Randomized Parallel Algorithms in Optimization

Stephen Wright

University of Wisconsin-Madison

July 2013

Collaborators @ UW-Madison

Victor Bittorf
Ji Liu
Ben Recht ($\rightarrow$ Berkeley)
Chris Ré ($\rightarrow$ Stanford)
Krishna Sridhar

Motivation

Why look at old, slow, simple algorithms?

- Often good for machine learning and big-data applications.
- Often a good fit for modern computers (multicore, NUMA, clusters) — parallel, asynchronous versions are possible.
- Easy to implement.
- Interesting new analysis, tied to plausible models of parallel computation and data.

"Asynchronicity is the key to speedup on modern architectures," says Bill Gropp (SIAM CS&E Plenary, 2013).

Stochastic Gradient

Typical form of f in learning problems:

$$f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x),$$

where each f_i is convex, and depends on a single item of data.

In classical SG, choose index $i_k \in \{1, 2, \ldots, m\}$ uniformly at random at iteration k, set

$$x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k), \quad \text{for some } \alpha_k > 0.$$

When f is strongly convex, the analysis of convergence of $E(\|x_k - x^*\|^2)$ is elementary (Nemirovski et al, 2009).

Averaging of iterates x_k and gradient estimates $\nabla f_{i_k}(x_k)$ (primal and dual averaging) can be applied to make behavior "smoother" and more robust.

Convergence of Classical SG

Define μ (convexity modulus), M, Lipschitz constant L:

$$\nabla f(x) - \nabla f(x')]^T (x - x') \geq \mu \|x - x'\|^2,$$

$$\frac{1}{m} \sum_{i=1}^{m} \|\nabla f_i(x)\|^2 \leq M^2,$$

$$\|\nabla f(x) - \nabla f(x')\| \leq L \|x - x'\|,$$

for all x, x' of interest. Obtain convergence in expected square error:

$$a_k := \frac{1}{2} E(\|x_k - x^*\|^2).$$

Elementary argument shows a recurrence:

$$a_{k+1} \leq (1 - 2\mu\alpha_k) a_k + \frac{1}{2} \alpha_k^2 M^2.$$

When we set $\alpha_k = 1/(k\mu)$, an inductive argument reveals a $1/k$ rate:

$$a_k \leq \frac{Q}{2k}, \qquad \text{for } Q := \max\left(\|x_1 - x^*\|^2, \frac{M^2}{\mu^2} \right).$$

Constant Step Size: $\alpha_k \equiv \alpha$

We can also obtain a "$1/k$ rate" for f strongly convex, for constant stepsize $\alpha_k \equiv \alpha$, when desired threshold ϵ for a_k is specified a priori.

From earlier analysis, have

$$a_{k+1} \leq (1 - 2\mu\alpha)a_k + \frac{1}{2}\alpha^2 M^2$$

which easily yields

$$a_k \leq (1 - 2\mu\alpha)^k a_0 + \frac{\alpha M^2}{4\mu}.$$

Given $\epsilon > 0$, choose α and K so that $a_k \leq \epsilon$ for all $k \geq K$. Choose so that both terms in the bound above are less than $\epsilon/2$:

$$\alpha := \frac{2\epsilon\mu}{M^2}, \qquad K := \frac{M^2}{4\epsilon\mu^2} \log\left(\frac{a_0}{2\epsilon}\right).$$

Parallel Stochastic Approximation

Several approaches tried for parallel stochastic approximation.

- **Dual Averaging (AIG):** Average gradient estimates evaluated in parallel on different cores. Requires message passing / synchronization (Dekel et al, 2011; Duchi et al, 2010).
- **Round-Robin (RR):** Cores evaluate ∇f_i in parallel and update centrally stored x in round-robin fashion. Requires synchronization (Langford et al, 2009).
- **Asynchronous:** HOGWILD!: Each core grabs the centrally-stored x and evaluates $\nabla f_i(x)$ for some random i, then writes the updates back into x (Niu et al, 2011). **Downpour SGD:** Similar idea for cluster (Dean et al, 2012).

HOGWILD!: Each processor runs independently:

1. Sample i_k from $\{1, 2, \ldots, m\}$;
2. Read current state of x from central memory, evalute $g := \nabla f_{i_k}(x)$;
3. **for** nonzero components g_v **do** $x_v \leftarrow x_v - \alpha g_v$;

HOGWILD! Convergence

- Updates can be "old" by the time they are applied, but we assume a bound of τ cycles on their age.
- Processors can overwrite each other's work, but sparsity of the ∇f_i helps — updates don't interfere too much with each other. Partial separability.

Define quantities that capture the interconnectedness of the functions f_i:

- ρ_i = number of indices j such that f_i and f_j depend on overlapping components of x.
- $\bar{\rho} = \sum_{i=1}^{m} \rho_i / m^2$: average rate of overlapping subvectors.

HOGWILD! Convergence (Niu, Recht, Ré, Wright, 2011; Richtarik ,2013)

Given $\epsilon \in (0, a_0 L)$, and setting

$$\alpha_k \equiv \frac{\mu\epsilon}{(1+2\tau\bar{\rho})LM^2m^2}$$

we have for

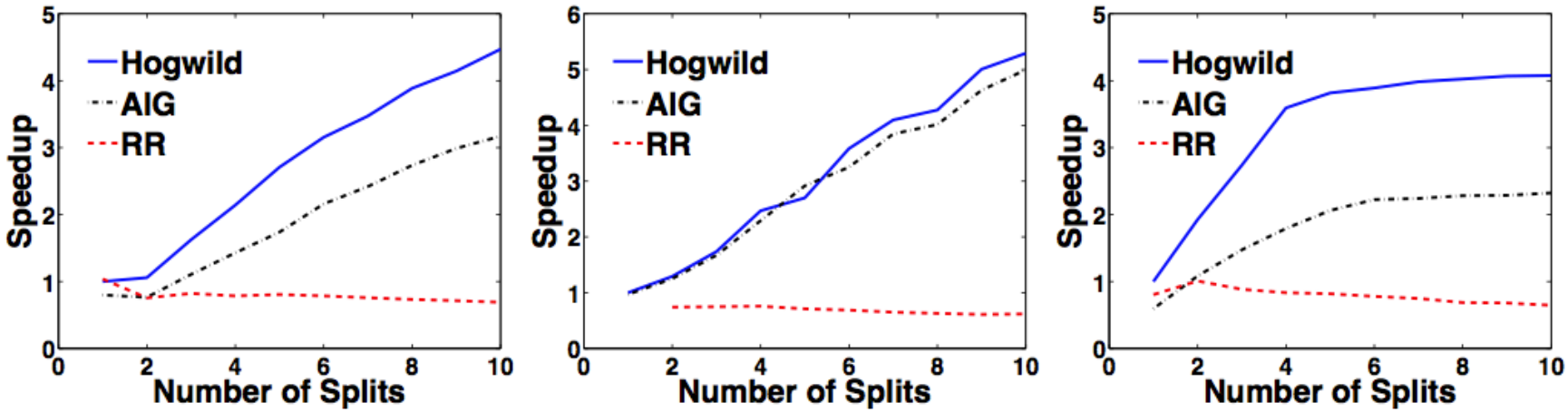
$$k \geq \frac{(1+2\tau\bar{\rho})LM^2m^2}{\mu^2\epsilon}\log\left(\frac{2La_0}{\epsilon}-1\right)$$

that

$$\min_{0\leq j\leq k} E(f(x_j)-f(x^*)) \leq \epsilon,$$

Recovers the sublinear $1/k$ convergence rate seen in regular SGD, with the delay τ and overlap measure ρ both appearing linearly.

HOGWILD! Performance



HOGWILD! compared with averaged gradient (AIG) and round-robin (RR). Experiments run on a 12-core machine in 2011. (10 cores used for gradient evaluations, 2 cores for data shuffling.)

HOGWILD! Performance

	data set	size (GB)	ρ	Δ	time (s)	speedup
SVM	RCV1	0.9	4.4E-01	1.0E+00	10	4.5
	Netflix	1.5	2.5E-03	2.3E-03	301	5.3
MC	KDD	3.9	3.0E-03	1.8E-03	878	5.2
	JUMBO	30	2.6E-07	1.4E-07	9,454	6.8
	DBLife	0.003	8.6E-03	4.3E-03	230	8.8
CUTS	Abdomen	18	9.2E-04	9.2E-04	1,181	4.1

Stochastic Coordinate Descent

Consider $\min f(x)$, where $f : \mathbb{R}^n \to \mathbb{R}^n$ is smooth and convex.

In SCD, at iteration j, pick an index $i_j \in \{1, 2, \ldots, n\}$ and take a step in component x_{i_j}. (We'll consider a short steepest-descent step.)

Many extensions possible:

- Block SCD
- Regularized: $f(x) + \psi(x)$ (need ψ to be separable to align with choice of coordinate blocks).
- Constraints: $\min f(x)$ s.t. $x \in \Omega$.

Stochastic Coordinate Descent: Key Constants

For simplicity, describe single-coordinate case (not blocks).

Need (many!) constants that characterize the problem:

- e_i: ith unit coordinate vector in $\mathbb{R}^n$;
- L_i: component Lipschitz constant ("(i,i) element of Hessian")

$$\|\nabla_i f(x + te_i) - \nabla_i f(x)\| \leq L_i t;$$

- $L_{\max} = \max_{i=1,2,\ldots,n} L_i$ ("max diagonal of Hessian");
- $L_{\text{res}} =$ restricted Lipschitz constant: $\|\nabla f(x) - \nabla f(x + te_i)\| \leq L_{\text{res}} t$ ("max row-norm of Hessian")
- $R = \sup_k \operatorname{dist}(x_k, \mathcal{S})$: maximum distance of iterates from solution set.
- $\|w\|_L := \left(\sum_{i=1}^n L_i w_i^2\right)^{1/2}$;
- $\mathcal{R}_L(x_0) := \max_y \max_{x^* \in \mathcal{S}} \{\|y - x^*\| : f(y) \leq f(x_o)\}$.

Diagonalicity of Hessian

The ratio $L_{\text{res}}/L_{\max}$ is particularly important — it measures the degree of diagonal dominance in the Hessian $\nabla^2 f(x)$ (Diagonalicity).

By convexity, we have

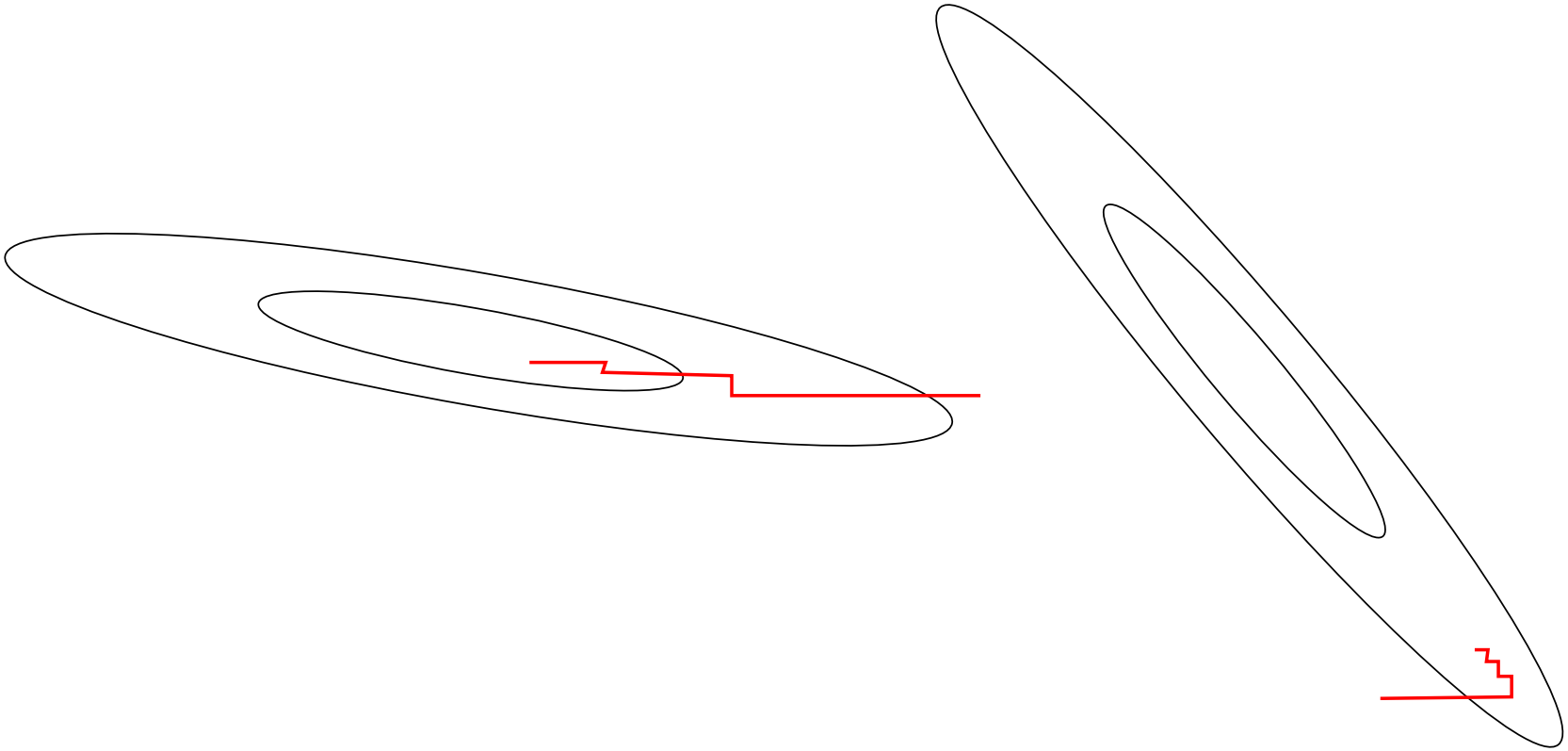
$$1 \leq \frac{L_{\text{res}}}{L_{\max}} \leq \sqrt{n}.$$

Closer to 1 if Hessian is nearly diagonally dominant (eigenvectors close to principal coordinate axes). Closer to $\sqrt{n}$ otherwise.

If A is $m \times n$ Gaussian random matrix and $f(x) = (1/2)\|Ax - b\|_2^2$, the ratio is $1 + O(\sqrt{n/m})$ (good case!)

Smaller $L_{\text{res}}/L_{\max} \Rightarrow$ easier to solve with coordinate descent!

Diagonalicity illustrated



Stochastic (Serial) SCD: (Richtarik and Takac, 2012) (Nesterov, 2012)

Iteration j:

- Choose partition $i_j \in \{1, 2, \ldots, m\}$ with equal probability;
- Set $x_{j+1} = x_j - \nabla_{i_j} f(x_j)/L_{i_j}$.

For convex f, have high probability convergence of f to within a specified threshold ϵ of $f(x^*)$ in $O(1/\epsilon)$ iterations.

Given desired precision ϵ and error prob ρ, define

$$K := \frac{2n\mathcal{R}_L^2(x_0)}{\epsilon} \log \frac{f(x_0) - f^*}{\epsilon\rho},$$

Have high-probability convergence in K iterations:

$$P(f(x_j) - f^* \leq \epsilon) \geq 1 - \rho, \quad \text{for } j \geq K.$$

If f is strongly convex with respect to $\|\cdot\|_L$, with modulus μ_L in this norm, there is expected convergence at a linear rate:

$$E[f(x_j) - f^*] \leq \left(1 - \frac{\mu_L}{4n}\right)^j (f(x_0) - f^*).$$

Asynchronous Unconstrained SCD

Similar computation model to HOGWILD!: asynchronous with maximum delay τ. Consider single-coordinate form.

At each iteration j:

- Choose i_j with equal probability from $\{1, 2, \ldots, n\}$;
- Update the i_j component:

$$x_{j+1} = x_j - \frac{\gamma}{L_{\max}} \nabla_{i_j} f(x_{k(j)}),$$

where $k(j)$ is some iterate prior to j but no more than τ cycles old: $j - k(j) \leq \tau$. Here γ is a constant steplength.

Each core runs this process concurrently and asynchronously.

Asynchronous USCD: Sublinear $(1/k)$ Rate (Richtarik, 2012)

In the case of (single-coordinate) asynchronous SGD, given tolerance ϵ, set

$$\gamma \equiv \frac{\mu L_{\max} n \epsilon}{(n + 2\tau) L^2 M^2}.$$

Then for

$$K \geq \frac{(n + 2\tau) L M^2}{\mu^2 \epsilon} \log\left(\frac{2 L D_0}{\epsilon} - 1\right)$$

we have a "morally $1/k$" rate:

$$\min_{0 \leq j \leq K} E[f(x_j) - f^*] \leq \epsilon.$$

Here, μ (convexity modulus of f), M (uniform bound on $\|\nabla f(x)\|$), L (Lipschitz for ∇f) are defined as earlier.

Asynchronous USCD: Linear Convergence (Liu, Wright, 2013)

Choose some $\rho > 1$ and pick γ small enough to ensure that

$$\rho^{-1} \leq \frac{\mathbb{E}(\|\nabla f(x_{j+1})\|^2)}{\mathbb{E}(\|\nabla f(x_j)\|^2)} \leq \rho.$$

Not too much change in gradient over each iteration, so not too much price to pay for using old information, in the asynchronous setting.

Can choose γ small enough to satisfy this property but large enough to get a linear rate.

Gory Details: Essentially Strongly Convex f

Essentially strongly convexity parameter μ:

$$f(x) - f(y) \geq \langle \nabla f(y), x - y \rangle + \frac{\mu}{2}\|x - y\|^2$$

for all $x, y \in \Omega$ with $P_{\mathcal{S}}(x) = P_{\mathcal{S}}(y)$. Weaker than usual strong convexity.[1]

For any $\rho > 1$, define

$$\psi = 1 + \frac{2\tau\rho^{\tau} L_{\text{res}}}{\sqrt{n} L_{\max}}$$

and

$$\gamma \leq \min\left\{ \frac{1}{\psi}, \frac{(\rho-1)\sqrt{n}L_{\max}}{2\rho^{\tau+1}L_{\text{res}}}, \frac{(\rho-1)\sqrt{n}L_{\max}}{L_{\text{res}}\rho^{\tau}(2 + \frac{L_{\text{res}}}{\sqrt{n}L_{\max}})} \right\}.$$

Linear rate:

$$\mathbb{E}(f(x_j) - f^*) \leq \left(1 - \frac{\mu\gamma}{nL_{\max}}(2 - \psi\gamma)\right)^j (f(x_0) - f^*).$$

[1] Example: $f(Ax)$ is essentially strongly convex if $f(.)$ is strongly convex.

A Particular Choice

Consider the regime in which

$$\tau + 1 \leq \frac{\sqrt{n} L_{\max}}{2 e L_{\text{res}}},$$

and define $\rho = 1 + \frac{2eL_{\text{res}}}{\sqrt{n}L_{\max}}$. Thus $\rho^{\tau+1} \leq e$ and $\gamma = 1/\psi = O(1)$. Get

$$\mathbb{E}(f(x_j) - f^*) \leq \left(1 - \frac{\mu}{2nL_{\max}}\right)^j (f(x_0) - f^*).$$

Converges to precision ϵ with probability at least $1 - \eta$ for

$$K \geq \frac{2nL_{\max}}{\mu} \left| \log \frac{f(x_0) - f^*}{\eta \epsilon} \right|.$$

The regime on τ is somewhat restrictive, but the degradation as τ exceeds this bound is not too severe. (Choose smaller γ.)

If $f(x) = \|Ax - b\|^2$ where $A \in \mathbb{R}^{m \times n}$ is a Gaussian random matrix, then $L_{\text{res}}/L_{\max}$ is bounded by $1 + O(\sqrt{n/m})$.

Discussion

Recall that short-step steepest descent on a strongly convex f gives linear convergence with rate approx:

$$1 - \frac{2}{(L/\mu)+1} \approx 1 - \frac{2\mu}{L}.$$

By comparison, n steps of asynchronous short-step steepest descent gives decrease factor approx:

$$1 - \frac{\mu}{2L_{\max}}.$$

When $L_{\max} \sim L$, suggests that about 4 times as many iterations would be needed by SCD. But we can run it in parallel!

Bound on τ is a measure of potential parallelization. When ratio $L_{\text{res}}/L_{\max}$ is favorable, get $\tau = O(\sqrt{n})$. Thus, expect linear rate on up to $O(\sqrt{n})$ cores running asynchronously in parallel.

Gory Details: Weakly Convex f ($\mu = 0$): Sublinear ($1/k$)

Defining ψ and γ as above, have

$$\mathbb{E}(f(x_j) - f^*) \leq \frac{1}{(f(x_0) - f^*)^{-1} + \frac{\gamma(2-\gamma\psi)}{2nL_{\max}R^2}j}.$$

Assuming $\tau + 1 \leq \frac{\sqrt{n}L_{\max}}{2eL_{\text{res}}}$ and setting ρ and γ as above, have

$$\mathbb{E}(f(x_j) - f^*) \leq \frac{1}{(f(x_0) - f^*)^{-1} + \frac{j}{4nL_{\max}R^2}}.$$

Roughly "$1/k$" behavior. To achieve precision ϵ with probability at least $1 - \eta$, need

$$K \geq 4nL_{\max}R^2 \left(\frac{1}{\eta\epsilon} - \frac{1}{f(x_0) - f^*}\right).$$

Asynchronous Constrained SCD (CSCD)

$$\min f(x) \text{ subject to } x \in \Omega,$$

where Ω is separable $\Omega = \Omega_1 \times \Omega_2 \times \cdots \Omega_n$.

At each iteration j:

- Choose i_j with equal probability from $\{1, 2, \ldots, n\}$;
- Update the i_j component:

$$x_{j+1} = P_{\Omega_j}\left(x_j - \frac{\gamma}{L_{\max}} \nabla_{i_j} f(x_{k(j)})\right),$$

where $k(j)$ is some iterate prior to j but no more than τ cycles old: $j - k(j) \leq \tau$. Here γ is a constant steplength.

Each core runs this process concurrently and asynchronously.

Asynchronous CSCD: Linear Convergence (Liu, Wright, 2013)

For purposes of analysis, define

$$\bar{x}_{j+1} = \arg\min_{x \in \Omega} \langle \nabla f(x_{k(j)}), x - x_j \rangle + \frac{L_{\max}}{2\gamma} \|x - x_j\|^2,$$

which would be obtained by "stepping in all components at once."

Motivation similar to ARK analysis: Choose some $\rho > (1 - 2/\sqrt{n})^{-1} > 0$ and pick γ small enough to ensure that

$$\mathbb{E}(\|x_{j-1} - \bar{x}_j)\|^2) \leq \rho \mathbb{E}(\|x_j - \bar{x}_{j+1}\|^2).$$

The condition is analogous to the "not much change in gradient" condition from the unconstrained case.

Can choose γ small enough to satisfy this, but large enough to get a linear rate.

Gory Details for f essentially strongly convex: Linear rate

For any $\rho > (1 - 2/\sqrt{n})^{-1} > 0$, define

$$\psi = 1 + \frac{L_{\text{res}} \tau \rho^{\tau}}{\sqrt{n} L_{\max}} \left(2 + \frac{L_{\max}}{\sqrt{n} L_{\text{res}}} + \frac{2\tau}{n} \right).$$

and choose

$$\gamma \leq \min \left\{ \frac{1}{\psi}, \left(1 - \frac{1}{\rho} - \frac{2}{\sqrt{n}} \right) \frac{\sqrt{n} L_{\max}}{4 L_{\text{res}} \tau \rho^{\tau}} \right\}.$$

We have for any $j \geq 0$

$$\mathbb{E}(f(x_j) - f^*) \leq \left(1 - \frac{\mu}{n(\mu + \gamma^{-1} L_{\max})} \right)^j \left(\frac{L_{\max} R^2}{2\gamma} + (f(x_0) - f^*) \right),$$

A Particular Choice

Consider the regime in which

$$\tau(\tau+1) \leq \frac{\sqrt{n}L_{\max}}{4eL_{\text{res}}},$$

and define

$$\rho = 1 + \frac{4e\tau L_{\text{res}}}{\sqrt{n}L_{\max}}.$$

Thus $\rho^{\tau+1} \leq e$ and $\gamma = 1/2$, and the rate simplifies to:

$$\mathbb{E}(f(x_j) - f^*) \leq \left(1 - \frac{\mu}{n(\mu + 2L_{\max})}\right)^j (L_{\max}R^2 + f(x_0) - f^*).$$

yielding convergence to precision ϵ with probability at least $1-\eta$ for

$$K \geq \frac{n(\mu + 2L_{\max})}{\mu} \left|\log \frac{L_{\max}R^2 + f(x_0) - f^*}{\eta\epsilon}\right|.$$

The regime on τ is somewhat restrictive, but the degradation as τ exceeds this bound is not too severe. (Choose smaller γ.)

Discussion

Linear convergence rate is roughly the same for the separable constrained case as for the unconstrained case BUT the restrictions on τ are tighter. For favorable ratio $L_{\text{res}}/L_{\max}$, have $\tau = O(n^{1/4})$, rather than $O(n^{1/2})$.

Gory Details for weakly convex f: Sublinear $(1/k)$

Defining ψ and γ as above, have

$$\mathbb{E}(f(x_j) - f^*) \leq \frac{n(L_{\max} R^2 + f(x_0) - f^*)}{j + n}.$$

Assuming $\tau(\tau + 1) \leq \frac{\sqrt{n} L_{\max}}{4 e L_{\text{res}}}$ and setting ρ and γ as above, have

$$\mathbb{E}(f(x_j) - f^*) \leq \frac{n(L_{\max} R^2 + 2\gamma(f(x_0) - f^*))}{2\gamma(j + n)}.$$

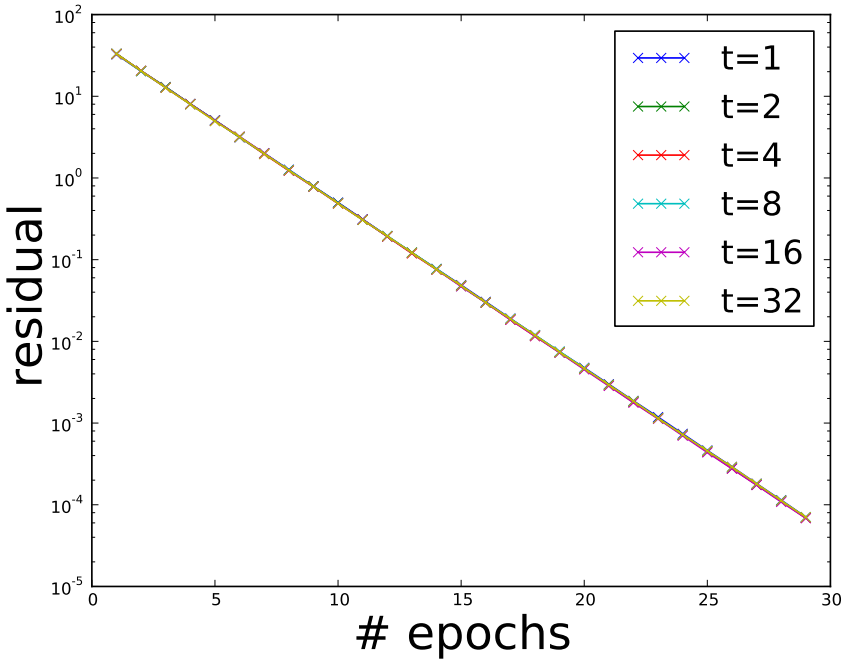
Roughly "$1/k$" behavior. To achieve precision ϵ with probability at least $1 - \eta$ in K iterations, K needs to satisfy

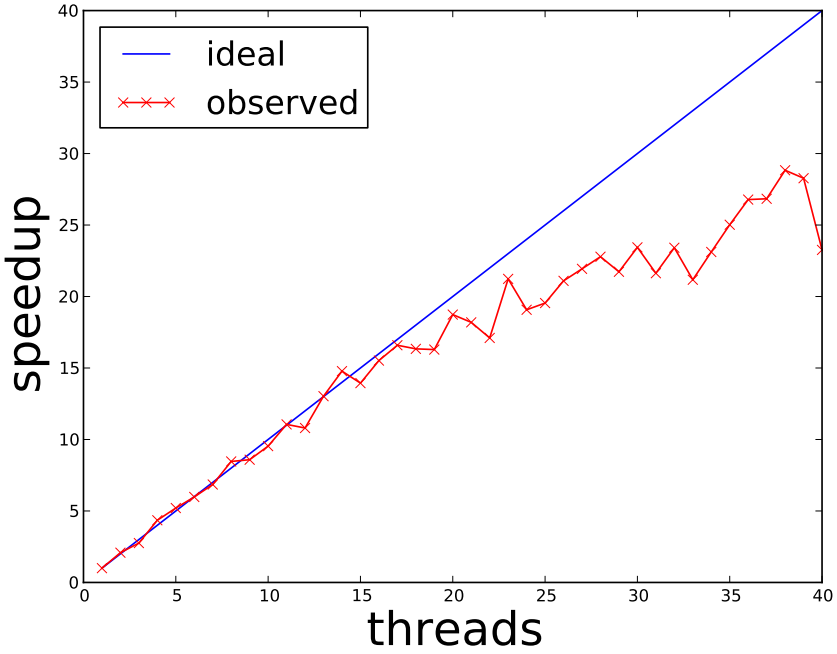
$$K \geq \frac{n(L_{\max} R^2 + f(x_0) - f^*)}{\eta \epsilon} - n.$$

Implemented on 4-socket, 40-core Intel Xeon

$$\min_x \quad \|Ax - b\|^2 + 0.5\|x\|^2$$

where $A \in \mathbb{R}^{m \times n}$ is a Gaussian random matrix ($m = 6000$, $n = 20000$, columns are normalized to 1). $L_{\text{res}}/L_{\max} \approx 2.2$. Choose $\gamma = 1$.

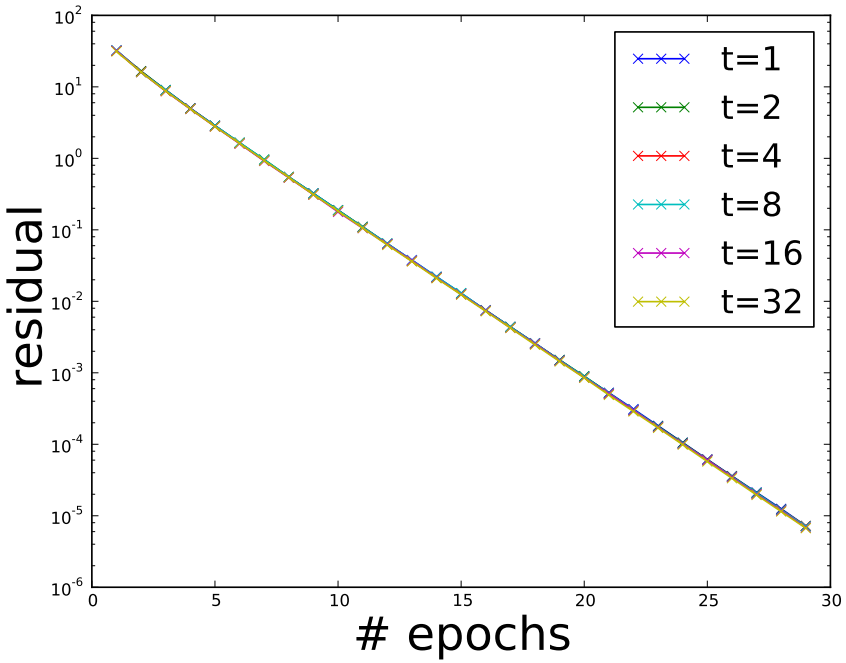


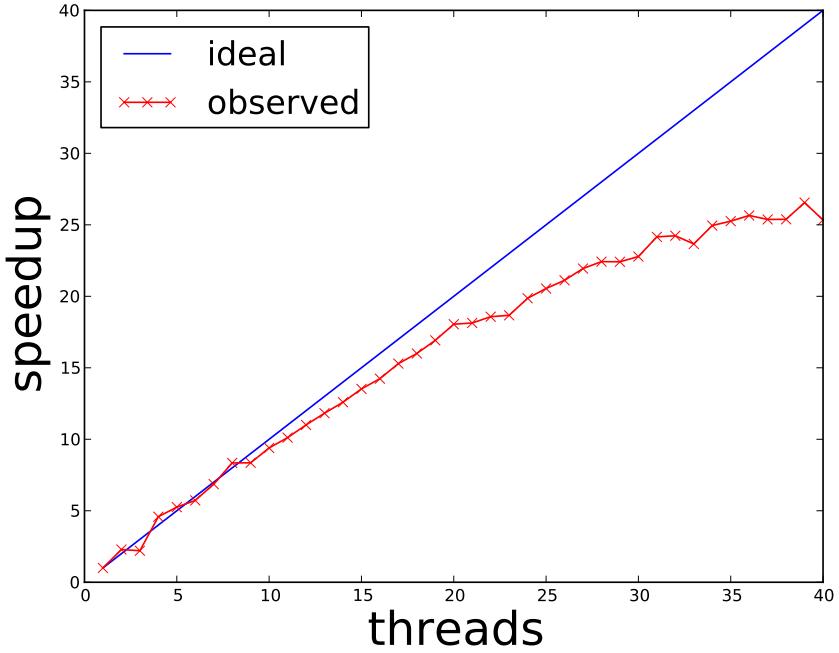


Constrained: Implemented on 4-socket, 40-core Intel Xeon

$$\min_{x \geq 0} \quad (x - z)^T (A^T A + 0.5I)(x - z) \quad ,$$

where $A \in \mathbb{R}^{m \times n}$ is a Gaussian random matrix ($m = 6000$, $n = 20000$, columns are normalized to 1) and z is a Gaussian random vector. $L_{\text{res}}/L_{\max} \approx 2.2$. Choose $\gamma = 1$.





SCD: Extensions

Approaches above are extendible for

- blocks of coordinates: partition indices $\{1, 2, \ldots, n\}$ into m disjoint blocks $[1], [2], , \ldots, [m]$.
- regularizers separable by coordinates or blocks: $f(x) + \lambda\psi(x)$, where $\psi(x) = \psi_1(x_{[1]}) + \psi_2(x_{[2]}) + \ldots + \psi_m(x_{[m]})$.

(Already done by Richtarik and Takac, in various settings.)

Avron, Druinsky, Gupta (2013) solve $Ax = b$, where A is symmetric positive definite. As well as the "HOGWILD!" model of asynchronous computation, they propose an "inconsistent read" model. Linear convergence.

Kaczmarz for $Ax = b$.

Consider linear equations $Ax = b$, where the equations are consistent and matrix A is $m \times n$, not necessarily square or full rank. Write

$$A = \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_m^T \end{bmatrix}, \quad \text{where } \|a_i\|_2 = 1,\ i = 1, 2, \ldots, m.$$

Iteration j of Kacmarz:

- Select row index $i_j \in \{1, 2, \ldots, m\}$ randomly with equal probability.
- Set

$$x_{j+1} \leftarrow x_j - (a_{i_j}^T x_j - b_{i_j}) a_{i_j}.$$

Equivalent to applying SG to

$$f(x) := \frac{1}{2m} \sum_{i=1}^{m} (a_i^T x - b_i)^2 = \frac{1}{2m} \|Ax - b\|_2^2,$$

with steplength $\alpha_k \equiv 1$.

Randomized Kaczmarz Convergence

Strohmer and Vershynin (2009) get a *linear* rate:

$$E\left[\|x_{j+1} - P(x_{j+1})\|_2^2 \,|\, x_j\right] \leq \left(1 - \frac{\lambda_{\min,\mathrm{nz}}}{m}\right) \|x_j - P(x_j)\|_2^2,$$

where P denotes projection onto solution subspace, and $\lambda_{\max}$ and $\lambda_{\min,\mathrm{nz}}$ denote largest and smallest nonzero eigenvvalues of $A^T A$.

We can obtain essentially the same linear rate by analyzing SG applied to the sum-of-squares form, following standard analysis but using special structure, namely, all f_i have zero gradients:

$$\nabla f_i(x^*) = a_i(a_i^T x^* - b_i) = 0, \quad i = 1, 2, \ldots, m.$$

Elementary SG analysis for $\min \|Ax - b\|^2$.

Assume A scaled so that $\|a_i\| = 1$ for all i. $\lambda_{\min,\mathrm{nz}}$ denotes minimum nonzero eigenvalue of $A^T A$. $P(\cdot)$ is projection onto solution set.

Choose steplength $\alpha_k \equiv 1$, we have

$$\begin{aligned}\frac{1}{2}\|x_{j+1} - P(x_{j+1})\|^2 &\leq \frac{1}{2}\|x_j - a_{i_j}(a_{i_j}^T x_j - b_{i_j}) - P(x_j)\|^2 \\ &= \frac{1}{2}\|x_j - P(x_j)\|^2 - \frac{1}{2}(a_{i_j}^T x_j - b_{i_j})^2.\end{aligned}$$

Taking expectations:

$$\begin{aligned}E\left[\frac{1}{2}\|x_{j+1} - P(x_{j+1})\|^2 \,|\, x_j\right] &\leq \frac{1}{2}\|x_j - P(x_j)\|^2 - \frac{1}{2}E\left[(a_{i_j}^T x_j - b_{i_j})^2\right] \\ &= \frac{1}{2}\|x_j - P(x_j)\|^2 - \frac{1}{2m}\|Ax_j - b\|^2 \\ &\leq \left(1 - \frac{\lambda_{\min,\mathrm{nz}}}{m}\right)\frac{1}{2}\|x_j - P(x_j)\|^2.\end{aligned}$$

Asynchronous Kaczmarz (Liu, Wright, 2013)

At each iteration j:

- Select i_j from $\{1, 2, \ldots, m\}$ with equal probability;
- Select t_j from the support of a_{i_j} with equal probability;
- Update:

$$x_{j+1} = x_j - \gamma \|a_{i_j}\|_0 (a_{i_j}^T x_{k(j)} - b_{i_j}) E_{t_j} a_{i_j},$$

where

- $k(j)$ is some iterate prior to j but no more than τ cycles old: $j - k(j) \leq \tau$;
- γ is a constant steplength;
- E_t is the $n \times n$ matrix of all zeros, except for 1 in the (t, t) location.

As in HOGWILD!, different cores run this process concurrently, updating an x accessible to all.

ARK Analysis: Linear Convergence

Choice of γ follows a familiar strategy, but details differ.

Choose some $\rho > 1$ (typically $\rho = 1 + O(1/m)$) and choose γ small enough to ensure that

$$\rho^{-1}\mathbb{E}(\|Ax_j - b\|^2) \leq \mathbb{E}(\|Ax_{j+1} - b\|^2) \leq \rho\mathbb{E}(\|Ax_j - b\|^2).$$

Not too much change in the expected residual on any one iteration — for the usual reasons.

Depends on parameters

- $\chi := \max_{i=1,2,\ldots,m} \|a_i\|_0$ (maximum nonzero row count);
- $\alpha := \max_{i,t} \|a_i\|_0 \|AE_t a_i\| \leq \chi\|A\|$.

Gory Details

Choose any $\rho > 1$ and define γ via the following:

$$\psi = \chi + \frac{2\lambda_{\max}\tau\rho^{\tau}}{m}$$

$$\gamma \leq \min\left\{\frac{1}{\psi},\ \frac{m(\rho-1)}{2\lambda_{\max}\rho^{\tau+1}},\ m\sqrt{\frac{\rho-1}{\rho^{\tau}(m\alpha^2+\lambda_{\max}^2\tau\rho^{\tau})}}\right\}$$

Then have

$$\rho^{-1}\mathbb{E}(\|Ax_j - b\|^2) \leq \mathbb{E}(\|Ax_{j+1} - b\|^2) \leq \rho\mathbb{E}(\|Ax_j - b\|^2)$$

$$\mathbb{E}(\|x_{j+1} - P(x_{j+1})\|^2) \leq \left(1 - \frac{\lambda_{\min,\mathrm{nz}}\gamma}{m\chi}(2-\gamma\psi)\right)\mathbb{E}(\|x_j - P(x_j)\|^2),$$

A particular choice of ρ leads to simplified results, in a reasonable regime.

A Particular Choice

Assume

$$2e\lambda_{\max}(\tau+1) \leq m$$

and set $\rho = 1 + 2e\lambda_{\max}/m$. Can show that $\gamma = 1/\psi$ for this case, so expected convergence is

$$\mathbb{E}(\|x_{j+1} - P(x_{j+1})\|^2) \leq \left(1 - \frac{\lambda_{\min,\mathrm{nz}}}{m(\chi+1)}\right) \mathbb{E}(\|x_j - P(x_j)\|^2).$$

Converges to precision ϵ with probability at least $1-\eta$ in

$$K \geq \frac{m(\chi+1)}{\lambda_{\min,\mathrm{nz}}} \left|\log \frac{\|x_0 - P(x_0)\|^2}{\eta\epsilon}\right| \quad \text{iterations.}$$

In the regime $2e\lambda_{\max}(\tau+1) \leq m$ considered here the delay τ doesn't really interfere with convergence rate.

Discussion

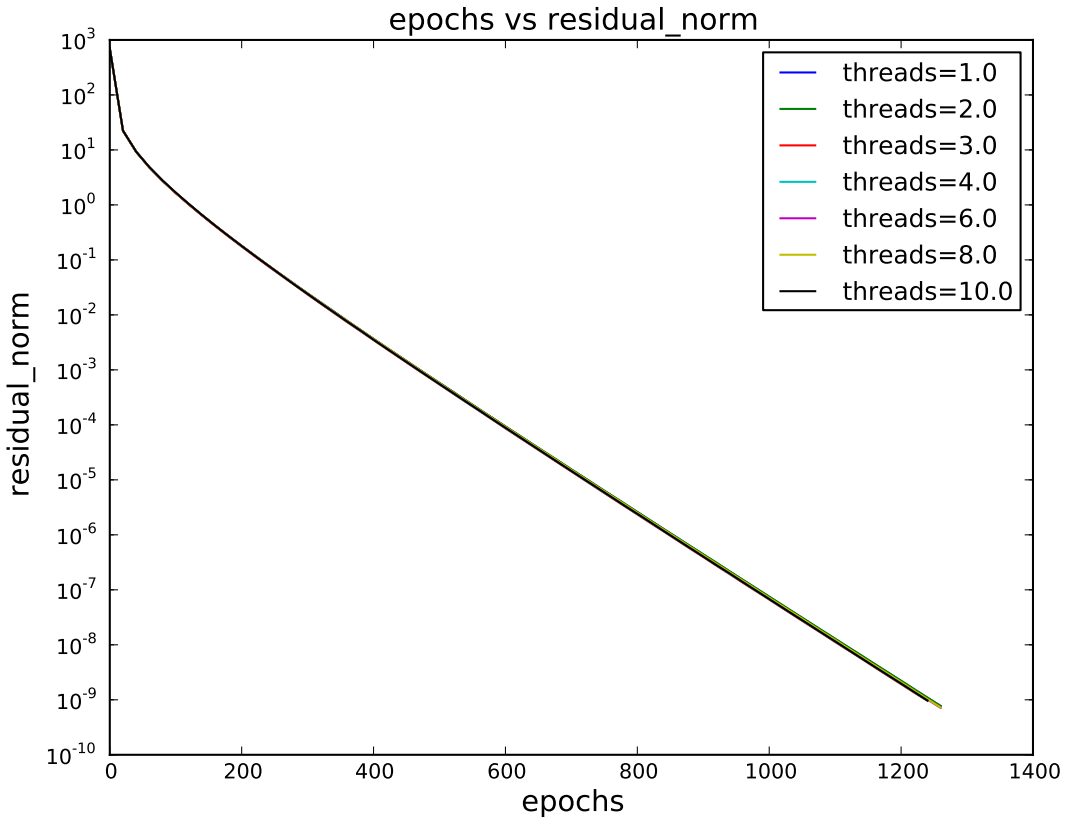
For random matrices A with unit rows, we have $\lambda_{\max} \approx (1 + O(m/n))$, with high probability.

Conditions on τ are less strict than for the SCD algorithms. For random matrices A, with m and n of the same order, have $\tau = O(m) = O(n)$.

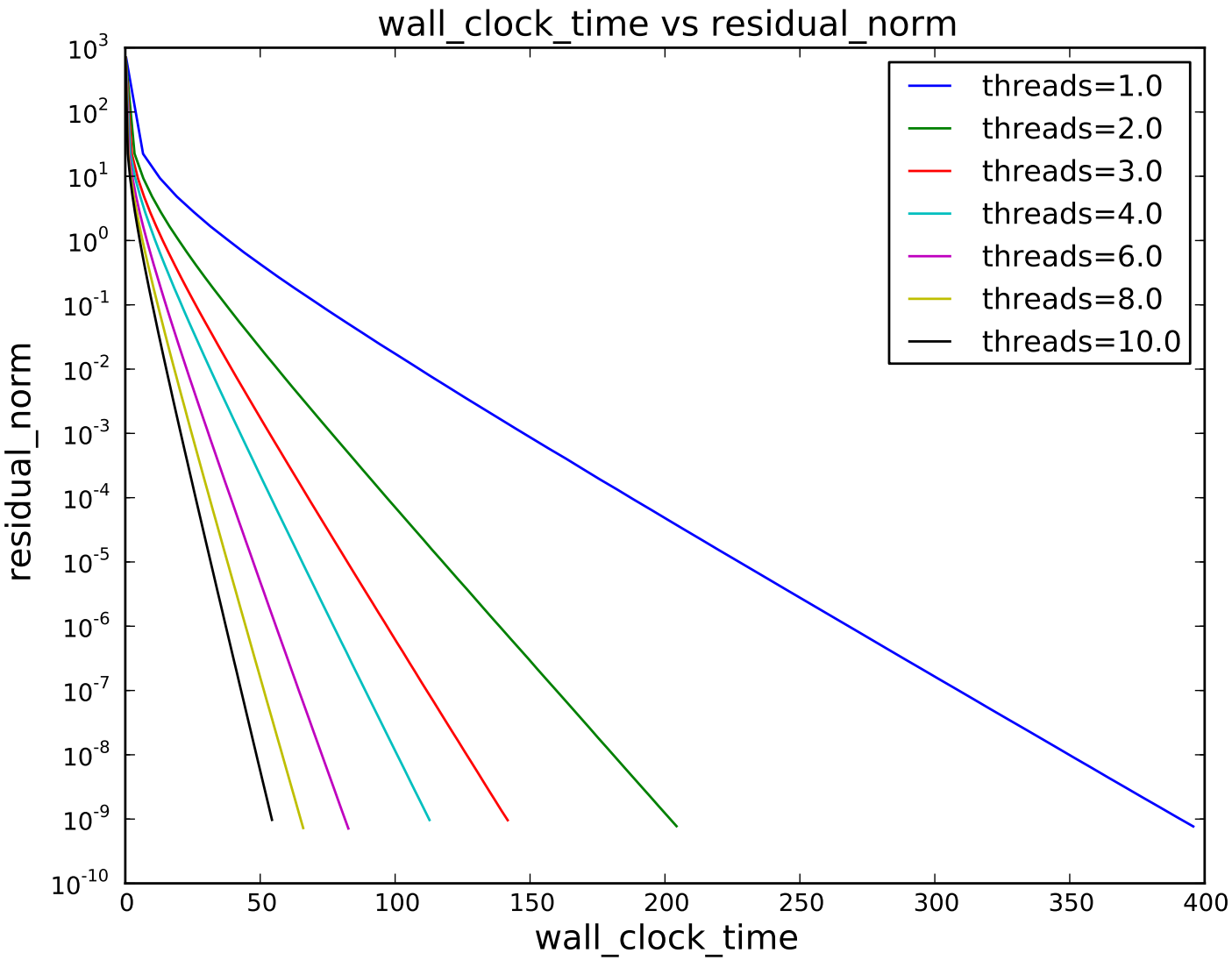
(Recall $\tau = O(n^{1/4})$ for CSCD and $\tau = O(n^{1/2})$ for USCD.)

Asynchronous Kaczmarz Results

Applied asynchronous parallel Kaczmarz to $Ax = b$ where A is $100,000 \times 80,000$ and 3% dense. (1.8GB total)



Asynchronous Kaczmarz: Timings



Extreme Linear Programming

State-of-the-art solvers for large linear programs (LPs) are based on simplex and interior-point methods. An alternative approach based on

- augmented Lagrangian / proximal-point
- iterative solvers for the bounded-QP subproblems (SOR, CG)

were studied in the late 1980s

- O. L. Mangasarian and R. DeLeone, "Serial and Parallel Solution of Large-Scale Linear Program by Augmented Lagrangian Successive Overrelaxations," 1987.
- S. J. Wright, "Implementing Proximal-Point Methods for Linear Programming," JOTA, 1990

These showed some promise on random, highly degenerate problems, but were terrible on the netlib test set and other problems arising in practice.

But this approach has potential appeal for:

- Cases in which only crude approximate LP solutions are needed.
- No matrix factorizations or multiplications are required. (Thus may be good for special problems, at extreme scale.)
- Multicore implementation is easy, when asynchronous solver is used on the QP subproblems.

Basics of the Approach

Primal-dual pair:

$$\min_x \ c^T x \ \text{ s.t. } \ Ax = b, \ \ x \geq 0$$

$$\max_u \ b^T u \ \text{ s.t. } \ A^T u \leq c.$$

"Proximal method of multipliers" subproblem is a bound-constrained convex QP:

$$x(\beta) := \arg\min_{x \geq 0} \ c^T x - \bar{u}^T (Ax - b) + \frac{\beta}{2} \|Ax - b\|^2 + \frac{1}{2\beta} \|x - \bar{x}\|_2^2,$$

where $(\bar{x}, \bar{u})$ is an estimate of the primal-dual solution and β is a penalty parameter.

Can solve a sequence of these, with updates to $\bar{u}$ and $\bar{x}$, and possible increases in β, in the familiar style of augmented Lagrangian.

Solve the QP using SCD on 32 cores.

A Perturbation Result

Make use of Renegar's measures δ_P and δ_D of relative distance to primal and dual infeasibility.

Theorem

Suppose that δ_P and δ_D are both positive, and let (x^*, u^*) be any primal-dual solution pair. If we define $C_* := \max(\|x^* - \bar{x}\|, \|u^* - \bar{u}\|)$, then the unique solution $x(\beta)$ of the QP subproblem satisfies

$$\|Ax(\beta) - b\| \leq (1/\beta)(1+\sqrt{2})C_*, \quad \|x(\beta) - x^*\| \leq \sqrt{6}C_*.$$

If in addition we have

$$\beta \geq \frac{10C_*}{\|d\| \min(\delta_P, \delta_D)},$$

then

$$|c^T x^* - c^T x(\beta)| \leq \frac{1}{\beta}\left[\frac{25C_*}{2\delta_P\delta_D} + 6C_*^2 + \sqrt{6}\|\bar{x}\|C_*\right].$$

LP Rounding Approximations

There are numerous NP-hard problems for which approximate solutions can be found using linear programming followed by rounding. Typical process:

- Construct a MIP formulation;
- Relax to an LP (replace binary variables by $[0, 1]$ intervals);
- Solve the LP approximately;
- Use LP solution to construct a feasible MIP solution ("rounding").

Examples: Vertex cover, set cover, set packing, multiway cut, maximal independent set.

Vertex Cover

Given a graph with edge set E, vertex set V, seek a subset of vertices such that every edge touches the subset. Cost to select a vertex v is c_v.

Binary programming formulation:

$$\min \sum_{v \in V} c_v x_v \quad \text{s.t.} \quad x_u + x_v \geq 1 \ \text{ for } (u,v) \in E; \quad x_v \in \{0,1\} \text{ for all } v \in V.$$

Relax the binary constraint to $x_v \in [0,1]$ to get an LP. Very large, but matrix A is highly sparse and structured.

Sample Results

instance	vertex cover		multiway cuts	
	n	size(A)	n	size(A)
frb59-26-1	126K	616K	1.3M	3.6M
Amazon	203K	956K	6.8M	21.3 M
DBLP	146K	770K	10.7M	33.7M
Google+	82K	1.5M	7.6M	24.1M

Table: Problem Sizes

Computation Times (Seconds)

Run on 32 cores Intel machine for max of one hour. Compared with Cplex IP and LP solvers. Times shown for reaching solutions of similar quality.

instance	Cplex IP	Cplex LP	Us
frb59-26-1 VC	-	5.1	0.65
Amazon VC	44	22	4.7
DBLP VC	23	21	3.2
Google+ VC	-	62	6.2
frb59-26-1 MC	54	360	29
Amazon MC	-	-	131
DBLP MC	-	-	158
Google+ MC	-	-	570

(Cplex IP sometimes faster than LP because the IP preprocessing can drastically simplify the problem, for some data sets.)

Conclusions

- Old methods are interesting again, because of modern computers and modern applications (particularly in machine learning).
- We can analyze asynchronous parallel algorithms, with a computing model that approximates reality pretty well (but there are others too).
- Disruptive?

FIN