Batched Stochastic Gradient Descent with Weighted Sampling

Deanna Needell

Partially supported by NSF CAREER #1348721 and the Alfred P. Sloan Foundation

Includes joint works with

Rachel Ward (UT Austin)

Jesus De Loera (UC Davis)

Jamie Haddock (UC Davis)

Objective

- Minimize:

$$F(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}) = \mathbb{E} f_i(\boldsymbol{x})$$

- Examples:
  - Linear Feasiblity (Ax ≤ b)
  - Least Squares

$$\boldsymbol{x}_{LS} \stackrel{\text{def}}{=} \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^m} \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2 = \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^m} \frac{1}{n}\sum_{i=1}^{n}\frac{n}{2}(\boldsymbol{b}_i - \langle \boldsymbol{a}_i, \boldsymbol{x}\rangle)^2 = \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^m} \mathbb{E} f_i(\boldsymbol{x})$$

  - Hinge Loss

$$\boldsymbol{x}_{HL} \stackrel{\text{def}}{=} \operatorname*{argmin}_{\boldsymbol{w}\in\mathbb{R}^m} \frac{1}{n}\sum_{i=1}^{n}[1 - y_i\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle]_+ + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2$$

Assumptions

- Strong Convexity:

$$\langle \boldsymbol{x}-\boldsymbol{y}, \nabla F(\boldsymbol{x})-\nabla F(\boldsymbol{y})\rangle \geq \mu\|\boldsymbol{x}-\boldsymbol{y}\|_2^2$$

- Residual:

$$\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\boldsymbol{x}_*)\|_2^2 \leq \sigma^2$$

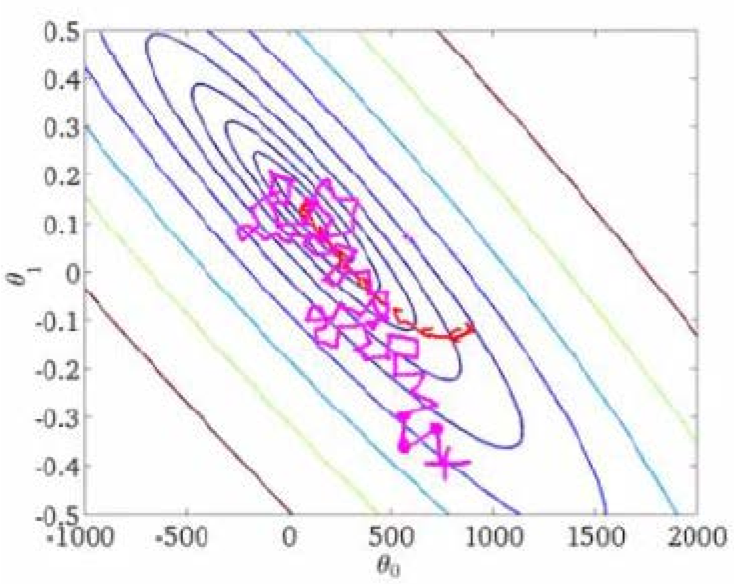
- Smoothness:

$$\|\nabla f_i(\boldsymbol{x})-\nabla f_i(\boldsymbol{y})\|_2 \leq L_i\|\boldsymbol{x}-\boldsymbol{y}\|_2$$

  - -or- functionals themselves have bounded Lipschitz (later)

Stochastic Gradient Descent

$$\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k - \gamma \nabla f_{i_k}(\boldsymbol{x}_k)$$



Convergence Guarantees

- Can guarantee $\mathbb{E}\|\mathbf{x}_k - \mathbf{x}_*\|_2^2 \leq \varepsilon$ after:
  - [Bach & Moulines `11]:

$$k = 2\log(\varepsilon_0/\varepsilon)\left(\left(\frac{\sqrt{\frac{1}{n}\sum_i L_i^2}}{\mu}\right)^2 + \frac{\sigma^2}{\mu^2\varepsilon}\right)$$

  - [N & Srebro & Ward `16]:

$$k = 2\log(\varepsilon/\varepsilon_0)\left(\frac{\sup_i L_i}{\mu} + \frac{\sigma^2}{\mu^2\varepsilon}\right)$$

Tightness

- Can guarantee $\mathbb{E}\|\mathbf{x}_k - \mathbf{x}_*\|_2^2 \leq \varepsilon$ after:
  - [N & Srebro & Ward `16]: $k = 2\log(\varepsilon/\varepsilon_0)\left(\frac{\sup_i L_i}{\mu} + \frac{\sigma^2}{\mu^2\varepsilon}\right)$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1/\sqrt{n} \\ 0 & 1/\sqrt{n} \\ \vdots & \vdots \\ 0 & 1/\sqrt{n} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$\frac{\sup_i L_i}{\mu} = n \sup_i \|\mathbf{a}_i\|^2 \|\mathbf{A}^\dagger\|^2 = n$$

Convergence Guarantees

- Can guarantee $\mathbb{E}\|\mathbf{x}_k - \mathbf{x}_*\|_2^2 \leq \varepsilon$ after:
  - [Bach & Moulines `11]: $k = 2\log(\varepsilon_0/\varepsilon)\left(\left(\frac{\sqrt{\frac{1}{n}\sum_i L_i^2}}{\mu}\right)^2 + \frac{\sigma^2}{\mu^2\varepsilon}\right)$
  - [N & Srebro & Ward `16]: $k = 2\log(\varepsilon/\varepsilon_0)\left(\frac{\sup_i L_i}{\mu} + \frac{\sigma^2}{\mu^2\varepsilon}\right)$
    - With *weighted sampling* (proportional to L_i):

$$k = 2\log(\varepsilon/\varepsilon_0)\left(\frac{\frac{1}{n}\sum_i L_i}{\mu} + \frac{(\sum_i L_i)^2}{n^2 L_{\min}}\frac{\sigma^2}{\mu^2\varepsilon}\right)$$

    - With partially *weighted sampling* (proportional to ½ + ½ L_i):

$$k = 4\log(\varepsilon_0/\varepsilon)\left(\frac{\frac{1}{n}\sum_i L_i}{\mu} + \frac{\sigma^2}{\mu^2\varepsilon}\right)$$

Convergence – Other scenarios

- Can guarantee $\mathbb{E}\|\mathbf{x}_k - \mathbf{x}_*\|_2^2 \le \varepsilon$ using partially weighted sampling after:
  - In the smooth, non-strongly convex case:

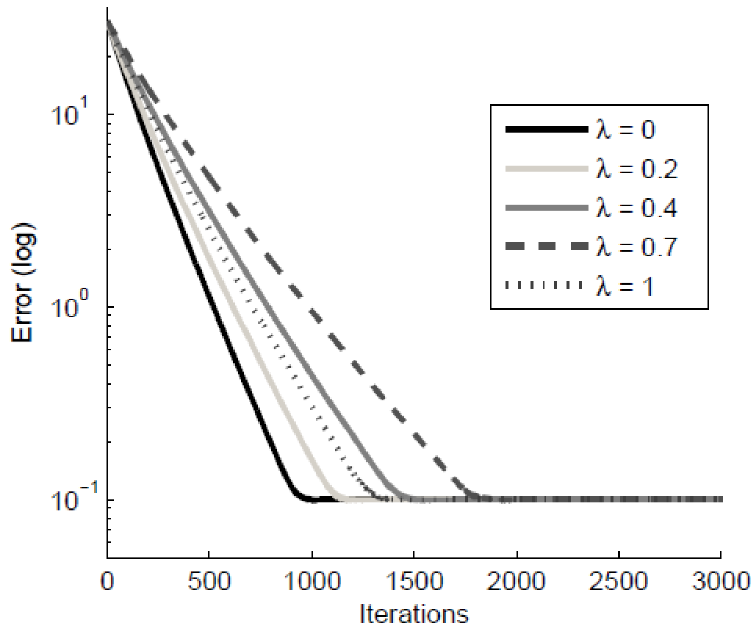
$$k = O\left(\frac{\overline{L}\|\boldsymbol{x}_\star\|_2^2}{\varepsilon} \cdot \frac{F(\boldsymbol{x}_\star) + \varepsilon}{\varepsilon}\right)$$

  - In the strongly convex, non-smooth case:
    - Using subgradients, and assuming functionals have Lipschitz G_i
      We have $\mathbb{E}[F(\boldsymbol{x}_k) - F(\boldsymbol{x}_\star)] \le \varepsilon$ after:

$$k = O\left(\frac{(\sum_i G_i)^2}{\mu\varepsilon}\right)$$

Experiments – Least Squares

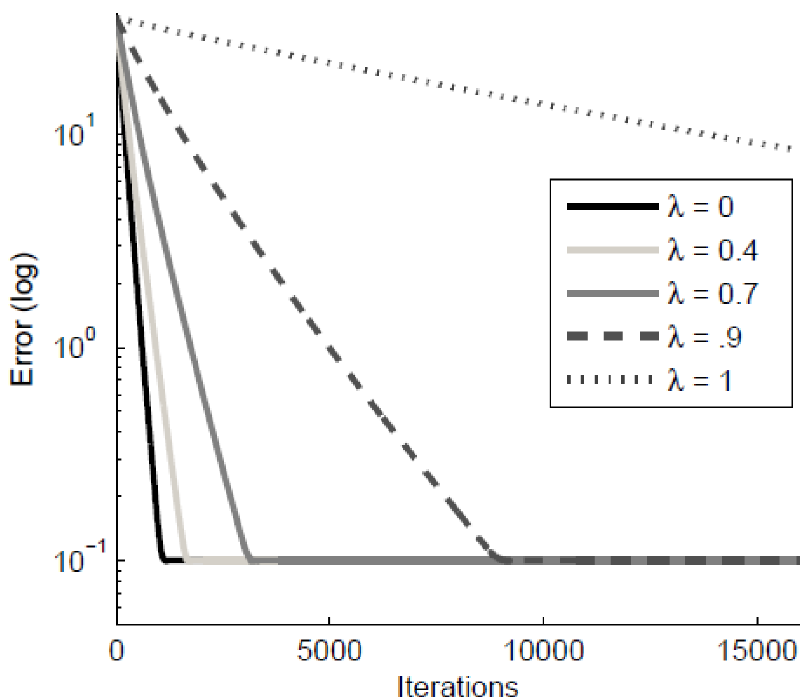
- Consider sampling with weights λ proportion of the time



Gaussian Matrix ~ N(0,1)

Experiments – Least Squares

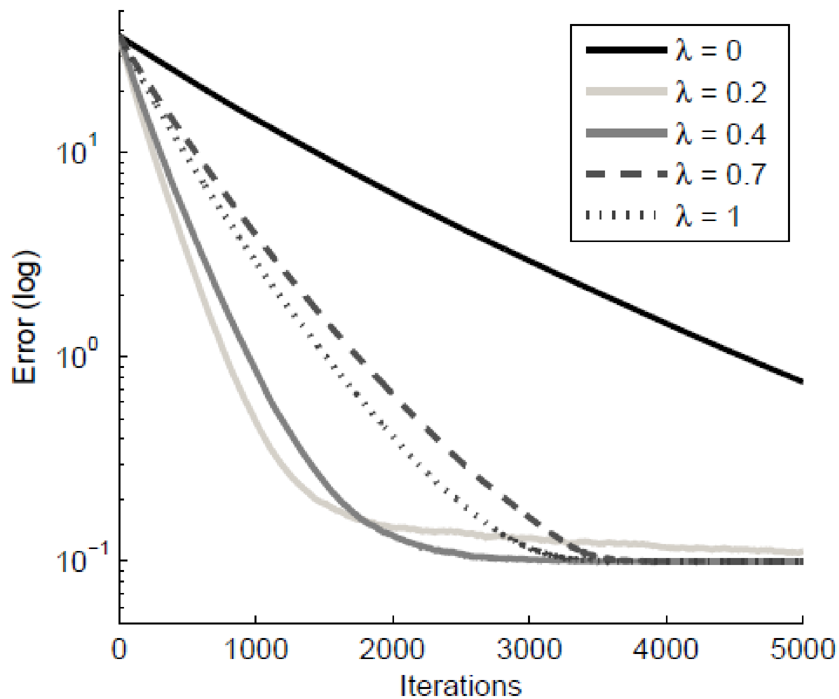
- Consider sampling with weights λ proportion of the time



Gaussian Matrix ~ N(0,1), last row N(0,100)

Experiments – Least Squares

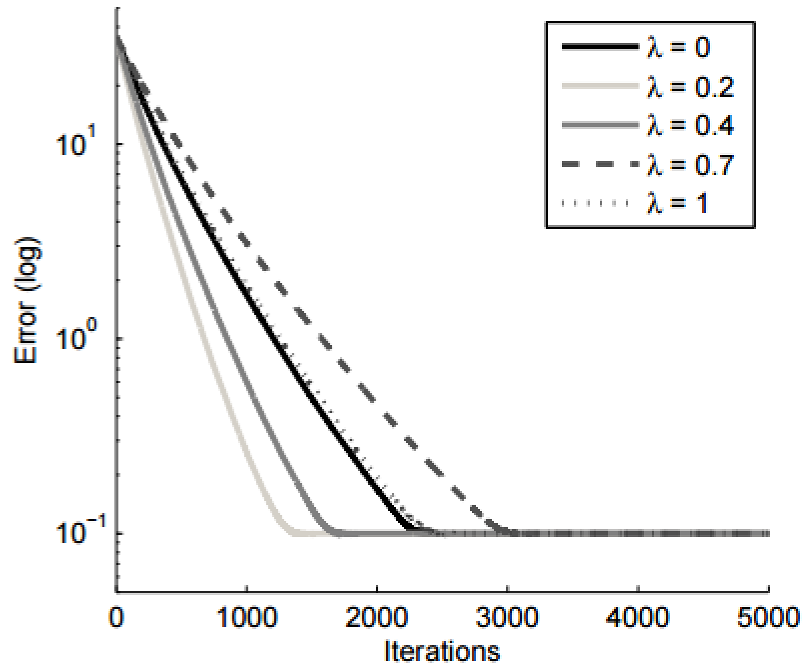
- Consider sampling with weights λ proportion of the time



Gaussian Matrix, $A_{ij} \sim N(0,j)$, large residual

Experiments – Least Squares

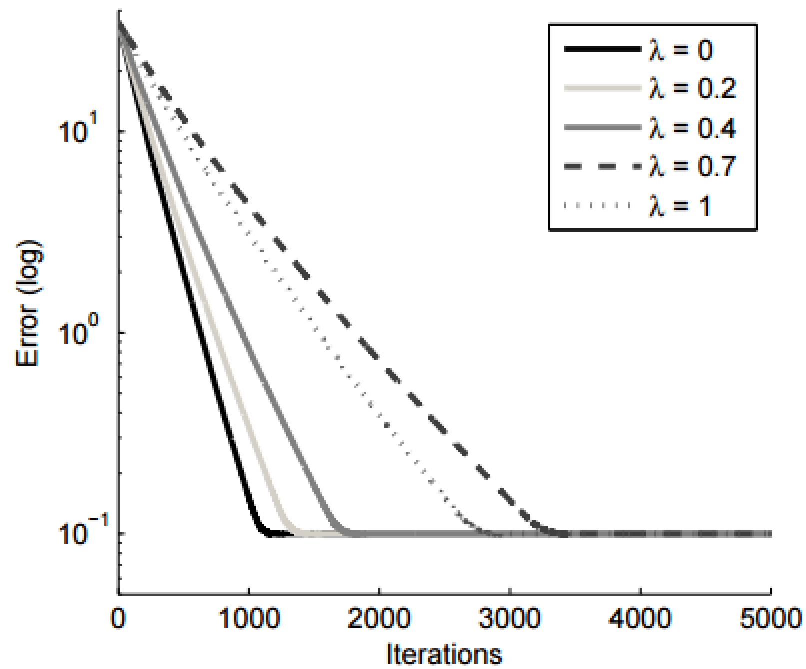
- Consider sampling with weights λ proportion of the time



Gaussian Matrix, A_{ij} ~ N(0,j), medium residual

Experiments – Least Squares

- Consider sampling with weights λ proportion of the time



Gaussian Matrix, $A_{ij} \sim N(0,j)$, small residual

SGD with batching and weighting

- Batch functionals into d batches of size b (b cores)

$$F(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}) = \mathbb{E} f_i(\boldsymbol{x}) \rightarrow F(\boldsymbol{x}) = \frac{1}{d}\sum_{i=1}^{d} g_{\tau_i}(\boldsymbol{x}) = \mathbb{E} g_{\tau_i}(\boldsymbol{x})$$

SGD with batching and weighting

- Batch functionals into d batches of size b (b cores)

$$F(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}) = \mathbb{E} f_i(\boldsymbol{x}) \rightarrow F(\boldsymbol{x}) = \frac{1}{d}\sum_{i=1}^{d} g_{\tau_i}(\boldsymbol{x}) = \mathbb{E} g_{\tau_i}(\boldsymbol{x})$$

- The strong convexity parameter μ for the function F remains invariant to the batching rule.

SGD with batching and weighting

- Batch functionals into d batches of size b (b cores)

$$F(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}) = \mathbb{E} f_i(\boldsymbol{x}) \rightarrow F(\boldsymbol{x}) = \frac{1}{d}\sum_{i=1}^{d} g_{\tau_i}(\boldsymbol{x}) = \mathbb{E} g_{\tau_i}(\boldsymbol{x})$$

- The strong convexity parameter μ for the function F remains invariant to the batching rule.

- The residual error σ_τ^2 such that $\frac{1}{d}\sum_{i=1}^{d} \|\nabla g_{\tau_i}(x_*)\|_2^2 \le \sigma_\tau^2$ can only **decrease** with increasing batch size, since

$$\sigma_\tau^2 = \frac{1}{d}\sum_{k=1}^{d} \| \frac{1}{b}\nabla\left(\sum_{k\in\tau_i} f_k(\boldsymbol{x})\right)\|_2^2 \le \frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\boldsymbol{x})\|_2^2 \le \sigma^2.$$

SGD with batching and weighting

- Batch functionals into d batches of size b (b cores)

$$F(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}) = \mathbb{E} f_i(\boldsymbol{x}) \rightarrow F(\boldsymbol{x}) = \frac{1}{d}\sum_{i=1}^{d} g_{\tau_i}(\boldsymbol{x}) = \mathbb{E} g_{\tau_i}(\boldsymbol{x})$$

- The strong convexity parameter μ for the function F remains invariant to the batching rule.

- The residual error σ_τ^2 such that $\frac{1}{d}\sum_{i=1}^{d} \|\nabla g_{\tau_i}(x_*)\|_2^2 \le \sigma_\tau^2$ can only **decrease** with increasing batch size, since

$$\sigma_\tau^2 = \frac{1}{d}\sum_{k=1}^{d} \|\frac{1}{b}\nabla\left(\sum_{k\in\tau_i} f_k(\boldsymbol{x})\right)\|_2^2 \le \frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\boldsymbol{x})\|_2^2 \le \sigma^2.$$

- The average Lipschitz constant $\overline{L}_\tau = \frac{1}{d}\sum_{i=1}^{d} L_{\tau_i}$ of the gradients of the batched functions g_{τ_i} can only **decrease** with increasing batch size, since by the triangle inequality, $L_{\tau_i} \le \frac{1}{b}\sum_{k\in\tau_i} L_k$, and thus

$$\frac{1}{d}\sum_{i=1}^{d} L_{\tau_i} \le \frac{1}{n}\sum_{k=1}^{n} L_k = \overline{L}.$$

SGD with batching and weighting

Theorem *Assume that the convexity and smoothness conditions on* $F(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x})$ *are in force. Consider the* $d = n/b$ *batches* $g_{\tau_i}(\boldsymbol{x}) = \frac{1}{b}\sum_{k\in\tau_i} f_k(\boldsymbol{x})$*, and the batched weighted SGD iteration*

$$\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k - \frac{\gamma}{d\cdot p(\tau_{i_k})}\nabla g_{\tau_{i_k}}(\boldsymbol{x}_k)$$

where batch τ_i *is selected at iteration* k *with probability*

$$p(\tau_i) = \frac{1}{2d} + \frac{1}{2d}\cdot\frac{L_{\tau_i}}{\overline{L}_\tau}. \tag{3.1}$$

For any desired ε*, and using a stepsize of*

$$\gamma = \frac{\mu\varepsilon}{4(\varepsilon\mu\overline{L}_\tau + \sigma_\tau^2)},$$

we have that after a number of iterations

$$k = 4\log(2\varepsilon_0/\varepsilon)\left(\frac{\overline{L}_\tau}{\mu} + \frac{\sigma_\tau^2}{\mu^2\varepsilon}\right),$$

the following holds in expectation with respect to the weighted distribution (3.1): $\mathbb{E}^{(p)}\|\mathbf{x}_k - \mathbf{x}_*\|_2^2 \le \varepsilon$.

Least Squares Case

- Non-batched: $f_i(\boldsymbol{x}) = \frac{n}{2}(b_i - \langle \boldsymbol{a}_i, \boldsymbol{x}\rangle)^2$

(1) The individual Lipschitz constants are bounded by $L_i = n\|\boldsymbol{a}_i\|_2^2$, and the average Lipschitz constant by $\frac{1}{n}\sum_i L_i = \|\boldsymbol{A}\|_F^2$ (where $\|\cdot\|_F$ denotes the Frobenius norm),

(2) The strong convexity parameter is $\mu = \frac{1}{\|\boldsymbol{A}^{-1}\|^2}$ (where $\|\boldsymbol{A}^{-1}\| = \sigma_{\min}^{-1}(\boldsymbol{A})$ is the reciprocal of the smallest singular value of $\boldsymbol{A}$),

(3) The residual is $\sigma^2 = n\sum_i \|\boldsymbol{a}_i\|_2^2 |\langle \boldsymbol{a}_i, \boldsymbol{x}_*\rangle - \boldsymbol{a}_i|^2$.

- Batched: $g_{\tau_i}(\boldsymbol{x}) = \dfrac{d}{2}\|\boldsymbol{A}_{\tau_i}\boldsymbol{x} - \boldsymbol{b}_{\tau_i}\|_2^2$

  - $$L_{\tau_i} = \sup_{\boldsymbol{x},\boldsymbol{y}} \frac{\|\nabla g_{\tau_i}(\boldsymbol{x}) - \nabla g_{\tau_i}(\boldsymbol{y})\|_2}{\|\boldsymbol{x} - \boldsymbol{y}\|_2} = \frac{n}{b}\|\boldsymbol{A}_{\tau_i}^* \boldsymbol{A}_{\tau_i}\|$$

  - $$\sigma_\tau^2 = d\sum_{i=1}^{d} \|\boldsymbol{A}_{\tau_i}^*(\boldsymbol{A}_{\tau_i}\boldsymbol{x}_* - \boldsymbol{b}_{\tau_i})\|_2^2 \leq d\sum_{i=1}^{d} \|\boldsymbol{A}_{\tau_i}\|^2 \|\boldsymbol{A}_{\tau_i}\boldsymbol{x}_* - \boldsymbol{b}_{\tau_i}\|_2^2$$

Examples in Least Squares

- Orthonormal systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}_{\tau_i}^* \boldsymbol{A}_{\tau_i} \| = \frac{n}{b} = \frac{1}{b}\overline{L}$

Examples in Least Squares

- Orthonormal systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}^*_{\tau_i} \boldsymbol{A}_{\tau_i} \| = \frac{n}{b} = \frac{1}{b}\overline{L}$
- Incoherent (nearly) normalized systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}^*_{\tau_i} \boldsymbol{A}_{\tau_i} \| \le C\frac{n}{b} \le \frac{C}{C'}\frac{\overline{L}}{b}$

Examples in Least Squares

- Orthonormal systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}_{\tau_i}^* \boldsymbol{A}_{\tau_i} \| = \frac{n}{b} = \frac{1}{b}\overline{L}$
- Incoherent (nearly) normalized systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}_{\tau_i}^* \boldsymbol{A}_{\tau_i} \| \leq C\frac{n}{b} \leq \frac{C}{C'}\frac{\overline{L}}{b}$
- Incoherent non-normalized systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}_{\tau_i}^* \boldsymbol{A}_{\tau_i} \| \leq C \sum_{i=1}^{d} \max_{k \in \tau_i} \| \boldsymbol{a}_k \|_2^2$

Examples in Least Squares

- Orthonormal systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}_{\tau_i}^* \boldsymbol{A}_{\tau_i} \| = \frac{n}{b} = \frac{1}{b}\overline{L}$
- Incoherent (nearly) normalized systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}_{\tau_i}^* \boldsymbol{A}_{\tau_i} \| \le C\frac{n}{b} \le \frac{C}{C'}\frac{\overline{L}}{b}$
- Incoherent non-normalized systems: $\overline{L}_\tau = \sum_{i=1}^{d} \| \boldsymbol{A}_{\tau_i}^* \boldsymbol{A}_{\tau_i} \| \le C \sum_{i=1}^{d} \max_{k\in\tau_i} \| \boldsymbol{a}_k \|_2^2$
  - Batching in decreasing arrangement of row norms:

$$\begin{aligned}\overline{L}_\tau &\le C \sum_{i=1}^{d} \| \boldsymbol{a}_{((i-1)b+1)} \|_2^2 \\ &\le \frac{C}{b-1} \sum_{i=1}^{n} \| \boldsymbol{a}_i \|_2^2 \\ &\le \frac{C'}{b}\overline{L}.\end{aligned}$$

Practical Considerations

- How to compute the Lipschitz constants L_{τ_i} ?
  - Use upper bound: maximum row norm in batch
  - Power method:
    - After $T \geq \varepsilon^{-1}\log(\varepsilon^{-1}b)$ iterations, one obtains approximation $\hat{Q}_{\tau_i}$ s.t.

$$\| \boldsymbol{A}^*_{\tau_i} \boldsymbol{A}_{\tau_i} \| \geq \hat{Q}_{\tau_i} \geq \frac{\| \boldsymbol{A}^*_{\tau_i} \boldsymbol{A}_{\tau_i} \|}{1+\varepsilon}$$

which yields

$$\overline{L}_\tau \geq \frac{b}{n} \sum_{i=1}^{d} \frac{n}{b} \hat{Q}_{\tau_i} \geq \frac{\overline{L}_\tau}{1+\varepsilon}$$

at a computational cost (over b cores) of just $b\varepsilon^{-1}\log(\varepsilon^{-1}\log(b))$

Non-smooth Hinge Loss

Corollary 4.3. *Consider* $P(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n}[y_i\langle \boldsymbol{x}, \boldsymbol{a}_i\rangle]_+ + \frac{\lambda}{2}\|\boldsymbol{x}\|_2^2$. *Consider the batched weighted SGD iteration*

$$\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k - \frac{1}{\mu k p(\tau_i)}\left(\lambda \boldsymbol{x}_k + \frac{1}{b}\sum_{j\in\tau_i}\chi_j(\boldsymbol{x}_k) y_j \boldsymbol{a}_j\right), \tag{4.5}$$

where $\chi_j(\boldsymbol{x}) = 1$ *if* $y_j\langle \boldsymbol{x}, \boldsymbol{a}_j\rangle < 1$ *and* 0 *otherwise. Let* $\boldsymbol{A}_\tau$ *have rows* $y_j\boldsymbol{a}_j$ *for* $j \in \tau$. *For any desired* ε, *we have that after*

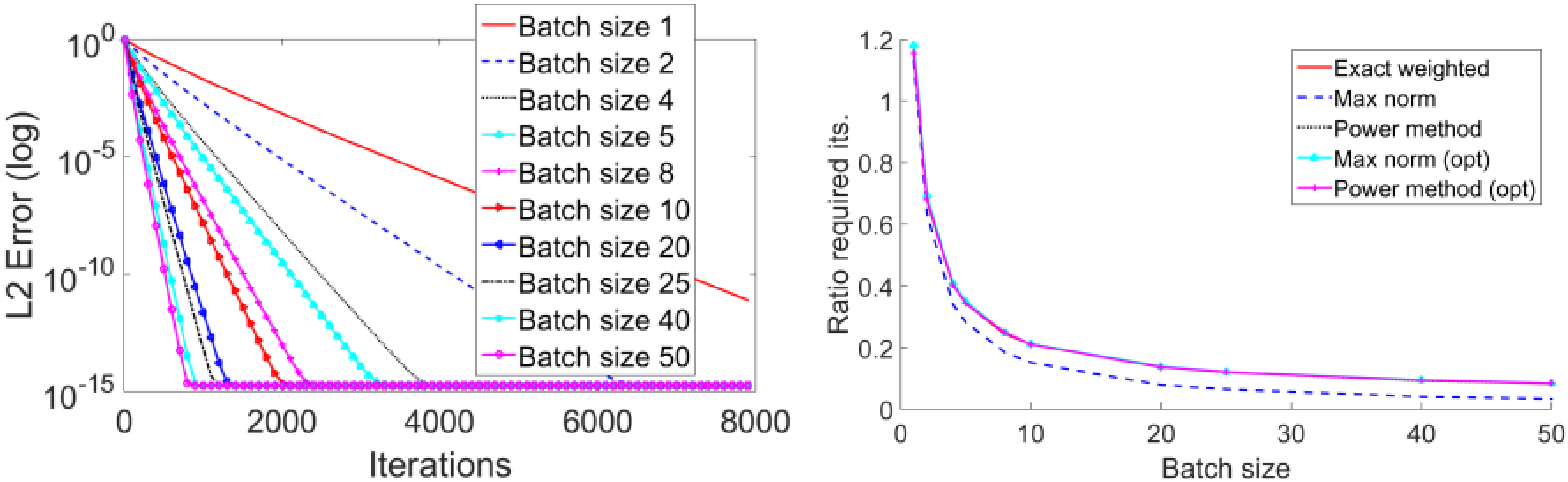
$$k = \frac{C\min(\alpha, 1-\alpha)\left(\lambda + \frac{\sqrt{b}}{n}\sum_{i=1}^{d}\|\boldsymbol{A}_{\tau_i}\|\right)^2}{\lambda\varepsilon} \tag{4.6}$$

iterations of (4.5) *with weights*

$$p(\tau_i) = \frac{\|\boldsymbol{A}_{\tau_i}\| + \lambda\sqrt{b}}{\frac{n}{\sqrt{b}}\lambda + \sum_j \|\boldsymbol{A}_{\tau_j}\|}, \tag{4.7}$$

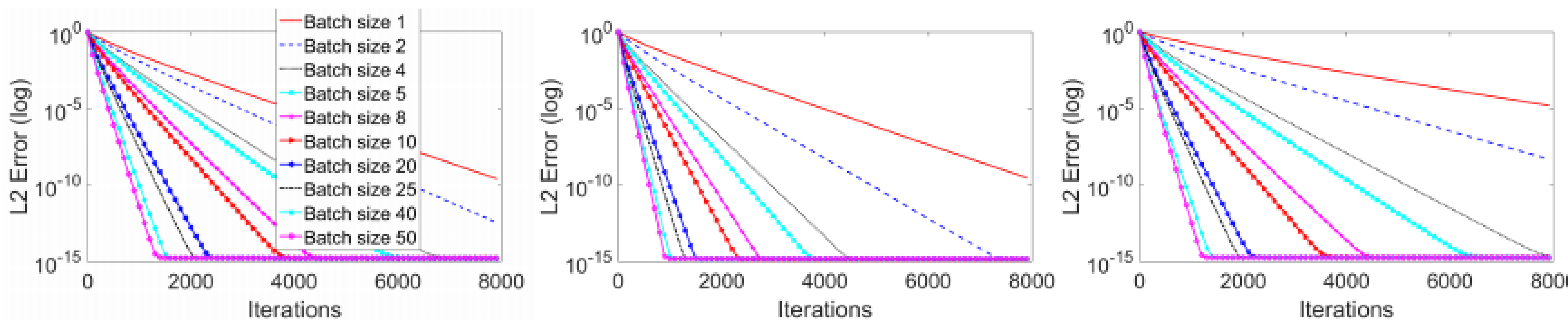
it holds that $\mathbb{E}^{(p)}[P(\mathbf{x}_k) - P(\mathbf{x}_*)] \le \varepsilon$.

Least Squares Experiments



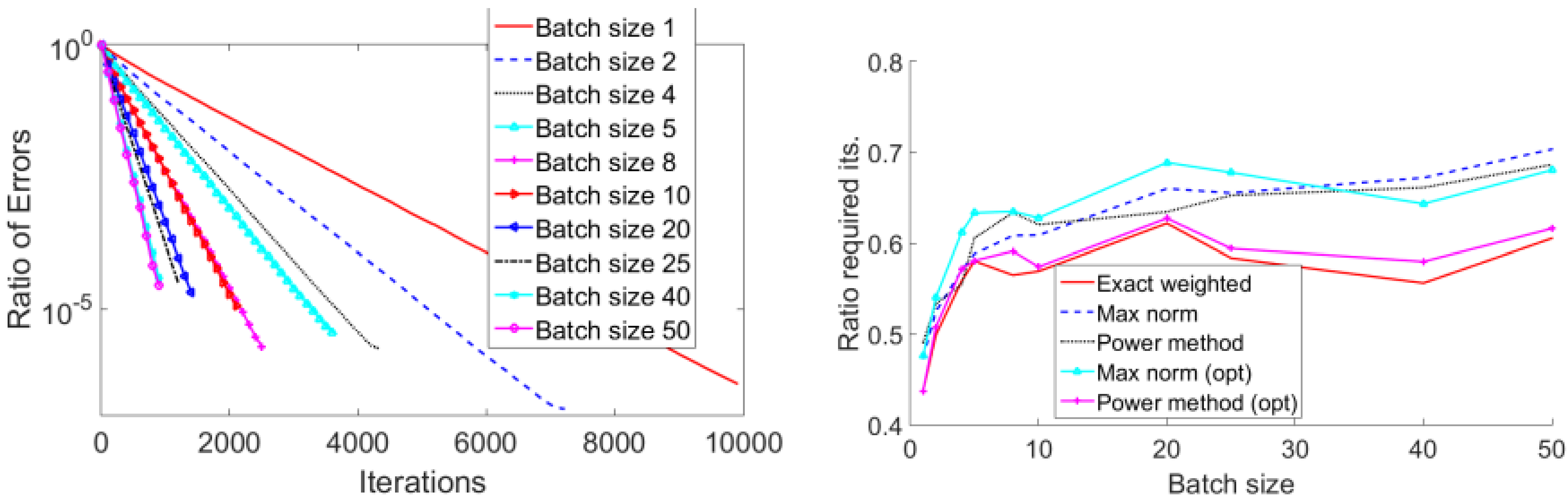
Gaussian systems. Right: Ratio of required iterations to reach error tolerance for batched SGD with weighting compared to classical SGD. "(opt)" denotes optimal step size was used.

Least Squares Experiments



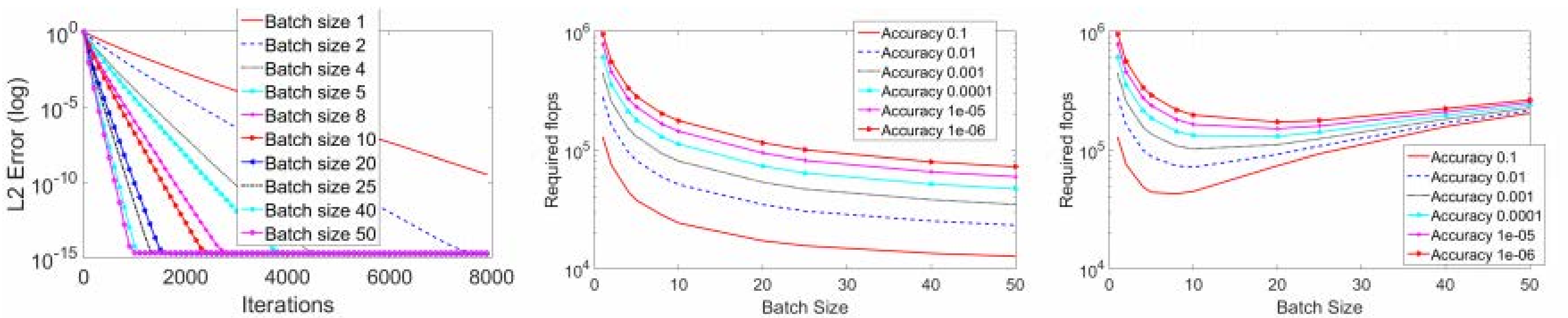
Gaussian systems with varying row norms. Left: Random batches, weighted sampling. Center: Sequential batches, weighted SGD. Right: Sequential batched, unweighted SGD.

Least Squares Experiments: weighting



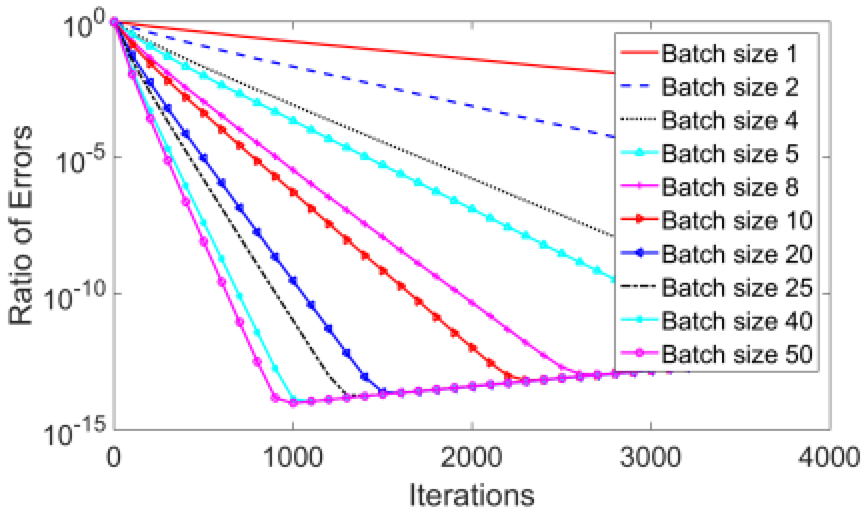
Gaussian systems with varying row norms. Left: Error ratios for weighted vs. unweighted SGD. Right: Ratio of required iterations to reach error tolerance for weighted versus unweighted SGD. "(opt)" denotes optimal step size was used.

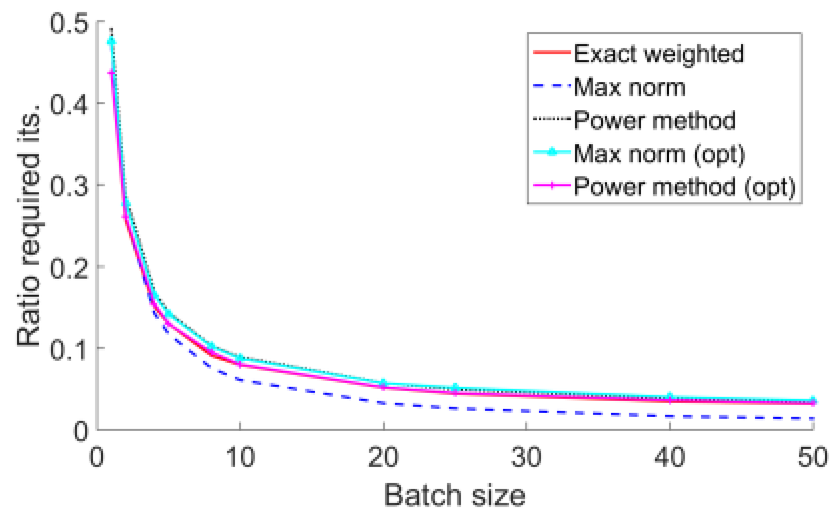
Least Squares Experiments: batching



Gaussian systems with varying row norms. Left: Error ratios for batched weighted SGD versus classical. Right: Ratio of required iterations to reach error tolerance for batched weighted SGD versus classical. "(opt)" denotes optimal step size was used.

Least Squares Experiments: power method





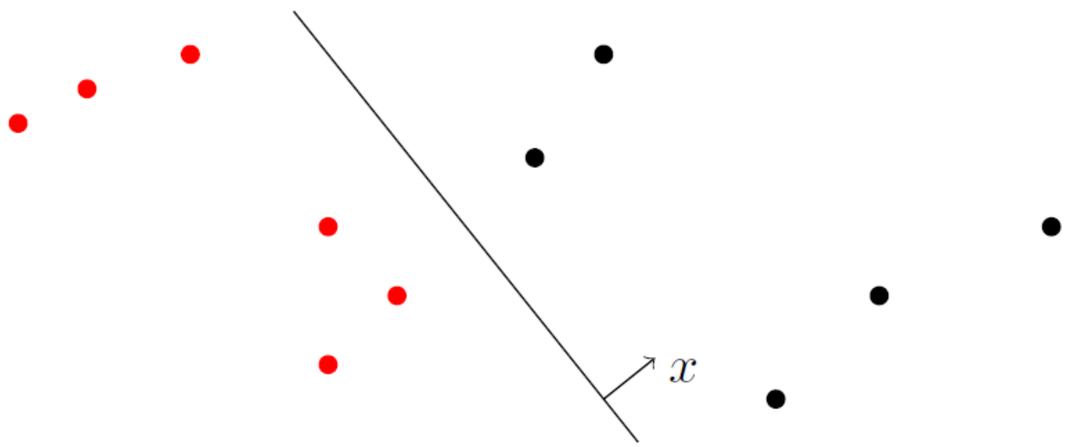
Gaussian systems with varying row norms. Left: Convergence. Center: Flops versus batch size to achieve error tolerance, shared over b cores. Right: Flops versus batch size to achieve error tolerance (single core).

Linear Feasibility

- SVM Classification

Given binary classified training data, $\{(a_i, y_i)\}_{i=1}^m$ where $a_i \in \mathbb{R}^{n-1}$ and

$$y_i = \begin{cases} 1 & \text{if } a_i \in \text{ class } 1 \\ -1 & \text{if } a_i \in \text{ class } 2 \end{cases}$$



find a linear classifier $F(a_i) = x^T a_i + z$ so that

$$y_i F(a_i) \geq 0 \text{ for all } i = 1, \ldots, m.$$

Linear Feasibility

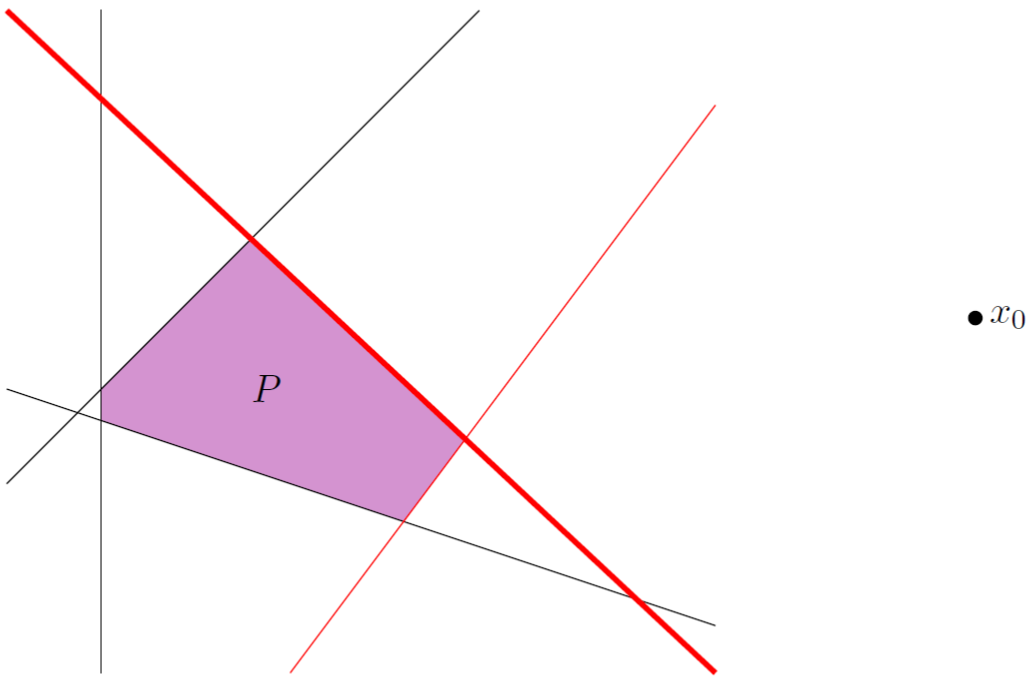
- Method of Motzkin [`54] to find point in polytope P given by Ax < b:

Given $x_0 \in \mathbb{R}^n$, fix $0 < \lambda \leq 2$ and iteratively construct approximations to P:

1. If x_k is feasible, stop.
2. Choose $i_k \in [m]$ as $i_k := \operatorname*{argmax}_{i \in [m]} a_i^T x_{k-1} - b_i$.
3. Define $x_k := x_{k-1} - \lambda \frac{a_{i_k}^T x_{k-1} - b_{i_k}}{||a_{i_k}||^2} a_{i_k}$.
4. Repeat.

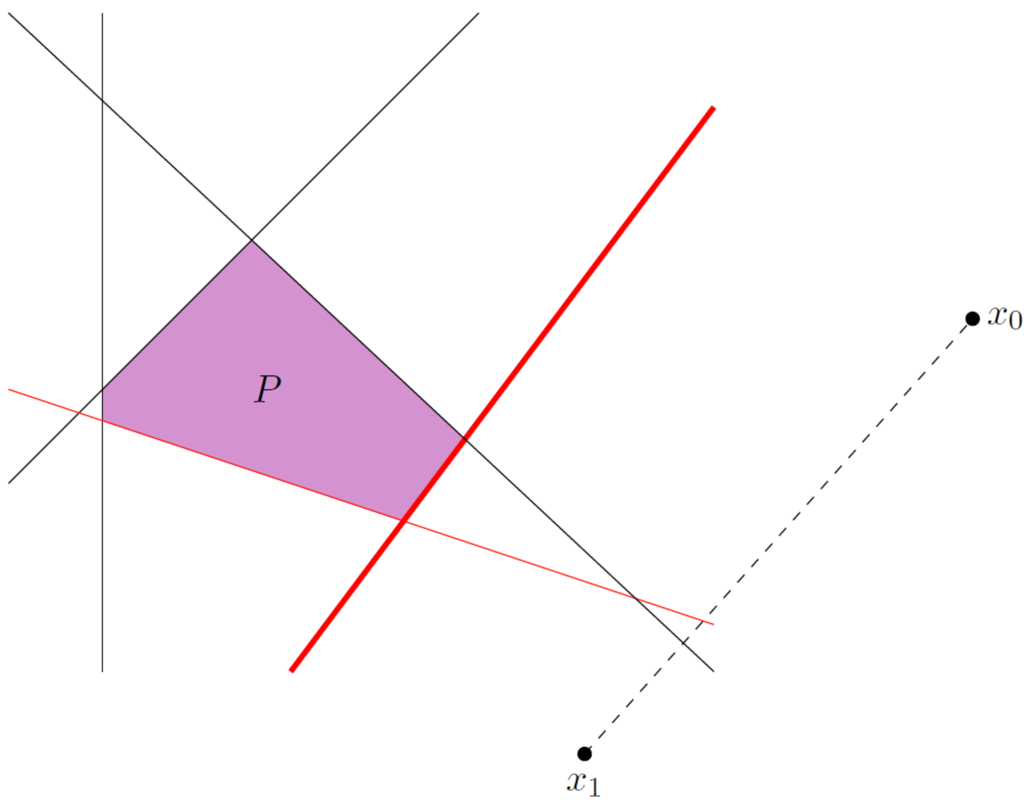
Linear Feasibility

- Method of Motzkin [`54] to find point in polytope P given by Ax < b:



Linear Feasibility

- Method of Motzkin [`54] to find point in polytope P given by Ax < b:



Linear Feasibility

- Method of Motzkin [`54] to find point in polytope P given by Ax < b:
  - Pros: Monotonically decreasing, accelerated convergence
  - Cons: Computationally expensive
  - Motivation: Use batched version of Motzkin's Method

Batched Motzkin's Method

Given $x_0 \in \mathbb{R}^n$, fix $0 < \lambda \leq 2$ and iteratively construct approximations to P in the following way:

1. If x_k is feasible, stop.
2. Choose $\tau_k \subset [m]$ to be a sample of size β constraints chosen uniformly at random from among the rows of A.
3. From among these β rows, choose $i_k := \operatorname*{argmax}_{i \in \tau_k} a_i^T x_{k-1} - b_i$.
4. Define $x_k := x_{k-1} - \lambda \frac{(a_{i_k}^T x_{k-1} - b_{i_k})^+}{||a_{i_k}||^2} a_{i_k}$.
5. Repeat.

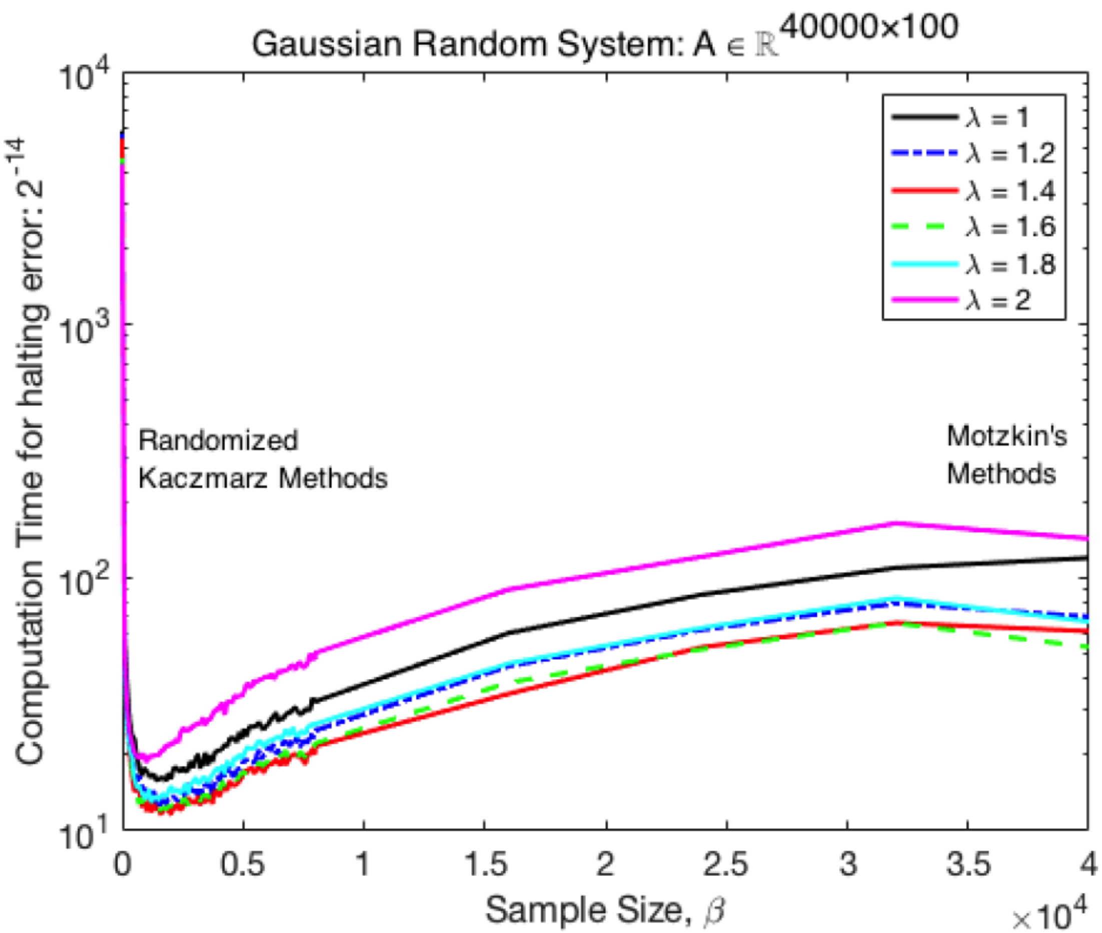
Batched Motzkin's Method

Let H denote the Hoffman constant (~ conditioning) of the system. Then:

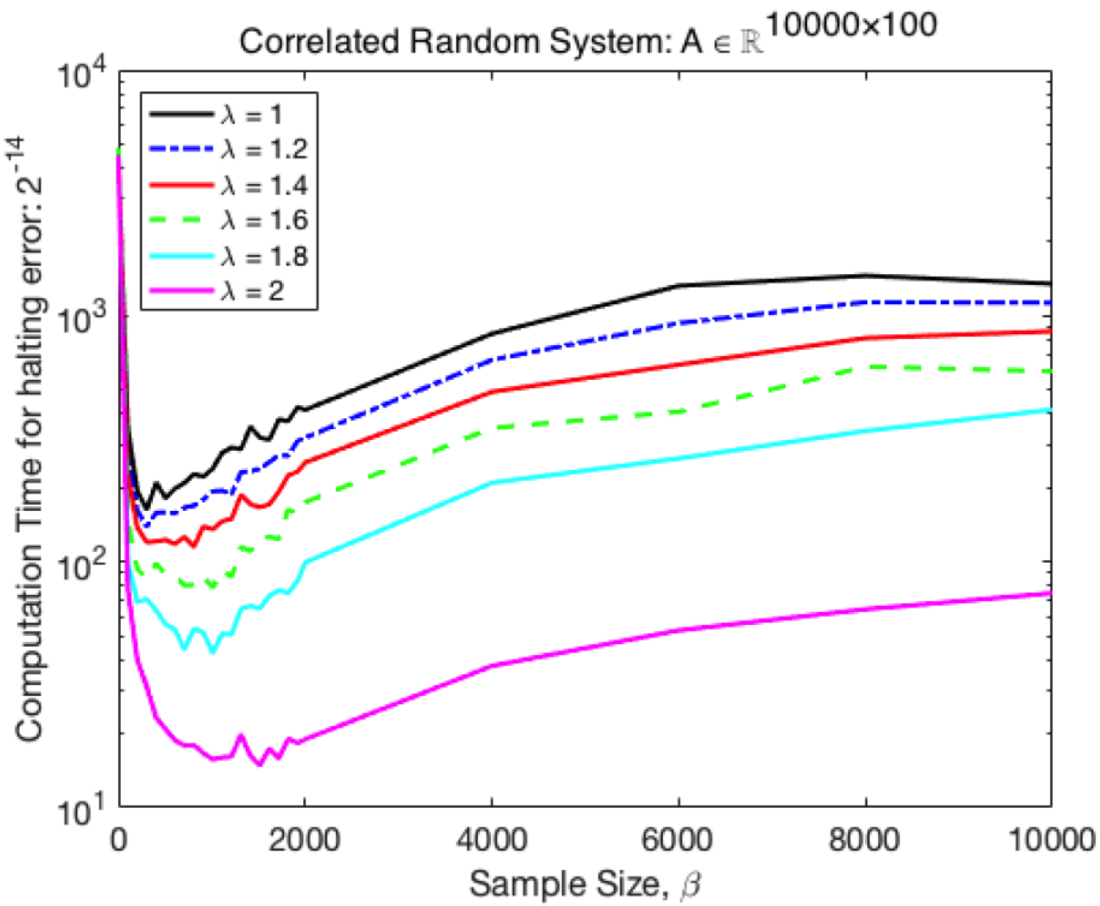
If the feasible region (for normalized A) is nonempty, then the SKM methods with samples of size β converge at least linearly in expectation:
Let s_{k-1} be the number of constraints satisfied by x_{k-1} and $V_{k-1} := \max\{m - s_{k-1}, m - \beta + 1\}$. Then, in the kth iteration,

$$\mathbb{E}\left[d(x_k, P)^2\right] \leq \left(1 - \frac{2\lambda - \lambda^2}{V_{k-1} H_2^2}\right) d(x_{k-1}, P)^2$$

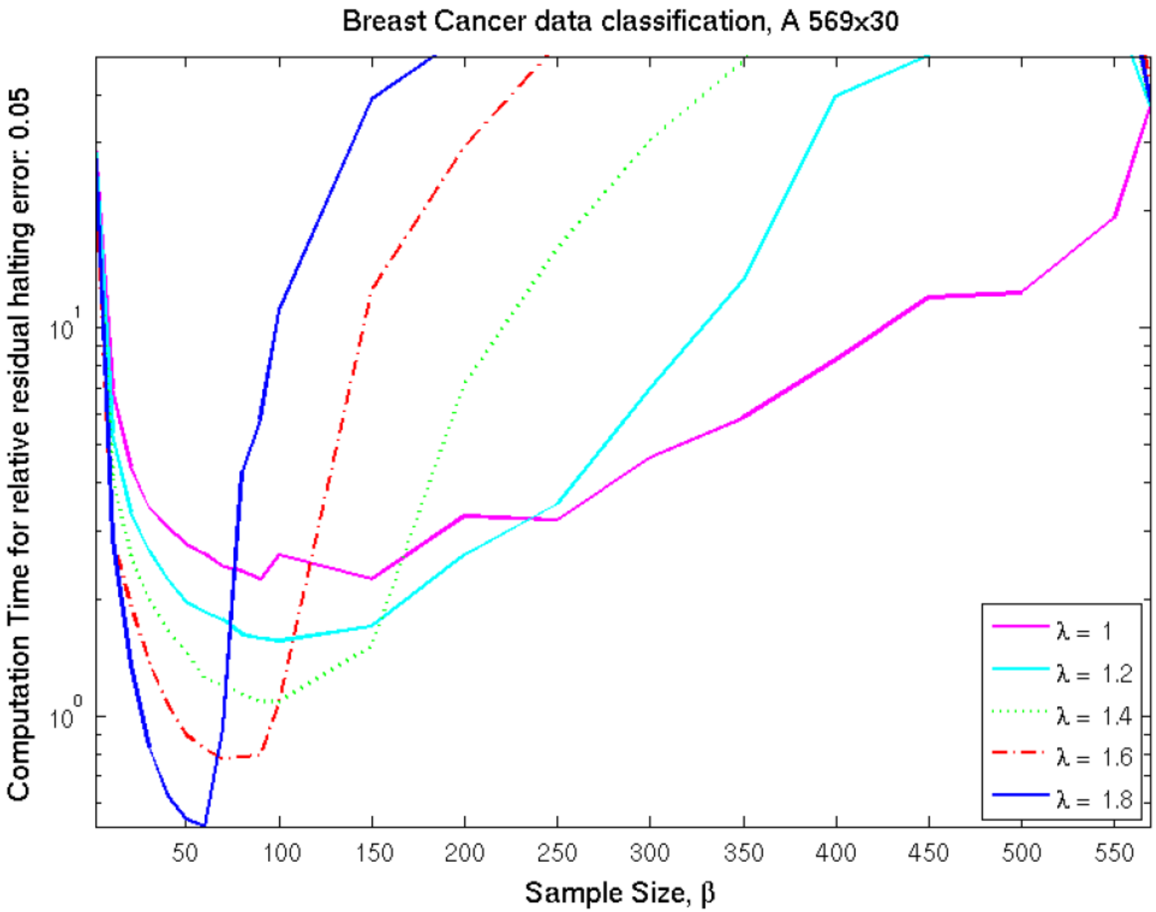
Batched Motzkin's Method



Batched Motzkin's Method



Batched Motzkin's Method



Thank you!

"Batched Stochastic Gradient Descent with Weighted Sampling"
by D. Needell and R. Ward.
Submitted.

"A Sampling Kaczmarz-Motzkin Algorithm for Linear Feasibility"
by J. A. De Loera, J. Haddock, D. Needell.
Submitted.

"Stochastic Gradient Descent and the Randomized Kaczmarz algorithm"
by D. Needell, N. Srebro, R. Ward.
Mathematical Programming Series A, vol. 155, num. 1, 549 - 573, 2016.

www.cmc.edu/pages/faculty/DNeedell deanna@math.ucla.edu