists of (some) subsets of } \{1, 2, \ldots, n\}$$

- f_J depends on blocks $x^{(i)}$ for $i \in J$ only
- $\max_{J \in \mathcal{J}} |J| \leq \omega$

Observe that:

- $1 \leq \omega \leq n$
- If $\omega = 1$, f is (block) separable (as Ψ) and hence the main optimization problem can be decomposed into n independent problems

Partial Separability of f

Definition

Function f is (block) partially separable of degree ω if

$$f(x) = \sum_{J \in \mathcal{J}} f_J(x), \qquad \mathcal{J} \text{ consists of (some) subsets of } \{1, 2, \ldots, n\}$$

- f_J depends on blocks $x^{(i)}$ for $i \in J$ only
- $\max_{J \in \mathcal{J}} |J| \leq \omega$

Observe that:

- $1 \leq \omega \leq n$
- If $\omega = 1$, f is (block) separable (as Ψ) and hence the main optimization problem can be decomposed into n independent problems

Example: $f(x) = \frac{1}{2}\|Ax - b\|^2 = \sum_j \underbrace{\frac{1}{2}(A_j^T x - b_j)^2}_{f_j(x)}$,

- $\omega = \max \#$ of nonzeros in A_j (a row of A)

ESO for Partially Separable f and Doubly Uniform $\hat{S}$

Theorem

Assume that

- f is partially separable of degree ω
- $\hat{S}$ is a doubly uniform sampling

Then f admits an (α, β)-ESO with respect to $\hat{S}$ with

$$\alpha = \frac{\mathbf{E}[|\hat{S}|]}{n}, \qquad \beta = 1 + \frac{(\omega - 1)\left(\frac{\mathbf{E}[|\hat{S}|^2]}{\mathbf{E}[|\hat{S}|]} - 1\right)}{\max(1, n-1)}.$$

Remarks:

- We have computed ESO for some other samplings as well, but will not talk about them here

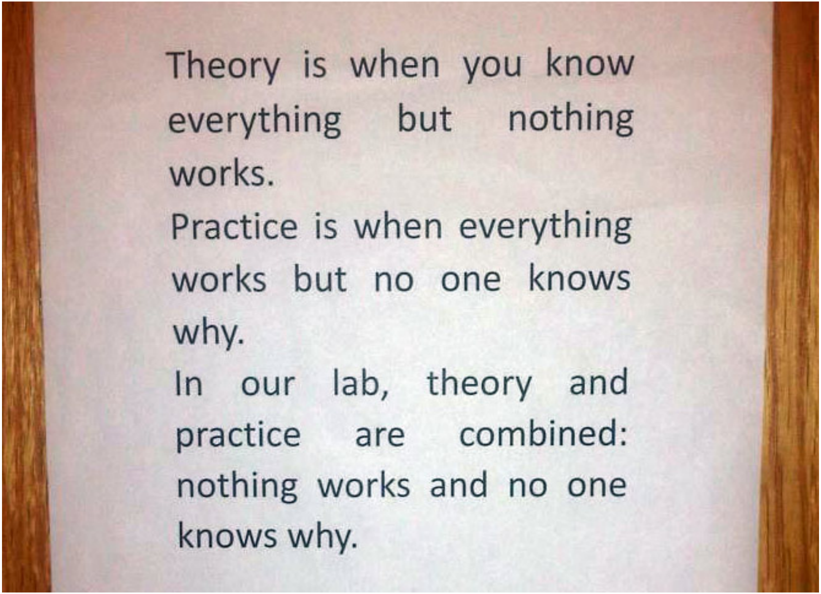
ESO Coefficients

Sampling $\hat{S}$	α	β
Doubly Uniform (DU)	$\frac{\mathbf{E}[\lvert\hat{S}\rvert]}{n}$	$1 + \frac{(\omega-1)\left(\frac{\mathbf{E}[\lvert\hat{S}\rvert^2]}{\mathbf{E}[\lvert\hat{S}\rvert]}-1\right)}{\max(1,n-1)}$
DU unreliable	$\frac{\tau p}{n}$	$1 + \frac{(\omega-1)(\tau-1)p}{\max(1,n-1)}$
DU reliable	$\frac{\tau}{n}$	$1 + \frac{(\omega-1)(\tau-1)}{\max(1,n-1)}$
DU fully parallel	1	ω
DU serial	$\frac{1}{n}$	1

Parallelization Speedup for DU Samplings

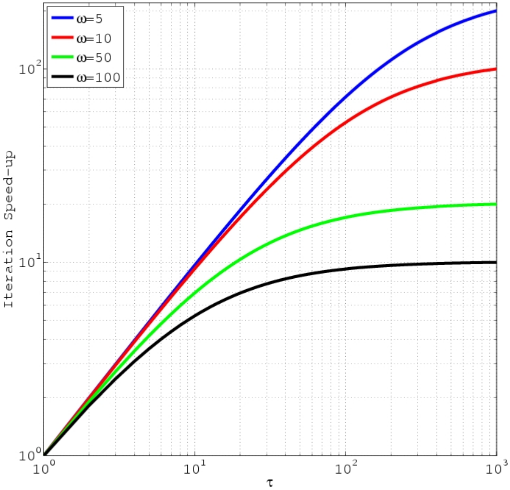
Sampling $\hat{S}$	$\frac{\beta}{\alpha}$ of sampling / $\frac{\beta}{\alpha}$ of DU serial
Doubly Uniform (DU)	$\dfrac{\max(1,n-1)}{\left(\frac{\mathbf{E}[\lvert\hat{S}\rvert^2]}{(\mathbf{E}\lvert\hat{S}\rvert)^2}-\frac{1}{\mathbf{E}[\lvert\hat{S}\rvert]}\right)(\omega-1)+\frac{1}{\mathbf{E}[\lvert\hat{S}\rvert]}\max(1,n-1)}$
DU unreliable	$\dfrac{\max(1,n-1)}{\left(1-\frac{1}{\tau}\right)(\omega-1)+\frac{1}{\tau}\frac{\max(1,n-1)}{p}}$
DU reliable	$\dfrac{\max(1,n-1)}{\left(1-\frac{1}{\tau}\right)(\omega-1)+\frac{1}{\tau}\max(1,n-1)}$
DU fully parallel	$\frac{n}{\omega}$
DU serial	1

Parallelization Speedup: Theory vs Practice



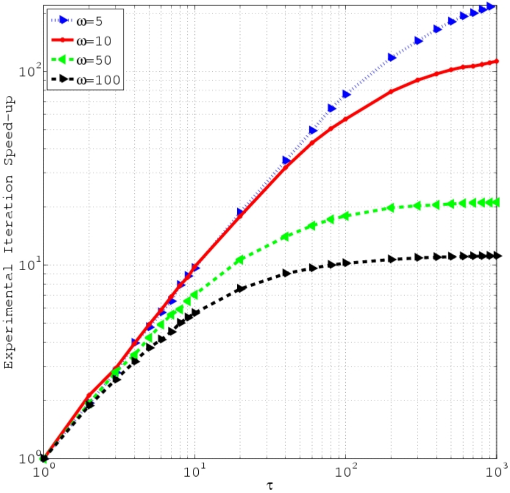
Parallelization Speedup (DU reliable): Theory

$$F(x) = f(x) = \frac{1}{2}\|Ax - b\|_2^2, \qquad A \in \mathbf{R}^{3000\times 1000}$$



Parallelization Speedup (DU reliable): Experiment

$$F(x) = f(x) = \frac{1}{2}\|Ax - b\|_2^2, \qquad A \in \mathbf{R}^{3000\times 1000}$$



A Problem with Billion Variables

LASSO problem:

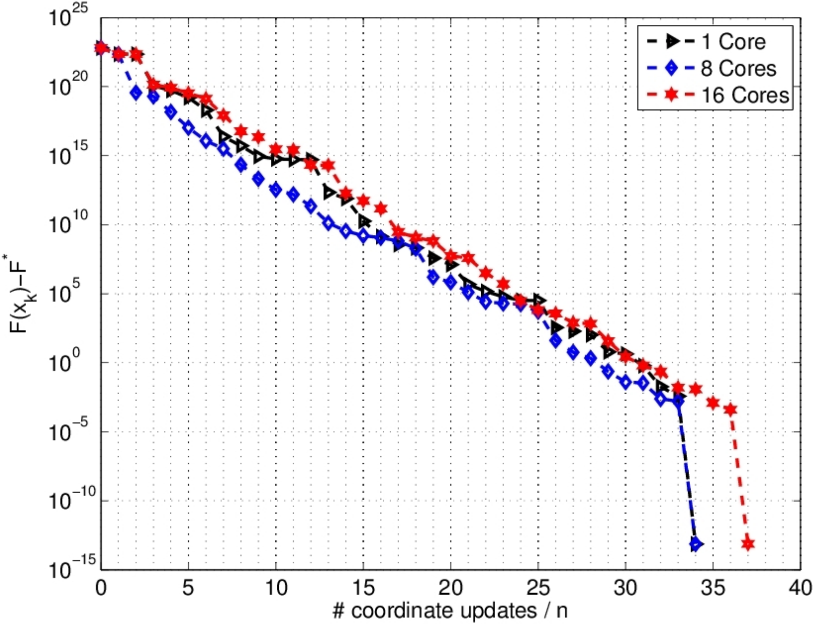
$$F(x) = \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1$$

The instance:

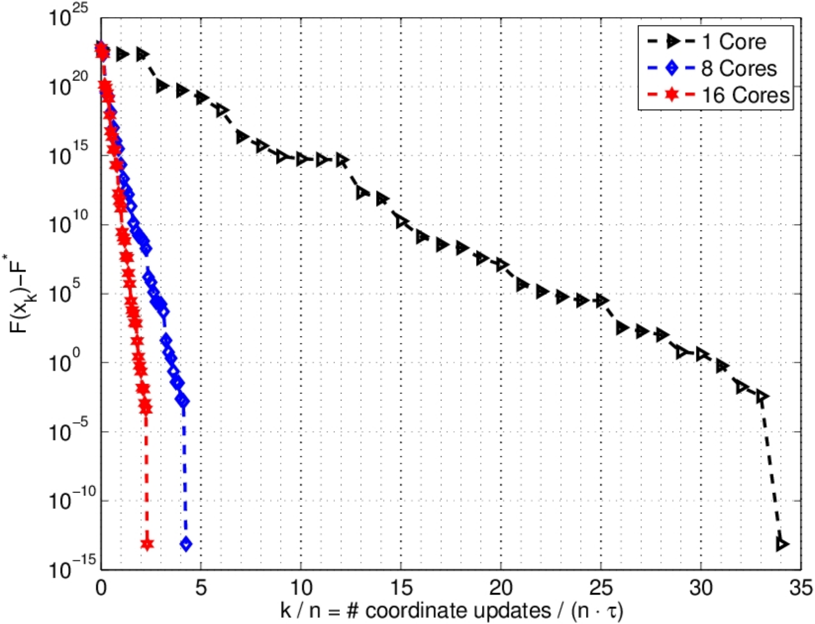
- A has
  - $m = 2 \times 10^9$ rows
  - $n = 10^9$ columns ($= \#$ of variables)
  - exactly 20 nonzeros in each column
  - on average 10 and at most 35 nonzeros in each row ($\omega = 35$)
- optimal solution x^* has 10^5 nonzeros

Solver: Asynchronous parallel coordinate descent method with nice (=reliable DU) sampling and $\tau = 1, 8, 16$ cores

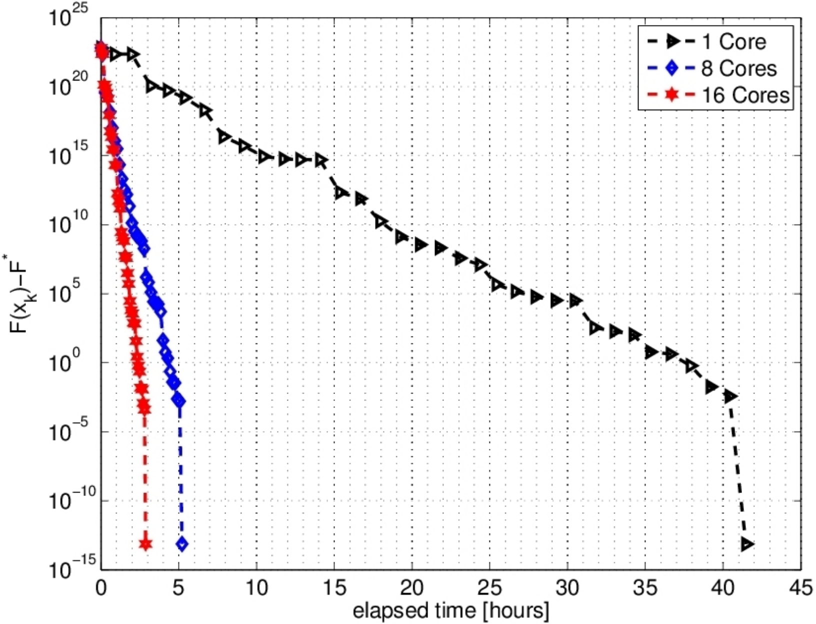
Coordinate Updates / n



Iterations / n



Wall Time



10^{25}
10^{20}
10^{15}
10^{10}
10^{5}
10^{0}
10^{-5}
10^{-10}
10^{-15}
$F(x_k)-F^*$
0
5
10
15
20
25
30
35
40
45
elapsed time [hours]
1 Core
8 Cores
16 Cores

Billion Variables — 1, 8 and 16 Cores

	$F(x_k) - F^*$			Elapsed Time		
$(k \cdot \tau)/n$	1 core	8 cores	16 cores	1 core	8 cores	16 cores
0	6.27e+22	6.27e+22	6.27e+22	0.00	0.00	0.00
1	2.24e+22	2.24e+22	2.24e+22	0.89	0.11	0.06
2	2.25e+22	3.64e+19	2.24e+22	1.97	0.27	0.14
3	1.15e+20	1.94e+19	1.37e+20	3.20	0.43	0.21
4	5.25e+19	1.42e+18	8.19e+19	4.28	0.58	0.29
5	1.59e+19	1.05e+17	3.37e+19	5.37	0.73	0.37
6	1.97e+18	1.17e+16	1.33e+19	6.64	0.89	0.45
7	2.40e+16	3.18e+15	8.39e+17	7.87	1.04	0.53
⋮	⋮	⋮	⋮	⋮	⋮	⋮
26	3.49e+02	4.11e+01	3.68e+03	31.71	3.99	2.02
27	1.92e+02	5.70e+00	7.77e+02	33.00	4.14	2.10
28	1.07e+02	2.14e+00	6.69e+02	34.23	4.30	2.17
29	6.18e+00	2.35e-01	3.64e+01	35.31	4.45	2.25
30	4.31e+00	4.03e-02	2.74e+00	36.60	4.60	2.33
31	6.17e-01	3.50e-02	6.20e-01	37.90	4.75	2.41
32	1.83e-02	2.41e-03	2.34e-01	39.17	4.91	2.48
33	3.80e-03	1.63e-03	1.57e-02	40.39	5.06	2.56
34	7.28e-14	7.46e-14	1.20e-02	41.47	5.21	2.64
35	-	-	1.23e-03	-	-	2.72
36	-	-	3.99e-04	-	-	2.80
37	-	-	7.46e-14	-	-	2.87