RCD SCD RCDM Shotgun UCDC RCDC PCDM SDCA RCD mSDCA ICD ASDCA RBCD ACDM
Acc-Prox-SDCA SPCDM Hydra Nsync AsySCD RCM APPROX DisDCA I-Prox-SDCA Asy-SPCD
DBCD Hydra^2 DBCD APCG SPDC CoCoA Quartz S2CD ALPHA SDNA CoCoA+ AdaSDCA dfSDCA

Modern Convex Optimization Methods for Large-scale Empirical Risk Minimization (Part II: Dual Methods)

International Conference on Machine Learning
July 2015

Peter Richtárik and Mark Schmidt

Introduction

EMPIRICAL RISK MINIMIZATION

Primal Problem: ERM

$1/\gamma$ - smooth & convex

regularization parameter

$$\min_{w \in \mathbb{R}^d} \left[P(w) \equiv \frac{1}{n} \sum_{i=1}^{n} \phi_i(A_i^\top w) + \lambda g(w) \right]$$

d = # features (parameters)

n = # samples

1 - strongly convex function (regularizer)

Is the difficulty in *n* or *d*?

- **Big n**
  - Work in the **primal**
  - Process one loss function (= one example) at a time
  - Type of methods: stochastic gradient descent (modern variants: SAG, SVRG, S2GD, mS2GD, SAGA, S2CD, MISO, FINITO, …)
- **Big d**
  - Work in the **primal**
  - Process one primal variable at a time
  - Type of methods: randomized coordinate descent (e.g., Hydra, Hydra2)
- **Big n**
  - Work in the **dual**
  - Process one dual variable (=one example) at a time
  - Type of methods: randomized coordinate descent (modern variants: RCDM, PCDM, Shotgun, SDCA, APPROX, Quartz, ALPHA, SDNA, SPDC, ASDCA, …)
  - E.g. SDCA = run coordinate descent on the dual problem

Dual Problem

$$D(\alpha) \equiv -\lambda g^*\left(\underbrace{\frac{1}{\lambda n}\sum_{i=1}^{n} A_i\alpha_i}_{\in \mathbb{R}^d}\right) - \frac{1}{n}\sum_{i=1}^{n}\phi_i^*(\overbrace{-\alpha_i}^{\in \mathbb{R}^m})$$

1 – smooth & convex (g^*)

γ - strongly convex (ϕ_i^*)

$$g^*(w') = \max_{w \in \mathbb{R}^d} \left\{ (w')^\top w - g(w) \right\} \qquad \phi_i^*(a') = \max_{a \in \mathbb{R}^m} \left\{ (a')^\top a - \phi_i(a) \right\}$$

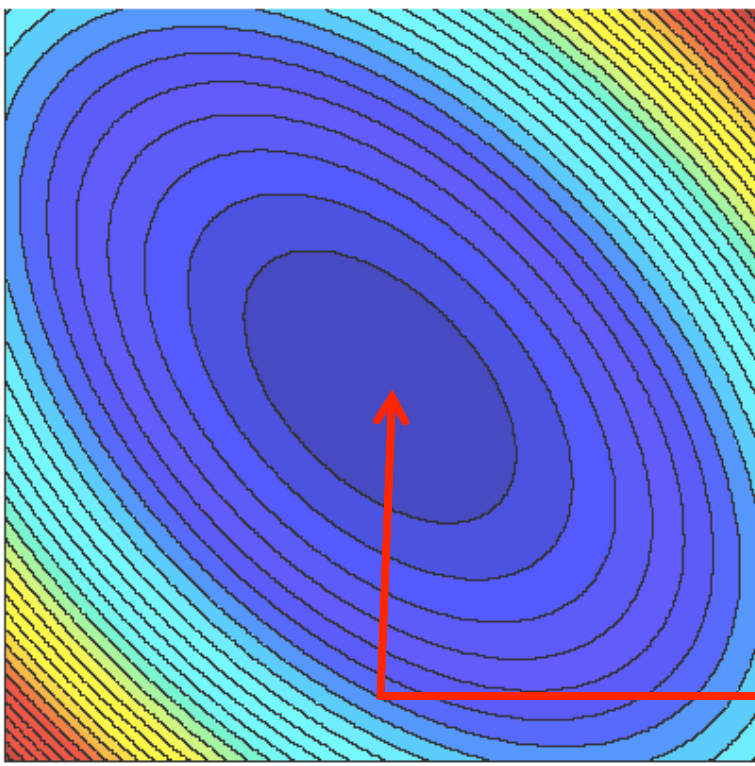
$$\max_{\alpha = (\underbrace{\alpha_1}_{\in \mathbb{R}^m}, \ldots, \underbrace{\alpha_n}_{\in \mathbb{R}^m}) \in \mathbb{R}^N = \mathbb{R}^{nm}} D(\alpha)$$

RANDOMIZED COORDINATE DESCENT

Coordinate Descent in 2D

Contours of a function

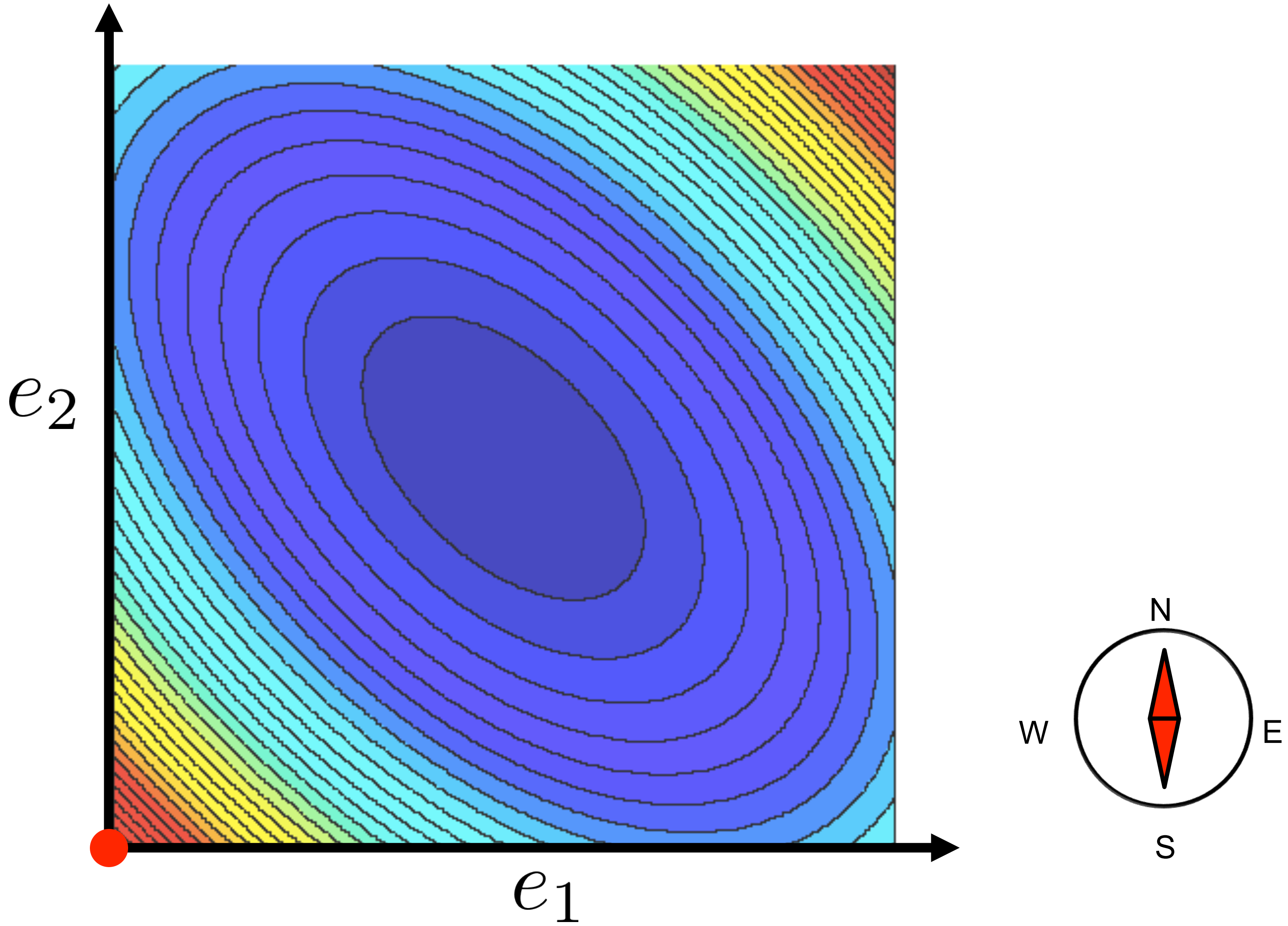
$F : \mathbf{R}^2 \rightarrow \mathbf{R}$



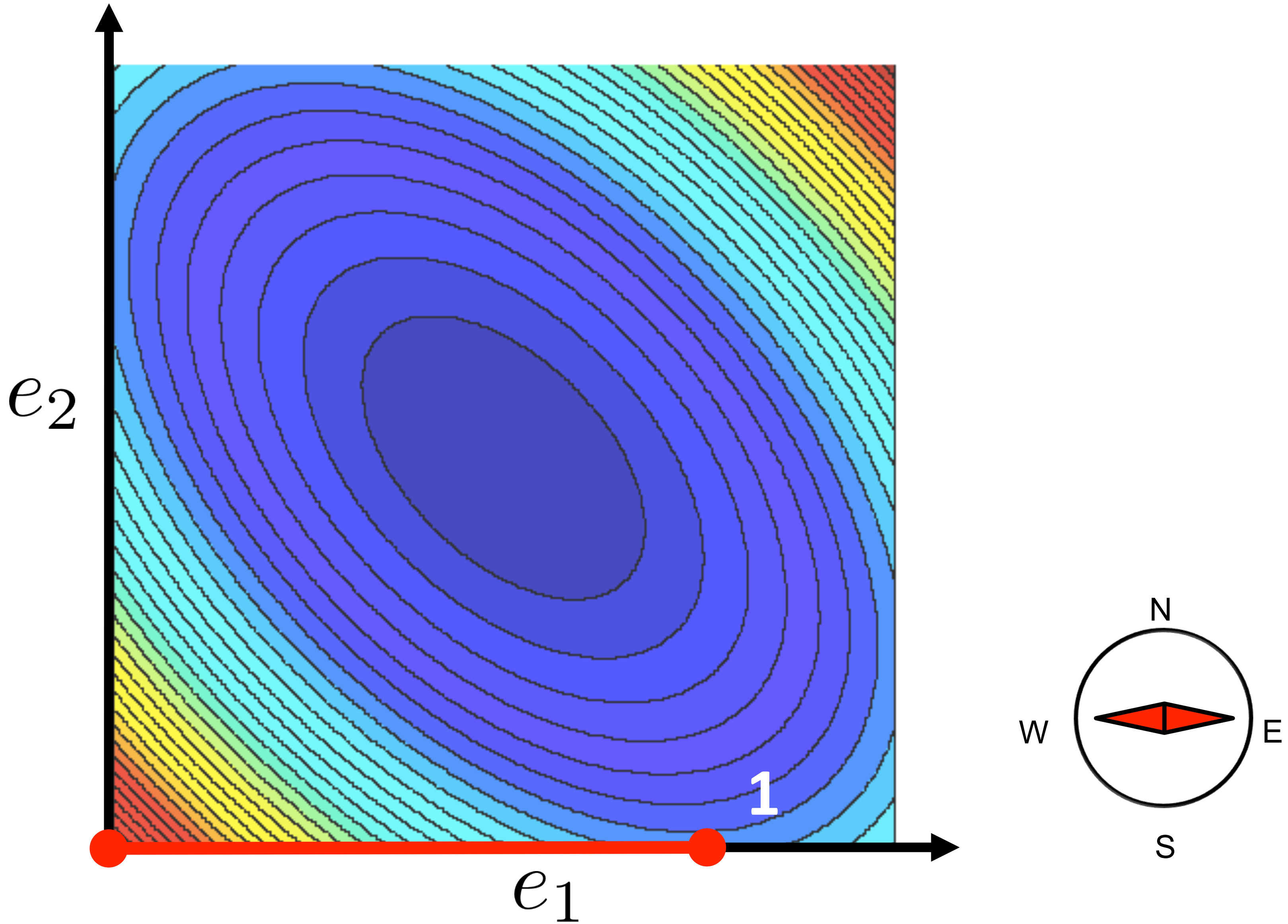
Goal:

Find the minimizer of F

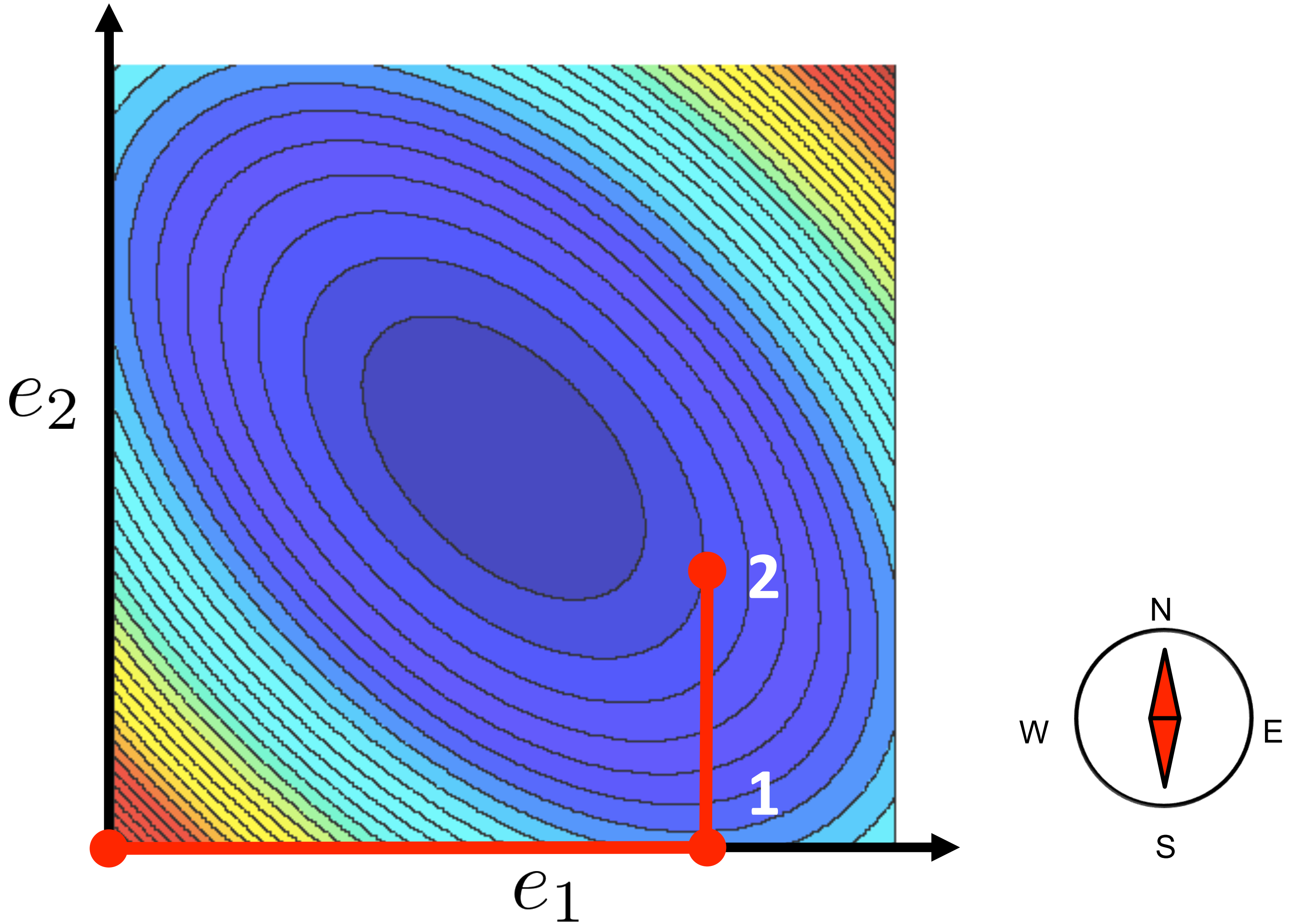
Randomized Coordinate Descent in 2D



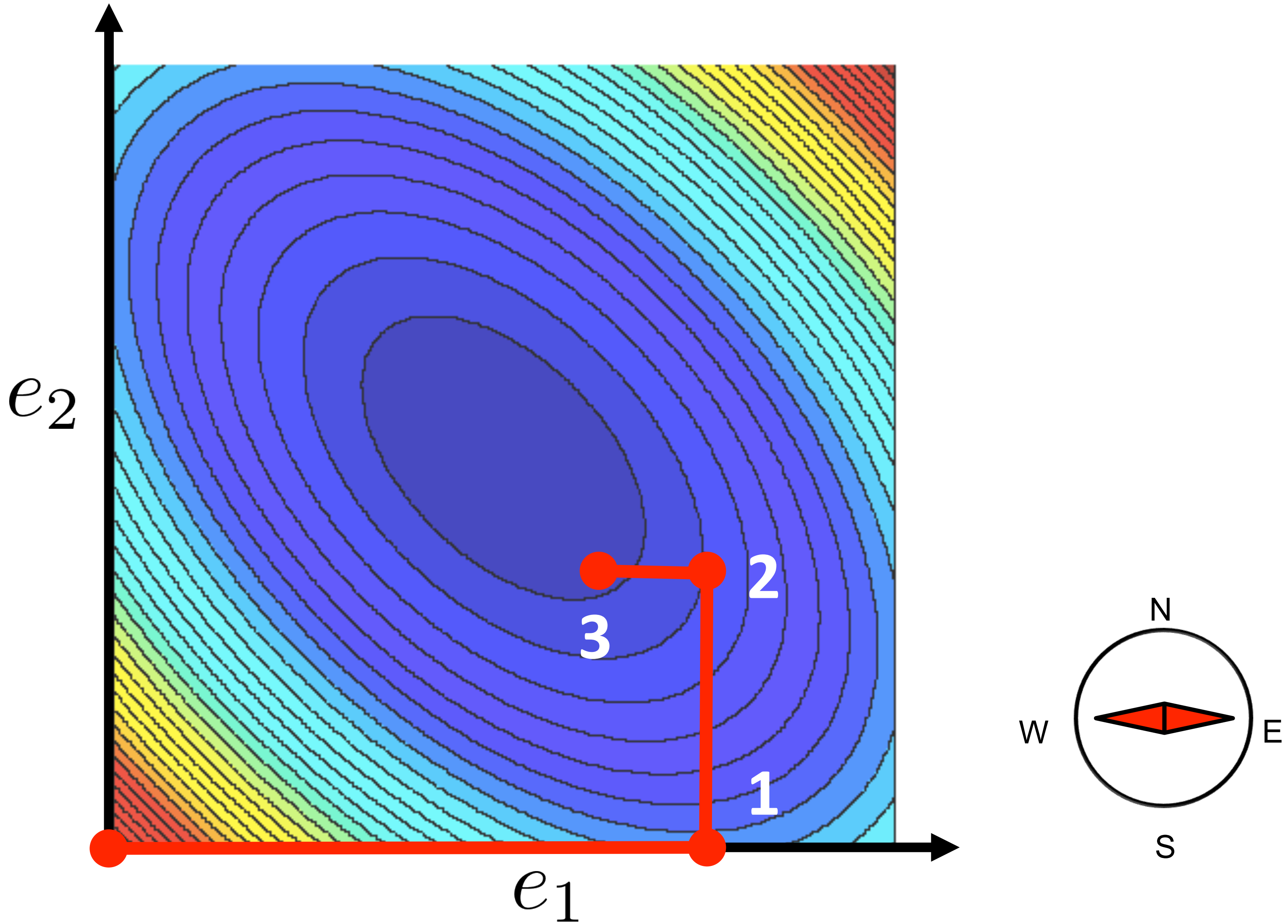
Randomized Coordinate Descent in 2D



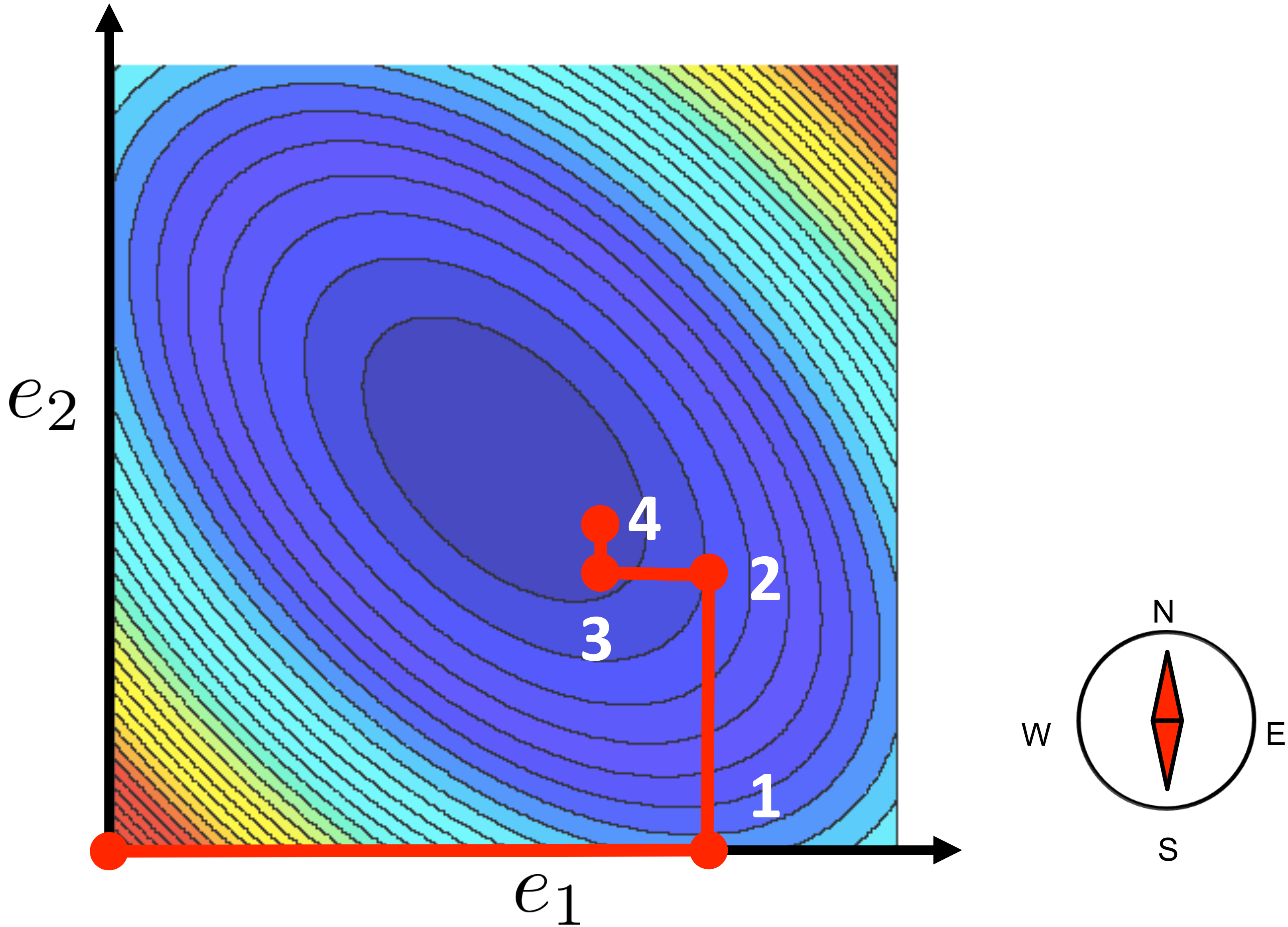
Randomized Coordinate Descent in 2D



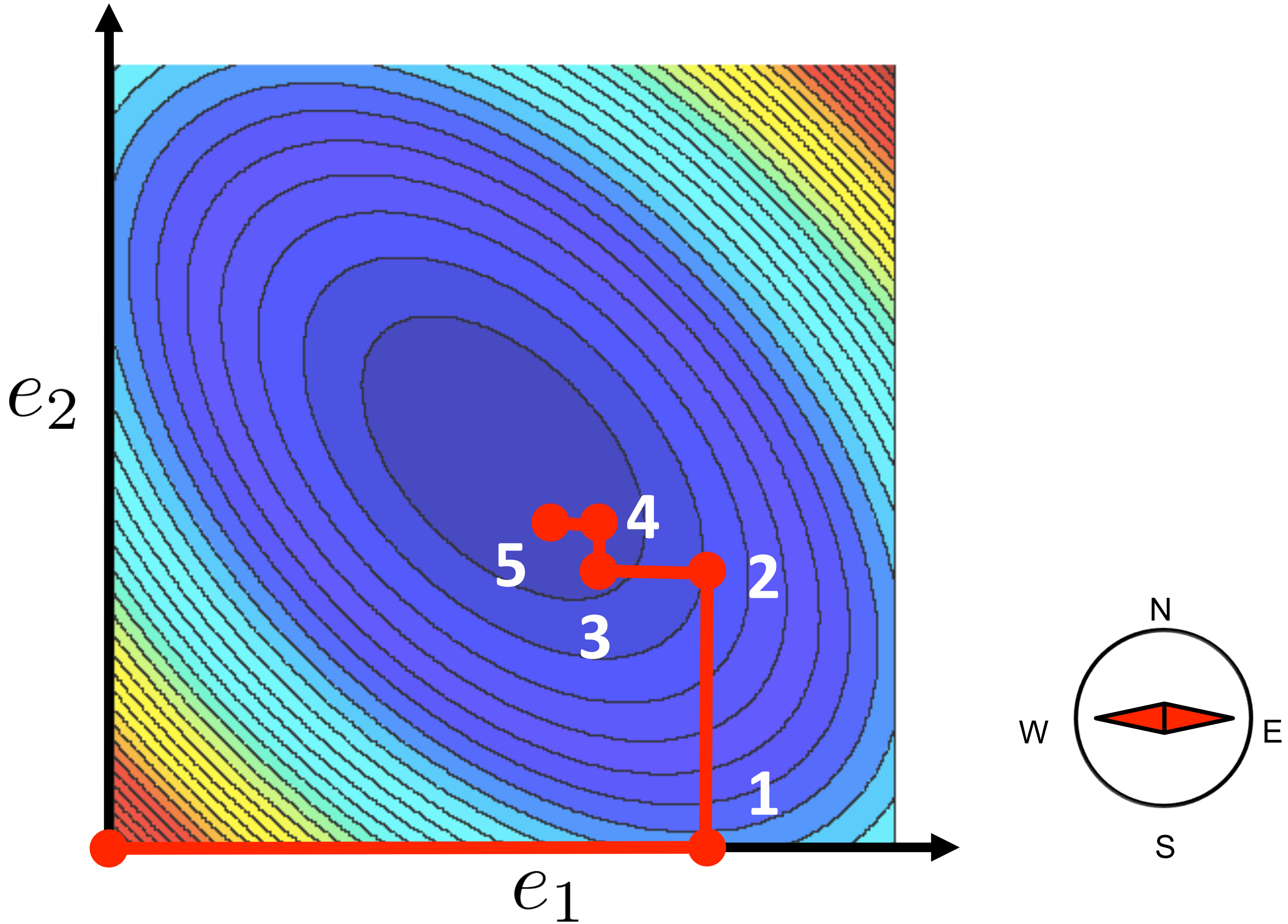
Randomized Coordinate Descent in 2D



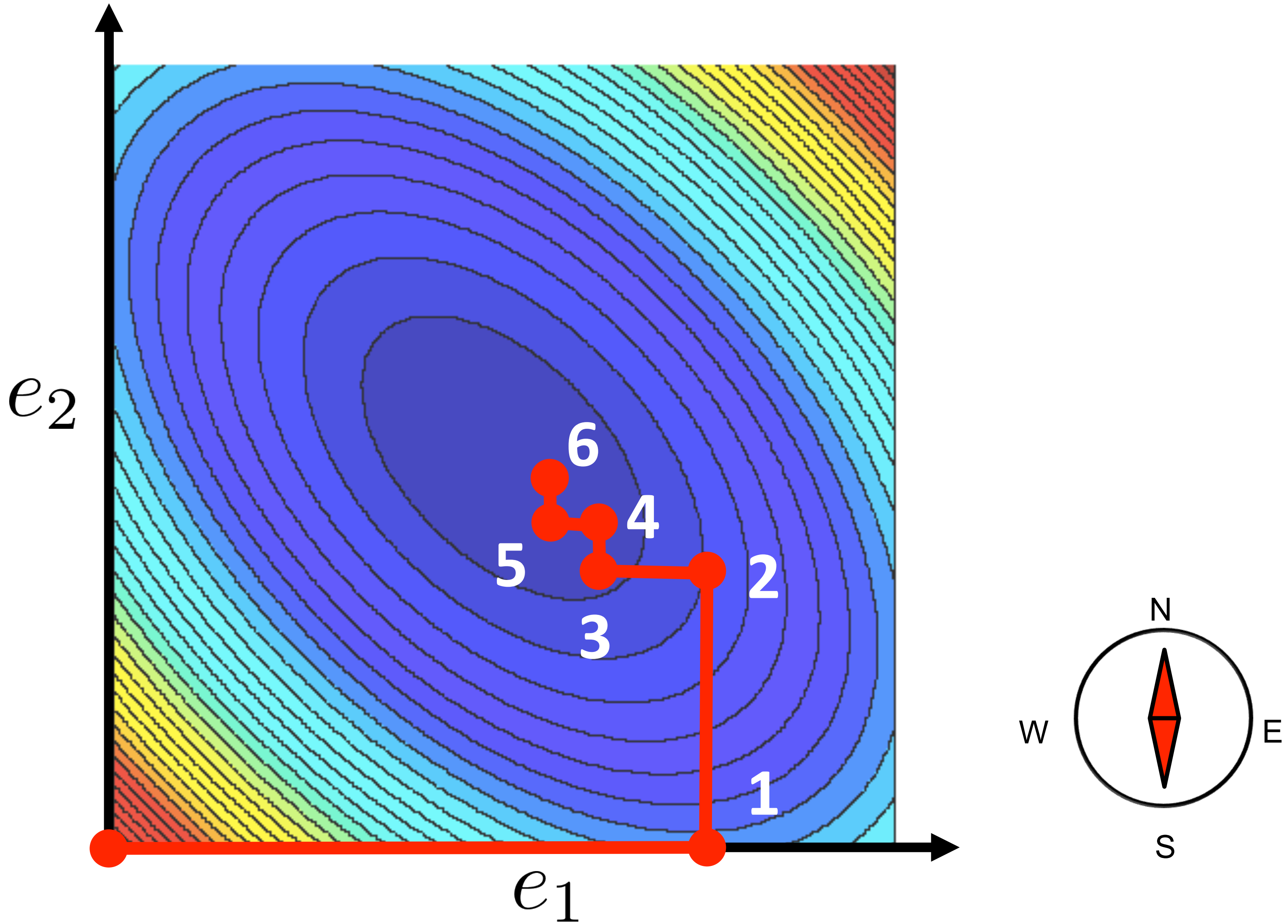
Randomized Coordinate Descent in 2D



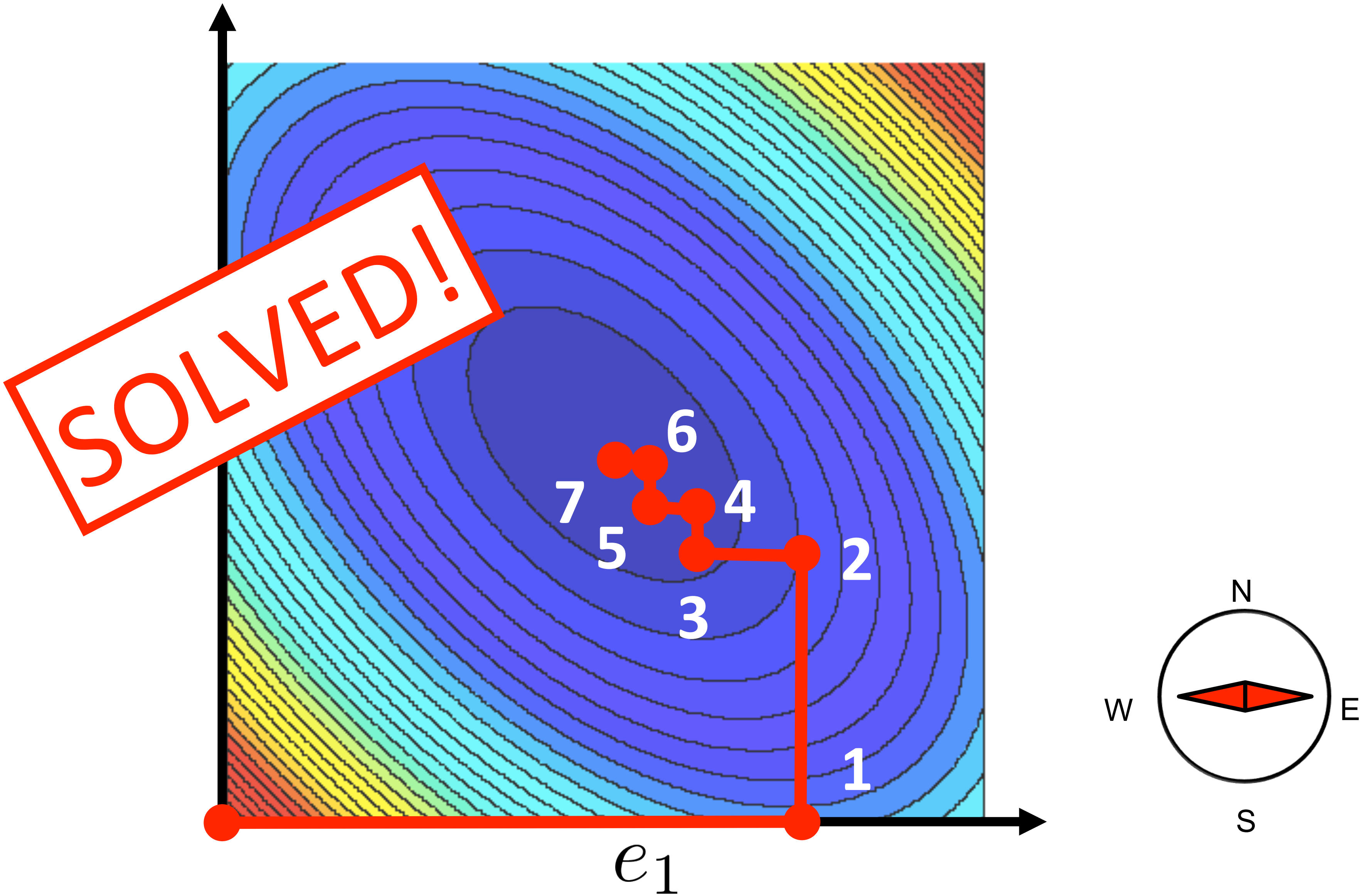
Randomized Coordinate Descent in 2D



Randomized Coordinate Descent in 2D



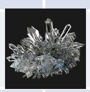
Randomized Coordinate Descent in 2D



BIBLIOGRAPHY (Randomized Coordinate Descent)

Citation	Algorithm	Paper
[Leventhal & Lewis 08]	RCD	**Randomized methods for linear constraints: convergence rates and conditioning.** *Mathematics of OR* 35(3), 641-654, 2010 (arXiv:0806.3015)
[S-Shwartz & Tewari 09]	SCD	**Stochastic methods for L1-regularized loss minimization.** *ICML* 2009
[Nesterov 10]	UCDM, RCDM, ACDM	**Efficiency of coordinate descent methods on huge-scale optimization problems.** *SIAM J. on Optimization*, 22(2):341–362, 2012 (CORE Discussion Paper 2010/2)
[Bradley et al 11]	Shotgun 	**Parallel coordinate descent for L1-regularized loss minimization**. *ICML*, 2011 (arXiv: 1105.5379)
[R & Takáč 11a]	SCD	**Efficient serial and parallel coordinate descent methods for huge-scale truss topology design.** *Operations Research Proceedings*, 27-32, 2012 (Opt Online 08/2011)
[R & Takáč 11b]	UCDC, RCDC	**Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function.** *Mathematical Programming* 144(2), 1-38, 2014 (arXiv:1107.2848)
[R & Takáč 12]	PCDM	**Parallel coordinate descent methods for big data optimization.** *Mathematical Programming*, 2015 (arXiv:1212.0873)
[S-Shwartz & Zhang 12]	SDCA	**Stochastic dual coordinate ascent methods for regularized loss minimization.** *JMLR* 14, 567-599, 2013 (arXiv:1209.1873)
[Necoara & Clipici 13]	RCD	**A random coordinate descent algorithm for optimization problems with composite objective function and linear coupled constraints.** COAP 57(2), 303-337, 2014 (arXiv: 1302.3074)
[Takáč et al 13]	mSDCA	**Mini-batch primal and dual methods for SVMs.** *ICML* 2013 (arXiv:1303.2314)
[Tappenden, R, & Gondzio 13]	ICD	**Inexact coordinate descent.** arXiv:1304.5530, 2013
[S-Shwartz & Zhang 13a]	ASDCA	**Accelerated mini-batch stochastic dual coordinate ascent.** *NIPS* 2013 (arXiv: 1305.2581)

Citation	Algorithm	Paper
[Lu & Xiao 13]	RBCD	**On the complexity analysis of randomized block-coordinate descent methods.** *Mathematical Programming*, 2014 (arXiv:1305.4723)
[Patrascu & Necoara 13]		**Efficient random coordinate descent algorithms for large-scale structured nonconvex optimization**. *J of Global Optimization* 61(1), 19-46 (arXiv:1305.4027)
[Lee & Sidford 13]	ACDM	**Efficient accelerated coordinate descent methods and faster algorithms for solving linear systems.** *FOCS* 2013 (arXiv:1305.1922)
[Tappenden, R & Buke 13]	DQA vs PCDM	**Separable approximations and decomposition methods for the augmented Lagrangian.** *Optimization Methods and Software* 30(3), 643-668, 2015 (arXiv:1308.6774)
[S-Shwartz & Zhang 13b]	Acc Prox-SDCA	**Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization.** *Mathematical Programming* 2014 (arXiv:1309.2375)
[Fercoq & R 13a]	SPCDM	**Smooth minimization of nonsmooth functions with parallel coordinate descent methods.** arXiv:1309.5885, 2013
[R & Takáč 13a]	HYDRA	**Distributed coordinate descent method for learning with big data.** arXiv:1310.2059, 2013
[R & Takáč 13b]	SYNC	**On optimal probabilities in stochastic coordinate descent methods.** *Opt. Letters*, 2015 (arXiv:1310.3438)
[Liu et al 13]	AsySCD	**An asynchronous parallel stochastic coordinate descent algorithm.** *ICML* 2014 (arXiv:1311.1873)
[Shalit & Chechik 13]	RCM	**Efficient coordinate-descent for orthogonal matrices through Givens rotations.** *ICML* 2014 (arXiv:1312.0624)
[Fercoq & R 13b]	APPROX	**Accelerated, parallel and proximal coordinate descent.** arXiv:1312.5799, 2013
[Yang 13]	DisDCA	**Trading computation for communication: distributed stochastic dual coordinate ascent.** *NIPS* 2013
[Zhao & Zhang 14]	I-Prox SDCA, I-Prox SGD	**Stochastic optimization with importance sampling.** ICML 2015, arXiv:1401.2753, 2014

Citation	Algorithm	Paper
[Liu & Wright 14]	AsySPCD	**Asynchronous stochastic coordinate descent: parallelism and convergence properties.** *SIAM J. on Optimization*, 25(1), 351–376, 2015 (arXiv:1403.3862)
[Mahajan, Keerthi & Sundararajan 14]	DBCD	**A distributed block coordinate descent method for training l1 regularized linear classifiers.** arXiv:1405.4544, 2014
[Fercoq et al 14]	Hydra2	**Fast distributed coordinate descent for non-strongly convex losses.** *MLSP* 2014 (arXiv:1405.5300)
[Mareček, R and Takáč 14]	DBCD	**Distributed block coordinate descent for minimizing partially separable functions.** Numerical Analysis and Opt., Springer Proc. in Math. and Stat. (arXiv:1406.0238)
[Lin, Lu & Xiao 14]	APCG	**An accelerated proximal coordinate gradient method and its application to regularized empirical risk minimization.** *NIPS* 2014 (arXiv:1407.1296)
[Zhang & Xiao 14]	SPDC	**Stochastic primal-dual coordinate method for regularized empirical risk minimization.** *ICML* 2015 (arXiv:1409.3257)
[Jaggi, Smith, Takáč et al 14]	CoCoA	**Communication-efficient distributed dual coordinate ascent.** *NIPS* 2014 (arXiv:1409.1458)
[Qu, R & Zhang 14] 	QUARTZ	**Randomized dual coordinate ascent with arbitrary sampling.** arXiv:1411.5873, 2014
[Konečný, Q & R 14]	S2CD	**Semi-stochastic coordinate descent**. *NIPS* Optimization Workshop, 2014 (arXiv:1412.6293)
[Qu and R 14a]	ALPHA α	**Coordinate descent with arbitrary sampling I: algorithms and complexity.** arXiv:1412.8060, 2014
[Qu and R 14b]		**Coordinate descent with arbitrary sampling II: expected separable overapproximation.** arXiv:1412.8063, 2014
[Qu et al 15]	SDNA	**SDNA: Stochastic dual newton ascent for empirical risk minimization.** arXiv:1502.02268, 2015
[Ma, Smith, Jaggi et al 15]	CoCoA+	**Adding vs. averaging in distributed primal-dual optimization.** *ICML* 2015

Citation	Algorithm	Paper
[Tappenden, Takáč & R 15]	PCDM	**On the complexity of parallel coordinate descent.** arXiv:1503.03033, 2015
[Csiba, Qu & R 15]	AdaSDCA	**Stochastic dual coordinate ascent with adaptive probabilities.** *ICML* 2015
[Ene & Nguyen 15]	RCDM, APPROX	**Random coordinate descent methods for minimizing decomposable submodular functions.** *ICML* 2015 (arXiv:1502.02643)
[S-Shwartz 15]	SDCA	**SDCA without duality.** arXiv:1502.06177, 2015
[Csiba & R 15]	dfSDCA	**Primal method for ERM with flexible mini-batching schemes and non-convex losses**. arXiv:1506.02227, 2015
[Wright 15]		**Coordinate descent algorithms.** *Mathematical Programming* 151(1), 3-34, 2015 (arXiv:1502.04759)
[Gower & R 15]		**Randomized iterative methods for linear systems.** arXiv:1506.03296, 2015
[Nutini et al 15]		**Coordinate descent converges faster with the Gauss-Southwell rule than random selection.** *ICML* 2015

Coordinate Descent Tricks

Trick 1: Arbitrary Sampling
Trick 2: Acceleration
Trick 3: Duality
Trick 4: Curvature
Trick 5: Parallelization / Minibatching
Trick 6: Distributed Implementation
Trick 7: Line-search RCDM [Nesterov 10]
Trick 8: Inexactness ICD [Tappenden, R & Gondzio 13]
Trick 9: Asynchronicity AsySCD [Liu et al 13]
Trick 10: Adaptivity AdaSDCA [Csiba, Qu and R 15]

Trick 1
Arbitrary Sampling

Problem

Smooth and strongly convex

$$\min_{x \in \mathbb{R}^n} f(x)$$

Coordinate Descent with Arbitrary Sampling

i.i.d. subsets of [n] = {1, 2, ..., n}
(arbitrary distribution is allowed!)

Choose a random set S_t of coordinates

For $i \in S_t$ do

$$x_i^{t+1} \leftarrow x_i^t - \frac{1}{v_i}(\nabla f(x^t))^\top e_i$$

For $i \notin S_t$ do

$$x_i^{t+1} \leftarrow x_i^t$$

SYNC

[R & Takáč 13b]

Complexity Result

Theorem [R & Takáč 13b]

$$t \geq \left(\max_i \frac{v_i}{p_i \lambda}\right) \log\left(\frac{f(x^0) - f(x^*)}{\epsilon \rho}\right)$$

$p_i = \mathbf{P}(i \in S_t)$

strong convexity constant

$$\mathbf{P}\left(f(x^t) - f(x^*) \leq \epsilon\right) \geq 1 - \rho$$

Key Assumption

Parameters $v_1, \ldots, v_n$ satisfy:

$$\mathbf{E}\left[f\left(x + \sum_{i \in S_t} h_i e_i \right) \right] \leq f(x) + \sum_{i=1}^{n} p_i \nabla_i f(x) h_i + \sum_{i=1}^{n} p_i v_i h_i^2$$

Inequality must hold for all

$x, h \in \mathbb{R}^n$

$p_i = \mathbf{P}(i \in S_t)$

Proof

Theorem 3. *Let Assumptions 1 and 2 be satisfied. Choose* $x^0 \in \mathbf{R}^n$, $0 < \epsilon < \phi(x^0) - \phi^*$ *and* $0 < \rho < 1$, *where* $\phi^* := \min_x \phi(x)$. *Let*

$$\Lambda := \max_i \frac{w_i}{p_i v_i}. \tag{4}$$

If $\{x^k\}$ *are the random iterates generated by 'NSync, then*

$$K \geq \frac{\Lambda}{\gamma} \log\left(\frac{\phi(x^0)-\phi^*}{\epsilon\rho}\right) \quad \Rightarrow \quad \mathbf{Prob}(\phi(x^K) - \phi^* \leq \epsilon) \geq 1 - \rho. \tag{5}$$

Moreover, we have the lower bound $\Lambda \geq (\sum_i \frac{w_i}{v_i})/\mathbf{E}[|\hat{S}|]$.

Proof. We first claim that ϕ is μ-strongly convex with respect to the norm $\|\cdot\|_{w \bullet p^{-1}}$, i.e.,

$$\phi(x+h) \geq \phi(x) + \langle \nabla\phi(x), h\rangle + \frac{\mu}{2}\|h\|^2_{w\bullet p^{-1}}, \tag{6}$$

where $\mu := \gamma/\Lambda$. Indeed, this follows by comparing (3) and (6) in the light of (4). Let x^* be such that $\phi(x^*) = \phi^*$. Using (6) with $h = x^* - x$,

$$\phi^* - \phi(x) \overset{(6)}{\geq} \min_{h' \in \mathbf{R}^n} \langle \nabla\phi(x), h'\rangle + \frac{\mu}{2}\|h'\|^2_{w\bullet p^{-1}} = -\frac{1}{2\mu}\|\nabla\phi(x)\|^2_{p\bullet w^{-1}}. \tag{7}$$

Let $h^k := -(\mathrm{Diag}(w))^{-1}\nabla\phi(x^k)$. Then $x^{k+1} = x^k + (h^k)_{[\hat{S}]}$, and utilizing Assumption 1, we get

$$\mathbf{E}[\phi(x^{k+1}) \mid x^k] = \mathbf{E}[\phi(x^k + (h^k)_{[\hat{S}]})] \overset{(2)}{\leq} \phi(x^k) + \langle \nabla\phi(x^k), h^k\rangle_p + \frac{1}{2}\|h^k\|^2_{p\bullet w} \tag{8}$$

$$= \phi(x^k) - \frac{1}{2}\|\nabla\phi(x^k)\|^2_{p\bullet w^{-1}} \overset{(7)}{\leq} \phi(x^k) - \mu(\phi(x^k) - \phi^*). \tag{9}$$

Taking expectations in the last inequality and rearranging the terms, we obtain $\mathbf{E}[\phi(x^{k+1}) - \phi^*] \leq (1-\mu)\mathbf{E}[\phi(x^k) - \phi^*] \leq (1-\mu)^{k+1}(\phi(x^0) - \phi^*)$. Using this, Markov inequality, and the definition of K, we finally get $\mathbf{Prob}(\phi(x^K) - \phi^* \geq \epsilon) \leq \mathbf{E}[\phi(x^K) - \phi^*]/\epsilon \leq (1-\mu)^K(\phi(x^0) - \phi^*)/\epsilon \leq \rho$.

Let us now establish the last claim. First, note that (see [16, Sec 3.2] for more results of this type),

$$\sum_i p_i = \sum_i \sum_{S: i \in S} p_S = \sum_S \sum_{i: i \in S} p_S = \sum_S p_S |S| = \mathbf{E}[|\hat{S}|]. \tag{10}$$

Letting $\Delta := \{p' \in \mathbf{R}^n : p' \geq 0, \sum_i p'_i = \mathbf{E}[|\hat{S}|]\}$, we have

$$\Lambda \overset{(4)+(10)}{\geq} \min_{p' \in \Delta} \max_i \frac{w_i}{p'_i v_i} = \frac{1}{\mathbf{E}[|\hat{S}|]} \sum_i \frac{v_i}{w_i},$$

where the last equality follows since optimal p'_i is proportional to v_i/w_i. ☐

Copy-paste from the paper

How to compute the parameters?

Theorem [Qu & R 14a]

$M_j : \mathbb{R}^n \to \mathbb{R}^m$

γ_j-smooth

$$f(x) = \sum_j \phi_j(M_j x)$$

$$A^\top A = \sum_j \gamma_j M_j^\top M_j$$

Theorem [Qu & R 14a]

The assumption holds if for some matrix *A*, *f* satisfies

$$f(x+h) \le f(x) + \nabla f(x)^\top h + \frac{1}{2} h^\top A^\top A h$$

and *v* satisfies

$$P \circ A^\top A \preceq \mathrm{Diag}(p \circ v)$$

$P_{ij} = \mathbf{P}(\{i,j\} \subseteq S_t)$

Hadamard (element-wise) product

[Qu & R 14a] give formulas for *v* as a function of the data matrix *A* and sampling S_t

Insight 1: Importance Sampling Helps

$$\mathbf{P}(|S_t| = 1) = 1 \quad \Rightarrow \quad v = \mathrm{Diag}(A^\top A)$$

- If we update a single coordinate in each iteration, *P* is diagonal, and we get a simple formula for *v* (independent of the probability vector *p*)
- In particular, we can choose *p* which optimizes the complexity, which leads to importance sampling:

Importance sampling: $p_i = \frac{v_i}{\sum_i v_i} \quad \Rightarrow \quad \max_i \frac{v_i}{p_i \lambda} = \frac{\sum_i v_i}{\lambda}$

Uniform sampling: $p_i = \frac{1}{n} \quad \Rightarrow \quad \max_i \frac{v_i}{p_i \lambda} = \frac{n \max_i v_i}{\lambda}$

Average can be much smaller than max !

Bibliographic Remarks

- [Leventhal & Lewis 08] were first to study randomized CD methods (for linear systems & least squares). Moreover, they proposed nonuniform probabilities.
  - Convenient; not optimal
  - Optimal probabilities for linear systems can be computed via SDP: [Gower & R 15]
- [Nesterov 10] considered probabilities proportional to powers of coordinate-wise Lipschitz constants (for smooth convex minimization)
  - Not interpreted as optimal
- [R & Takáč 11b] gave complexity results for an arbitrary probability vector p
- [R & Takáč 13b] introduced arbitrary sampling (NSync)
  - Importance sampling as a corollary
  - Also studied importance sampling over subsets of coordinates (leads to LP)
- [Zhao & Zhang 14] studied stochastic optimization (I-Prox SGD and I-Prox SDCA) with importance sampling

Further Bibliographic Remarks

- [Qu, R & Zhang 14] were first to study ERM with arbitrary sampling (Quartz)
- [Qu & R 14a] studied standard and accelerated methods for convex composite problems with arbitrary sampling (ALPHA)
- [Csiba & R 15] extended the dual-free analysis of SDCA [S-Shwartz 15] to arbitrary sampling (dfSDCA)
  - analysis works also for non-convex loss functions as long as the average loss is convex
- [Konečný, Qu & R 14] studied a semi-stochastic coordinate descent method (S2CD) utilizing importance sampling

Insight 2: CD is faster than GD

$$S_t \equiv [n] \quad \Rightarrow \quad v_i = \lambda_{\max}(A^\top A)$$

$$p_i = 1$$

Standard condition number

Gradient Descent =
CD with deterministic sampling:

$$\frac{\lambda_{\max}(A^\top A)}{\lambda}$$

CD with importance sampling:

$$\frac{\mathrm{Tr}(A^\top A)}{\lambda}$$

1 iteration of CD is often *n* times cheaper than 1 iteration of GD. However, complexity of CD can be as good as complexity of GD, and is always at most *n* times as bad. So, CD is better.

Insight 3: Speedup and Flexibility

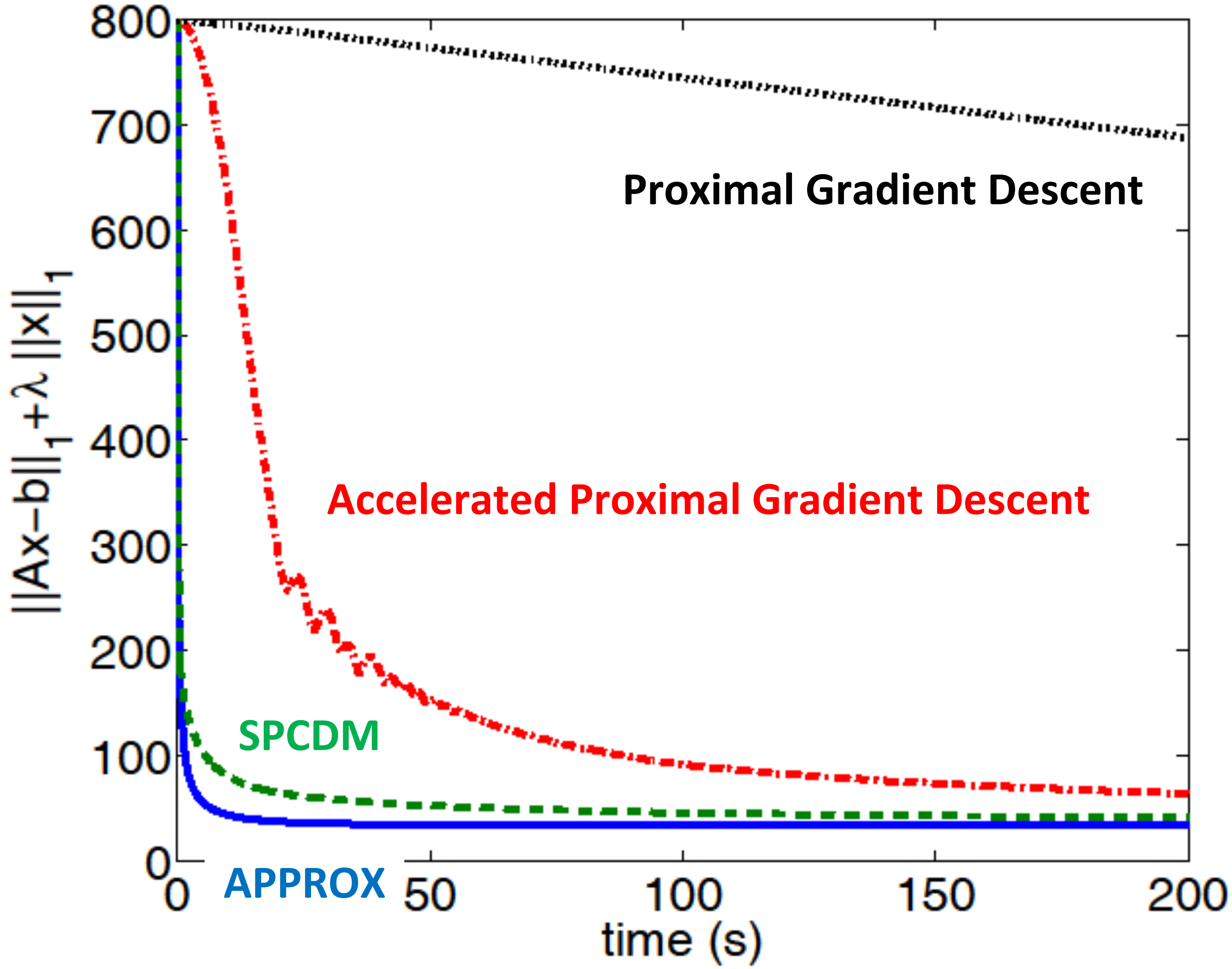
- **Speedup.** Complexity improves with the size of the mini-batch $|S_t|$, but less than linearly
  - The amount of speedup depends on
    - data sparsity [R & Takáč 12], [Fercoq & R 13b], [Qu & R 14b]
    - spectral properties of the data [Bradley et al 11], [Takáč et al 13], [R & Takáč 13a], [Fercoq et al 14], [Qu & R 14b]
  - Hence mini-batching helps if there are gains from parallelism or reduction of memory transfers
- **Flexibility.** Sometimes we may be forced to sample in a certain way (e.g., distributed implementation)
  - Results with arbitrary sampling say it's OK to sample as we like

Trick 2
Acceleration

α

Zheng Qu and P.R.
Coordinate descent with arbitrary sampling I: algorithms and complexity *arXiv:1412.8060,* 2014

L1 Regularized L1 Regression



A P PROX

[Fercoq & R 13b]

Dorothea dataset: $N = 100,000$ $m = 800$ $\omega = 6,061$

Problem

Smooth & convex

Convex

$$\min_{x \in \mathbb{R}^n} f(x) + \sum_{i=1}^{n} \psi_i(x_i)$$

ALPHA (for smooth minimization)

STEP 0: $z^0 = x^0$

STEP 1: $y^t \leftarrow (1-\theta_t)x^t + \theta_t z^t$

STEP 2: For $i \in S_t$

$$z_i^{t+1} \leftarrow z_i^t - \frac{p_i}{v_i \theta_t} \nabla_i f(y^t)$$

For $i \notin S_t$

$$z_i^{t+1} \leftarrow z_i^t$$

STEP 3: $x^{t+1} \leftarrow y^t + \theta_t Diag^{-1}(p)(z^{t+1} - z^t)$

i.i.d. random subsets of coordinates (any distribution allowed)

Same as in NSync

Complexity Theorem

$$\theta_0 = 1, \quad \theta_{t+1} = \frac{\sqrt{\theta_t^4 + 4\theta_t^2} - \theta_t^2}{2}$$

$$\mathbf{E}[f(x^t)] - f(y) \leq \frac{2\sum_{i=1}^n (x_i^0 - y_i)^2 \frac{v_i}{p_i^2}}{(t+1)^2}$$

Same as in NSync

Arbitrary point

$p_i = \mathbf{P}(i \in \hat{S})$

Insights

- **The result makes sense:** If a coordinate is optimal – do not update it!
- **Unification:**
  - Stochastic (CD, ACD) and deterministic (GD, AGD) methods
  - Single analysis recovers the best bounds

Bibliographic Remarks

- UCDM, RCDM, ACDM [Nesterov 10]
  - First combination of acceleration & randomized coordinate descent
  - Inefficient in both theory and practice
- ASDCA [S-Shwartz & Zhang 13a]
  - Interpolates between SDCA and Accelerated Gradient Descent
- Acc Prox-SDCA [S-Shwartz & Zhang 13b]
- APPROX [Fercoq & R 13b]
  - Efficient version of accelerated coordinate descent
  - Arbitrary uniform sampling
  - Incorporates accelerated coordinate descent & accelerated gradient descent as special cases
- APCG [Lin, Lu & Xiao 14]
  - Extension of APPROX to strongly convex functions & application to ERM
- SPDC [Zhang & Xiao 14]
  - Mini-batching, importance sampling, designed for ERM
- ALPHA [Qu & R 14a]
  - Extension of APPROX to arbitrary samplings
  - Unified analysis of non-accelerated and accelerated methods

Trick 3
Duality

Zheng Qu, P.R. and Tong Zhang
Randomized dual coordinate ascent with arbitrary sampling
arXiv:1411.5873, 2014

EMPIRICAL RISK MINIMIZATION

Primal Problem: ERM

$$\min_{w\in\mathbb{R}^d} \left[P(w) \equiv \frac{1}{n}\sum_{i=1}^{n} \phi_i(A_i^\top w) + \lambda g(w) \right]$$

- $1/\gamma$ - smooth & convex
- regularization parameter
- *d* = # features (parameters)
- *n* = # samples
- 1 - strongly convex function (regularizer)

Assumption 1

Loss functions have Lipschitz gradient

$$\|\nabla \phi_i(a) - \nabla \phi_i(a')\| \leq \frac{1}{\gamma} \|a - a'\|, \quad a, a' \in \mathbb{R}^m$$

Lipschitz constant

Assumption 2

Regularizer is 1-strongly convex

$$g(w) \geq g(w') + \langle \nabla g(w'), w - w' \rangle + \frac{1}{2}\|w - w'\|^2, \quad w, w' \in \mathbb{R}^d$$

subgradient

Dual Problem

$\in \mathbb{R}^m$

$$D(\alpha) \equiv -\lambda g^* \left(\frac{1}{\lambda n} \sum_{i=1}^{n} A_i \alpha_i \right) - \frac{1}{n} \sum_{i=1}^{n} \phi_i^*(-\alpha_i)$$

1 – smooth
& convex

$\in \mathbb{R}^d$

γ - strongly convex

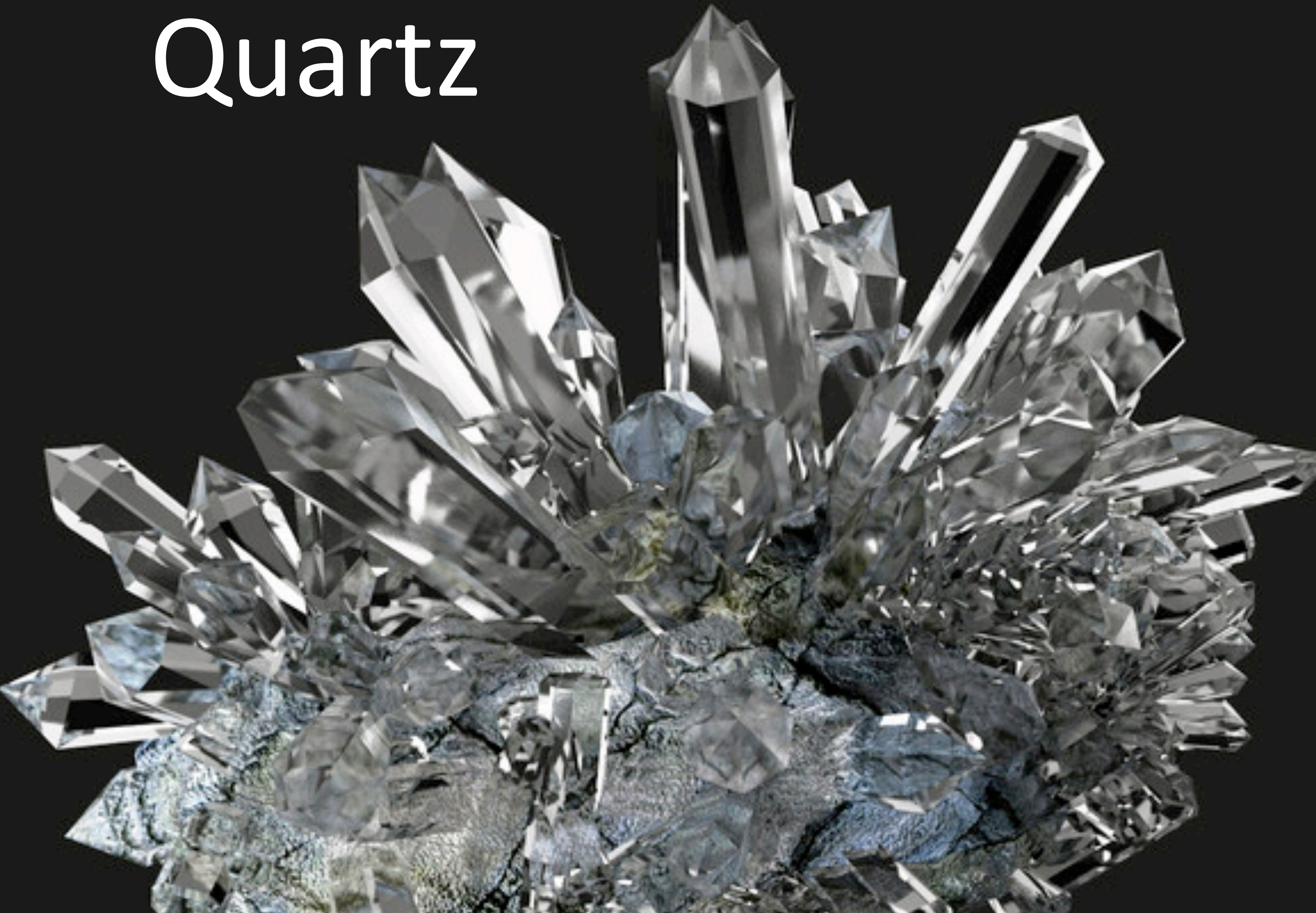
$$g^*(w') = \max_{w \in \mathbb{R}^d} \{(w')^\top w - g(w)\}$$

$$\phi_i^*(a') = \max_{a \in \mathbb{R}^m} \{(a')^\top a - \phi_i(a)\}$$

$$\max_{\alpha=(\alpha_1,\ldots,\alpha_n)\in\mathbb{R}^N=\mathbb{R}^{nm}} D(\alpha)$$

$\in \mathbb{R}^m$ $\in \mathbb{R}^m$

Quartz



Fenchel Duality

$$\bar{\alpha} = \frac{1}{\lambda n} \sum_{i=1}^{n} A_i \alpha_i$$

$$P(w) - D(\alpha) \;=\; \lambda\left(g(w) + g^*\left(\bar{\alpha}\right)\right) + \frac{1}{n}\sum_{i=1}^{n} \phi_i(A_i^\top w) + \phi_i^*(-\alpha_i) =$$

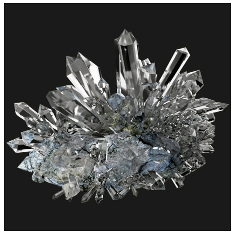
$$\underbrace{\lambda(g(w) + g^*\left(\bar{\alpha}\right) - \langle w, \bar{\alpha}\rangle)}_{\geq 0} + \frac{1}{n}\sum_{i=1}^{n} \underbrace{\phi_i(A_i^\top w) + \phi_i^*(-\alpha_i) + \left\langle A_i^\top w, \alpha_i\right\rangle}_{\geq 0}$$

≥ 0 ← **Weak duality** → ≥ 0

Optimality conditions

$$w = \nabla g^*(\bar{\alpha}) \qquad\qquad \alpha_i = -\nabla \phi_i(A_i^\top w)$$

The Algorithm



$$(\alpha^t, w^t) \quad \Rightarrow \quad (\alpha^{t+1}, w^{t+1})$$

Quartz: Bird's Eye View

STEP 1: PRIMAL UPDATE

$$w^{t+1} \leftarrow (1-\theta)w^t + \theta \nabla g^*(\bar{\alpha}^t)$$

STEP 2: DUAL UPDATE

Choose a random set S_t of dual variables

$p_i = \mathbf{P}(i \in S_t)$

For $i \in S_t$ do

$$\alpha_i^{t+1} \leftarrow \left(1 - \frac{\theta}{p_i}\right)\alpha_i^t + \frac{\theta}{p_i}\left(-\nabla \phi_i(A_i^\top w^{t+1})\right)$$

Randomized Dual Coordinate Ascent Methods for ERM

Algorithm	1-nice	1-optimal	τ-nice	arbitrary	additional speedup	direct p-d analysis	acceleration
SDCA	●						
mSDCA	●		●		●		
ASDCA	●		●				●
AccProx-SDCA	●						●
DisDCA	●		●				
Iprox-SDCA	●	●					
APCG	●						●
SPDC	●	●	●			●	●
Quartz	●	●	●	●	●	●	

SDCA: SS Shwartz & T Zhang, 09/2012
mSDCA M Takac, A Bijral, P R & N Srebro, 03/2013
ASDCA: SS Shwartz & T Zhang, 05/2013
AccProx-SDCA: SS Shwartz & T Zhang, 10/2013
DisDCA: T Yang, 2013
Iprox-SDCA: P Zhao & T Zhang, 01/2014
APCG: Q Lin, Z Lu & L Xiao, 07/2014
SPDC: Y Zhang & L Xiao, 09/2014
Quartz: Z Qu, P R & T Zhang, 11/2014

COMPLEXITY

Assumption 3
(Expected Separable Overapproximation)

Parameters $v_1, \dots, v_n$ satisfy:

$$\mathbf{E} \left\| \sum_{i \in \hat{S}} A_i \alpha_i \right\|^2 \leq \sum_{i=1}^{n} p_i v_i \|\alpha_i\|^2$$

inequality must hold for all

$\alpha_1, \dots, \alpha_n \in \mathbb{R}^m$

$p_i = \mathbf{P}(i \in \hat{S})$

Complexity

$$\theta = \min_i \frac{p_i \lambda \gamma n}{v_i + \lambda \gamma n}$$

Theorem [Qu, R & Zhang 14]

$$\mathbf{E}[P(w^t) - D(\alpha^t)] \leq (1-\theta)^t (P(w^0) - D(\alpha^0))$$

$$t \geq \max_i \left(\frac{1}{p_i} + \frac{v_i}{p_i \lambda \gamma n} \right) \log \left(\frac{P(w^0) - D(\alpha^0)}{\epsilon} \right)$$

$$\Rightarrow \quad \mathbf{E}\left[P(w^t) - D(\alpha^t)\right] \leq \epsilon$$

Example

Data: $n = 7 \times 10^5$

$\gamma = \frac{1}{4} \quad v_i \equiv \lambda_{\max}(A_i^\top A_i) \leq 1$

Method: $|S_t| \equiv 1 \quad p_i = \frac{1}{n} \quad \lambda = \frac{1}{n}$

$$(1-\theta)^n = 0.8187$$

$$(1-\theta)^{12n} = 0.0907 < \frac{1}{10}$$

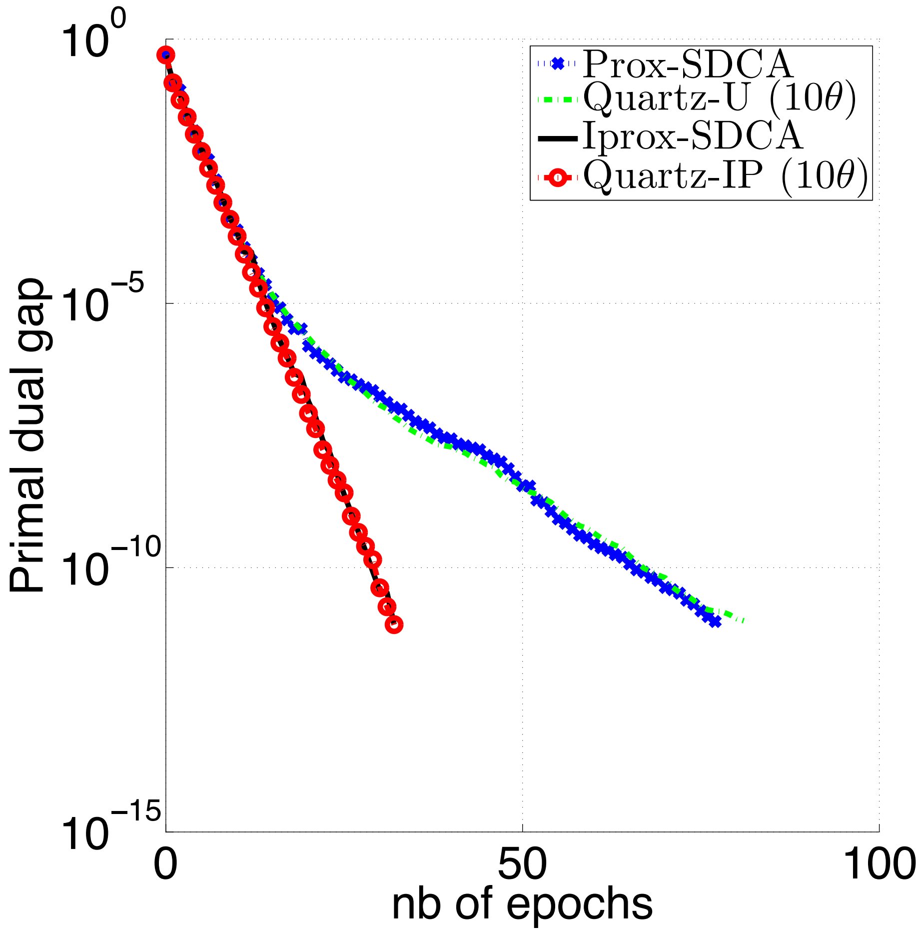
UPDATING 1 DUAL VARIABLE AT A TIME

Complexity of Quartz specialized to serial sampling

Optimal sampling	$n + \frac{\frac{1}{n}\sum_{i=1}^{n} L_i}{\lambda\gamma}$
Uniform sampling	$n + \frac{\max_i L_i}{\lambda\gamma}$

$$L_i \equiv \lambda_{\max}\left(A_i^\top A_i\right)$$

Experiment: Quartz vs SDCA, uniform vs optimal sampling



$$\text{Data} = \texttt{cov1}, \quad n = 522,911, \quad \lambda = 10^{-6}$$

Trick 4
Curvature

SDNA

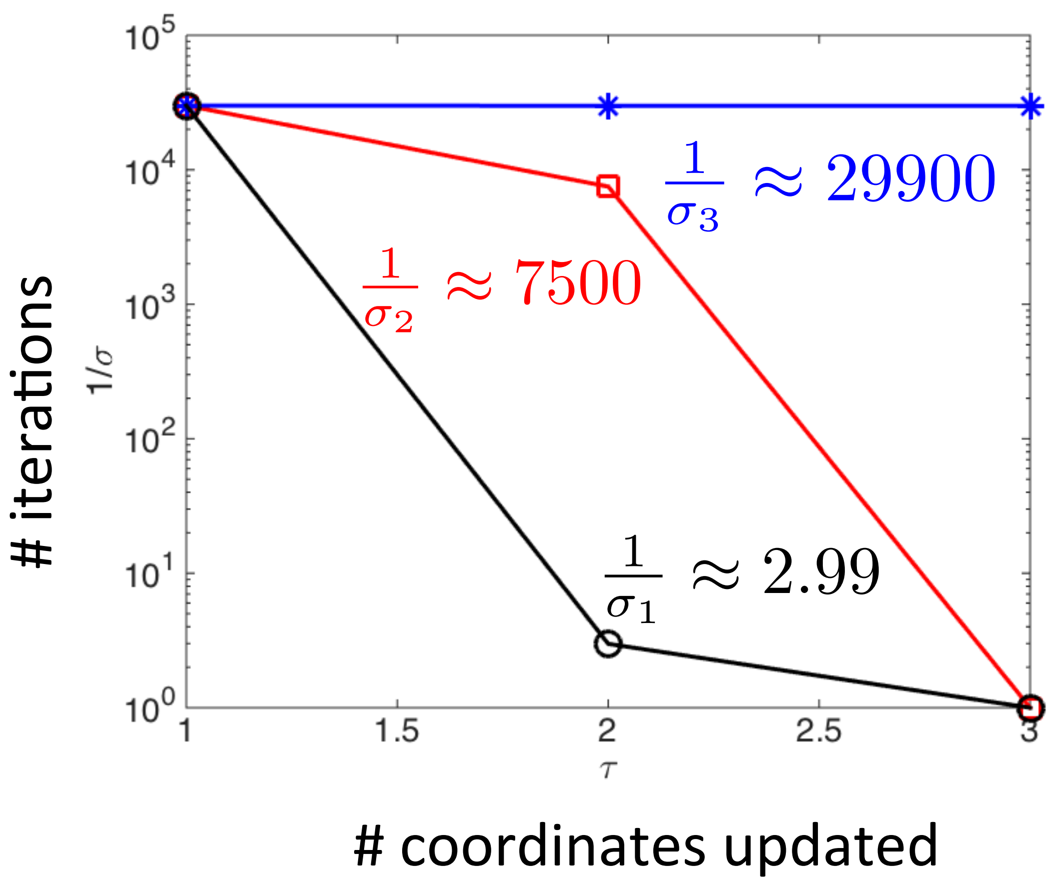
Zheng Qu, P.R., Martin Takáč and Olivier Fercoq
SDNA: Stochastic Dual Newton Ascent for empirical risk minimization
arXiv:1502.02268, 2015

The Power of Curvature

$$\min_{x\in\mathbb{R}^3}\left[f(x) = \frac{1}{2}x^T\mathbf{M}x + b^\top x + c\right]$$

$$\mathbf{M} = \begin{pmatrix} 1.0000 & 0.9900 & 0.9999 \\ 0.9900 & 1.0000 & 0.9900 \\ 0.9999 & 0.9900 & 1.0000 \end{pmatrix}$$

$$\text{condition number} \approx 3 \times 10^4$$



- Phenom. described in [Qu et al 15]
- Two points of view: “Exact line search in higher dimensional subspaces” or “inversion of random submatrices of the Hessian”
- Applied to ERM dual: **SDNA** (Stochastic Dual Newton Ascent)

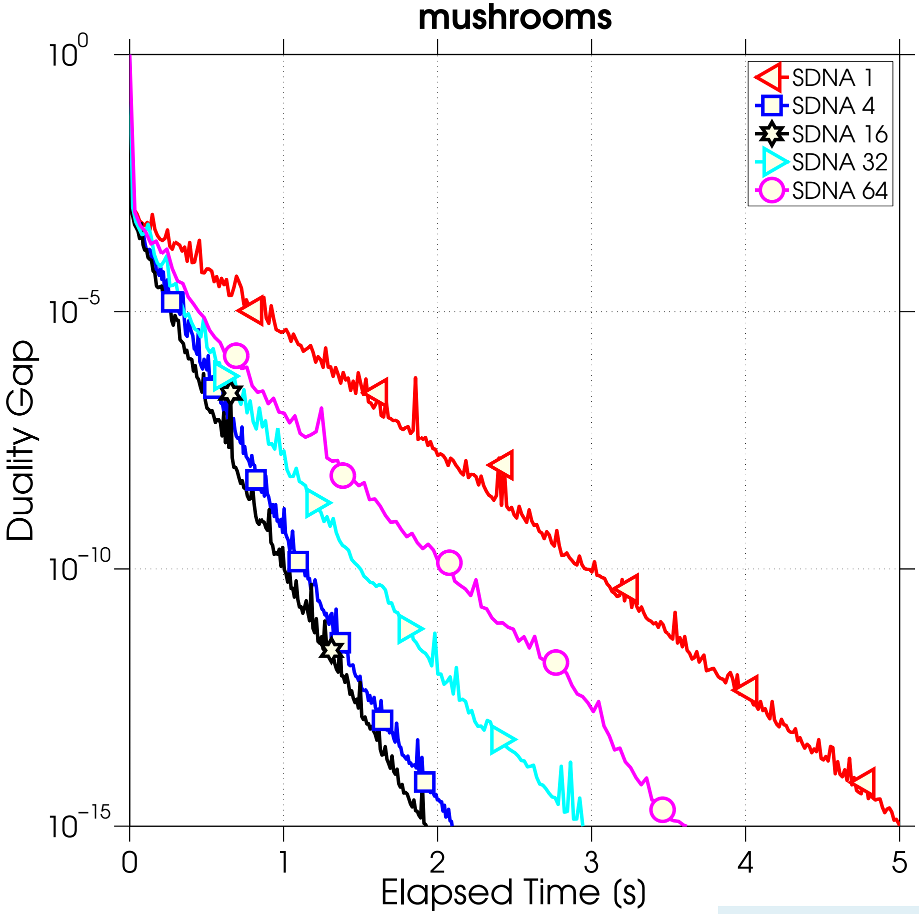
Real Dataset:

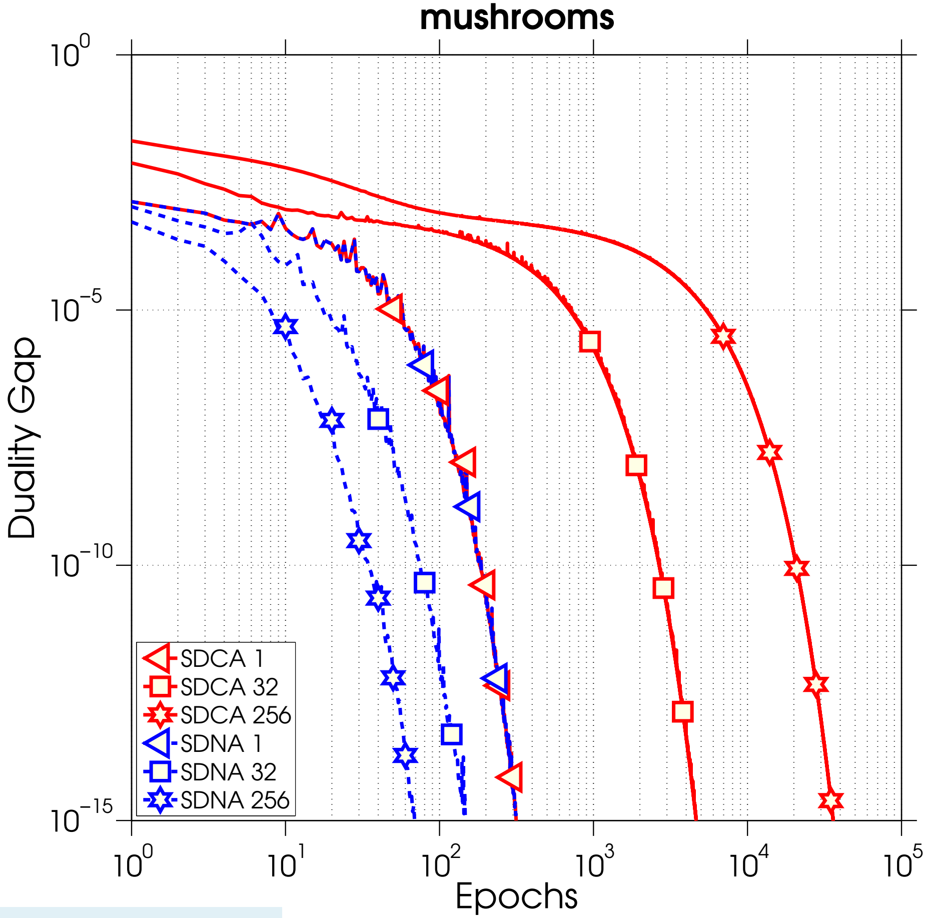
`mushrooms`

d = 112 n = 8,124

SDNA

Sampling “Smallish” Submatrices of the Hessian Helps





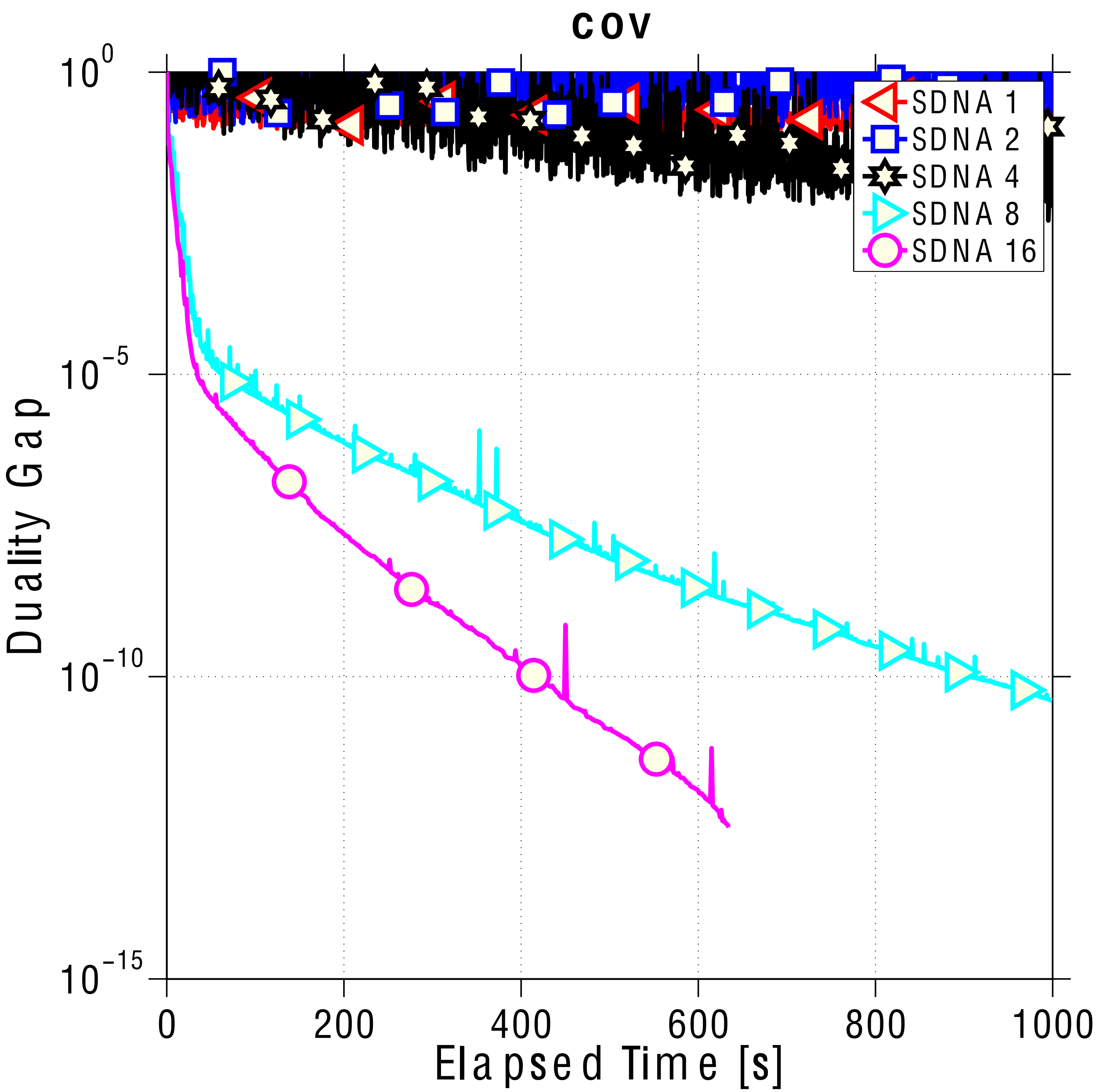
features: d = 112
examples: n = 8124

Real Dataset:

cov

$d = 54 \quad n = 581{,}012$

SDNA

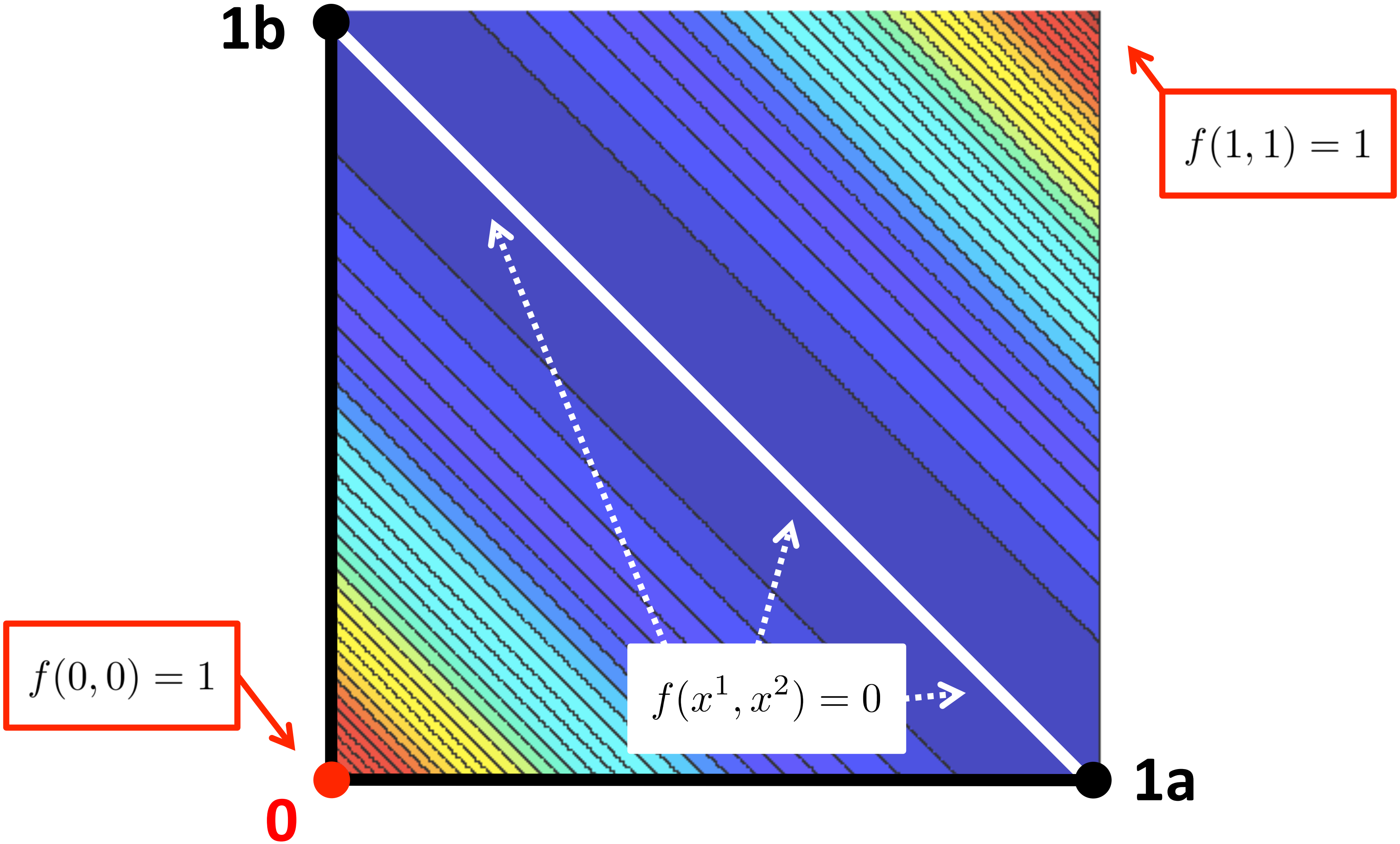

cov
Duality Gap
Elapsed Time [s]
SDNA 1
SDNA 2
SDNA 4
SDNA 8
SDNA 16
10^{0}
10^{-5}
10^{-10}
10^{-15}
0
200
400
600
800
1000

Trick 5
Parallelization
& Minibatching

NAIVE APPROACH

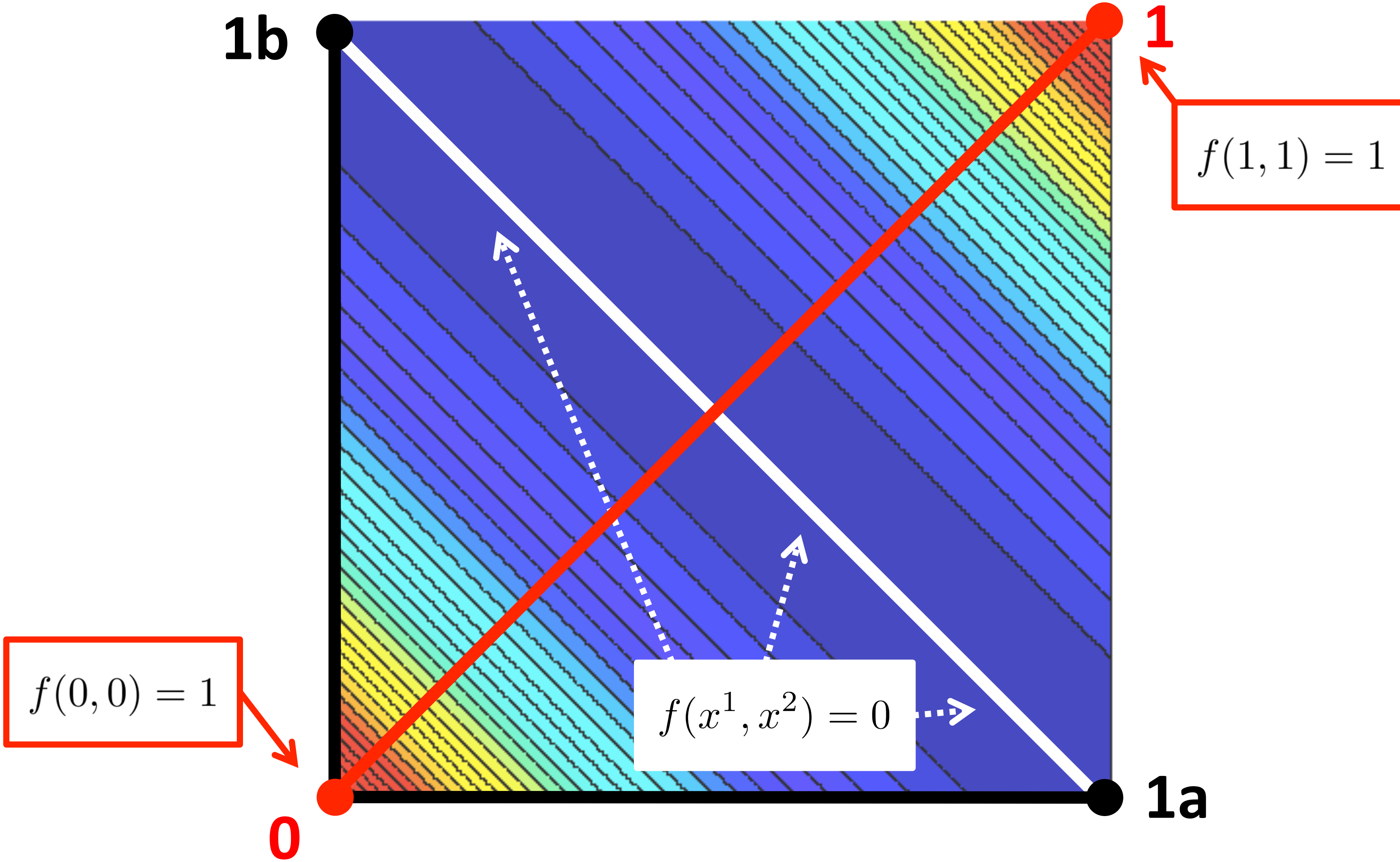
Failure of naive parallelization

$$f(x^1, x^2) = (x^1 + x^2 - 1)^2$$



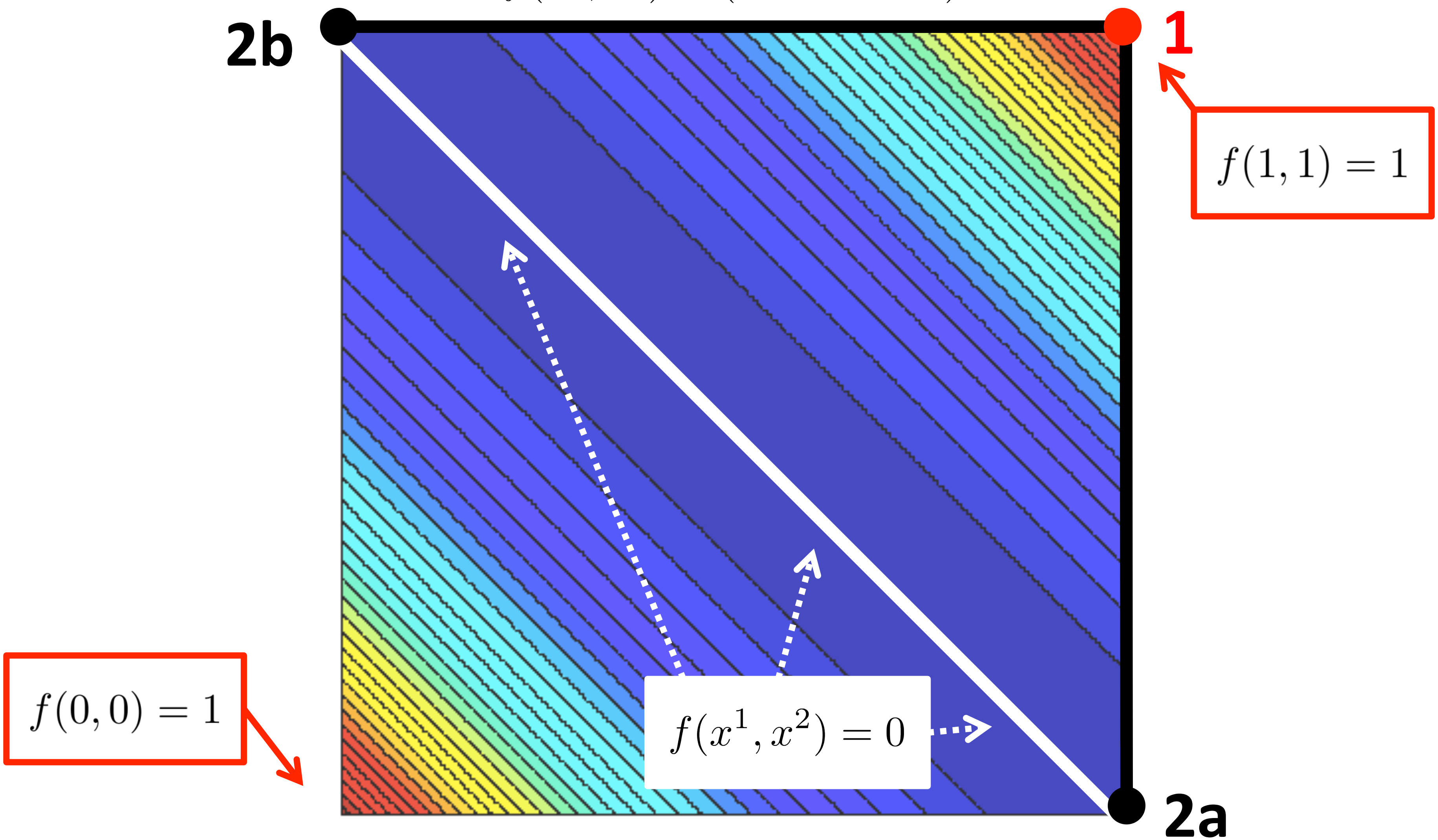
Failure of naive parallelization

$$f(x^1, x^2) = (x^1 + x^2 - 1)^2$$

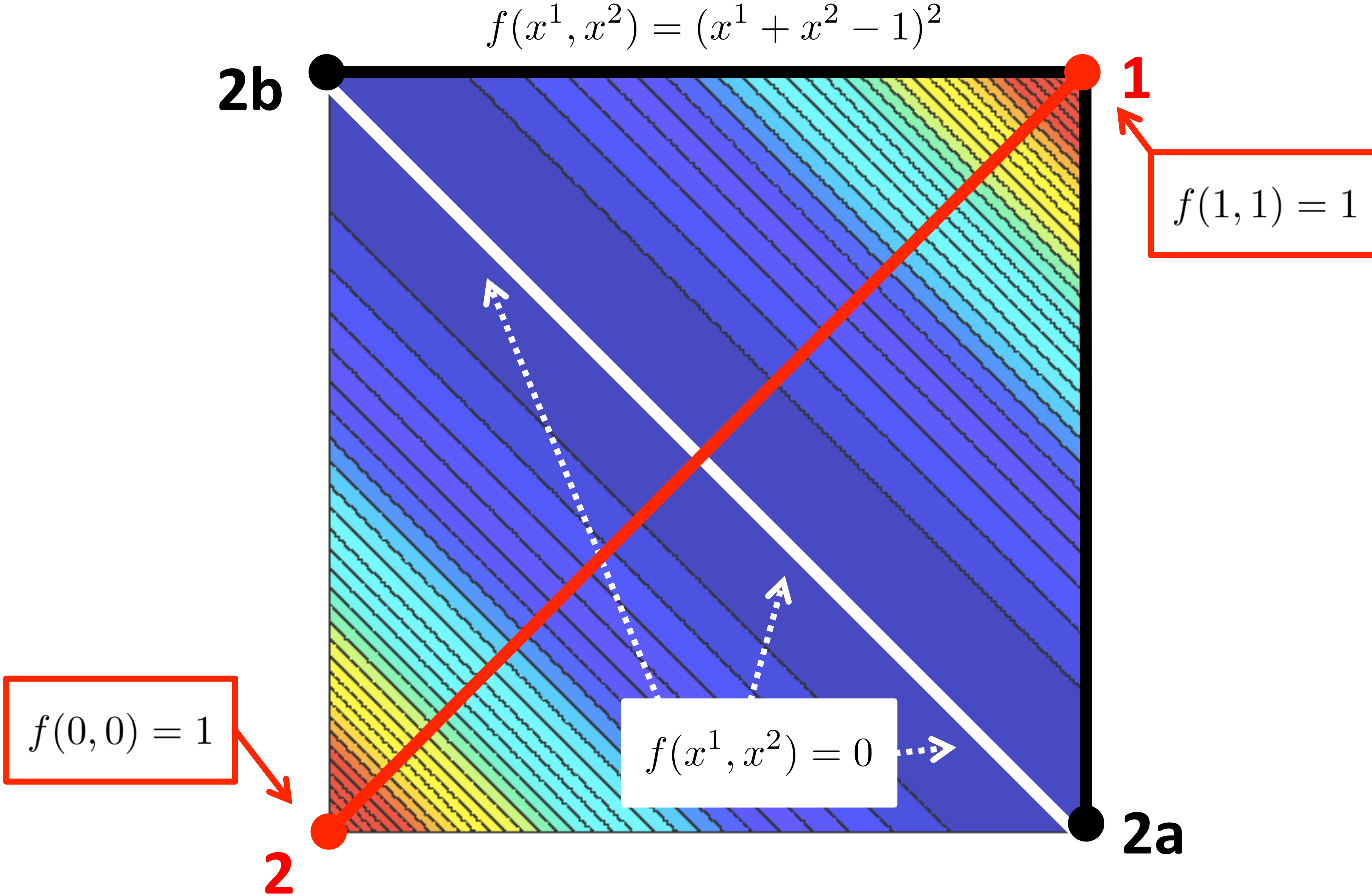


Failure of naive parallelization

$$f(x^1, x^2) = (x^1 + x^2 - 1)^2$$

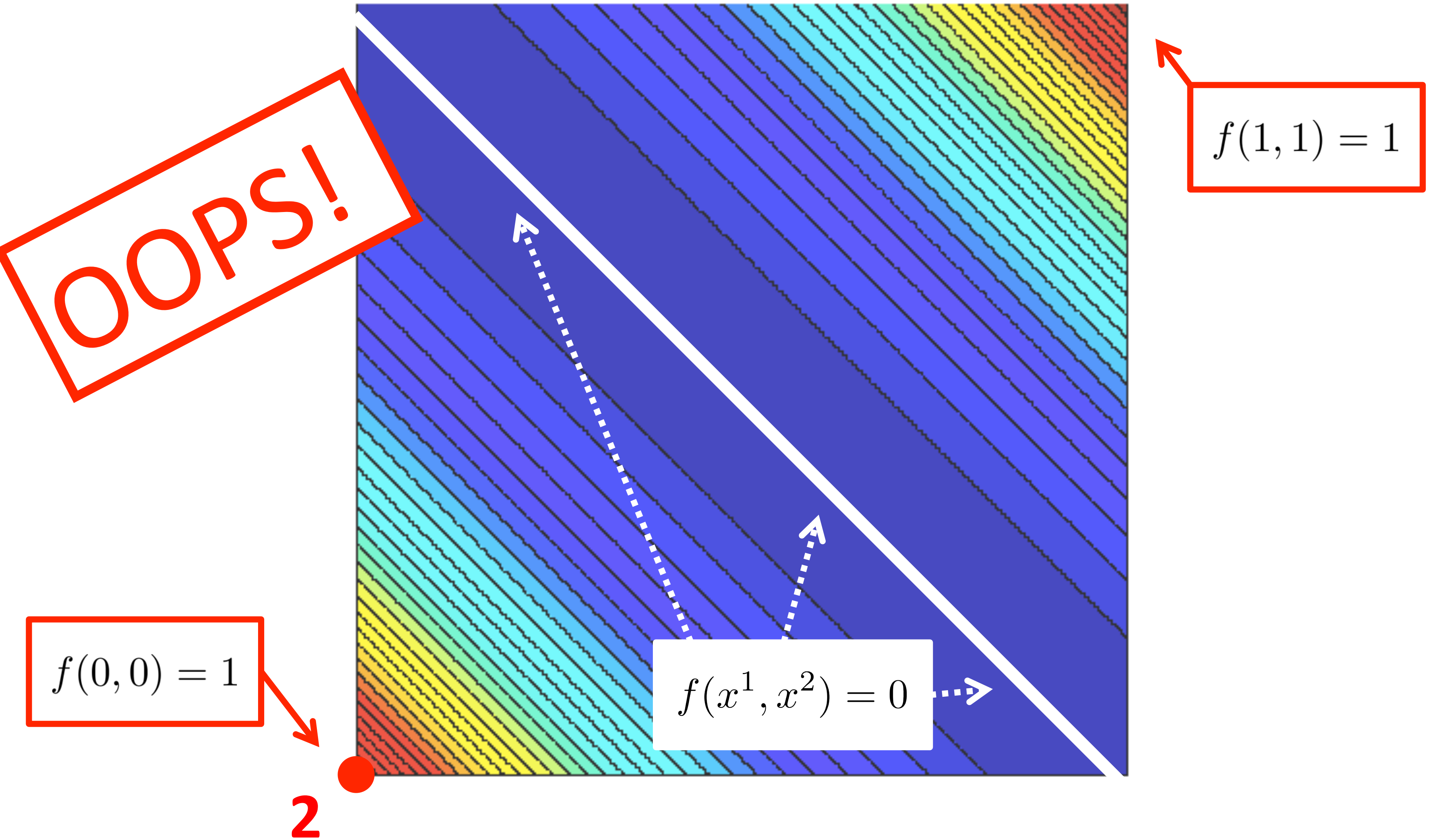


Failure of naive parallelization

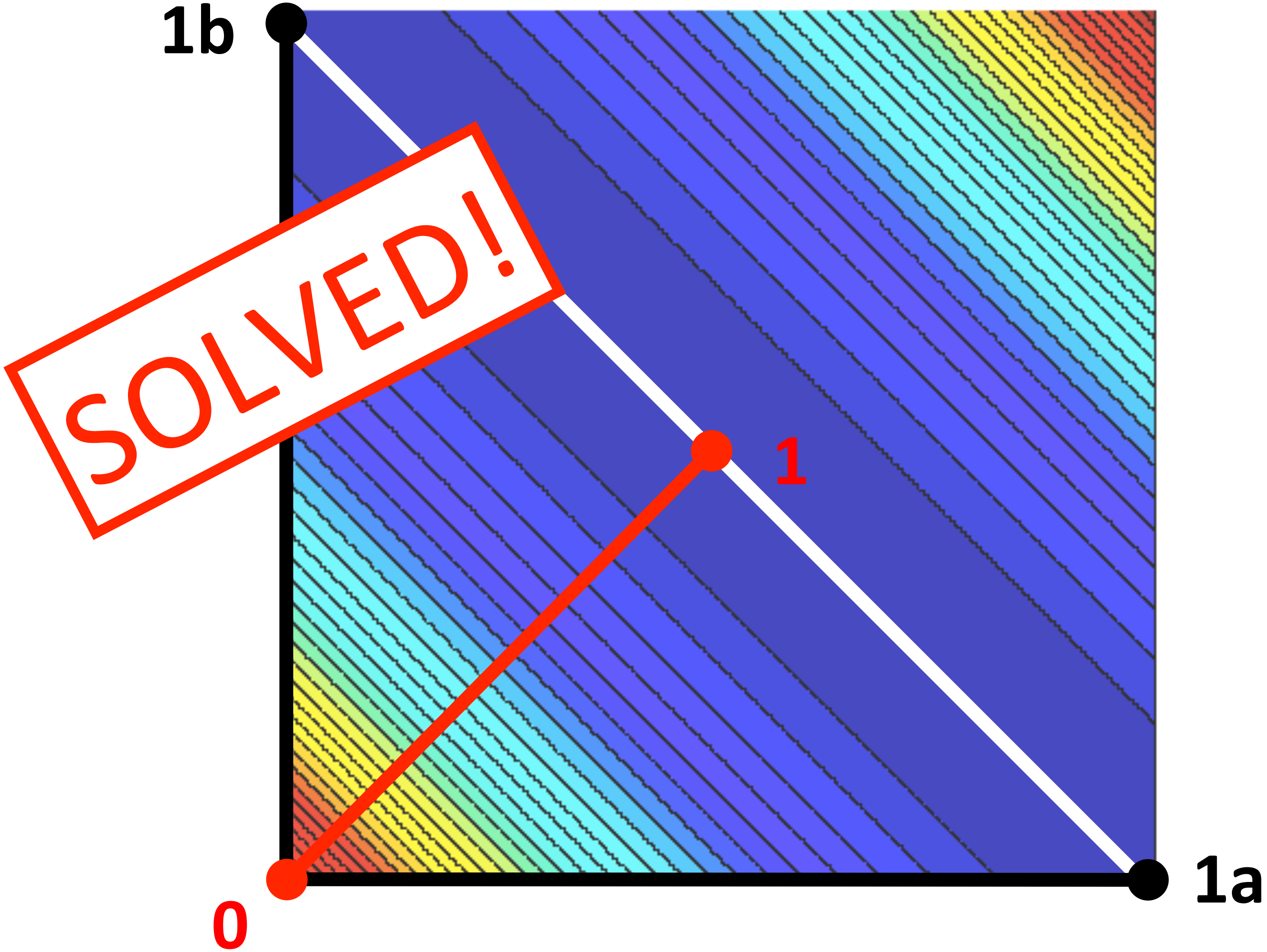


Failure of naive parallelization

$$f(x^1, x^2) = (x^1 + x^2 - 1)^2$$

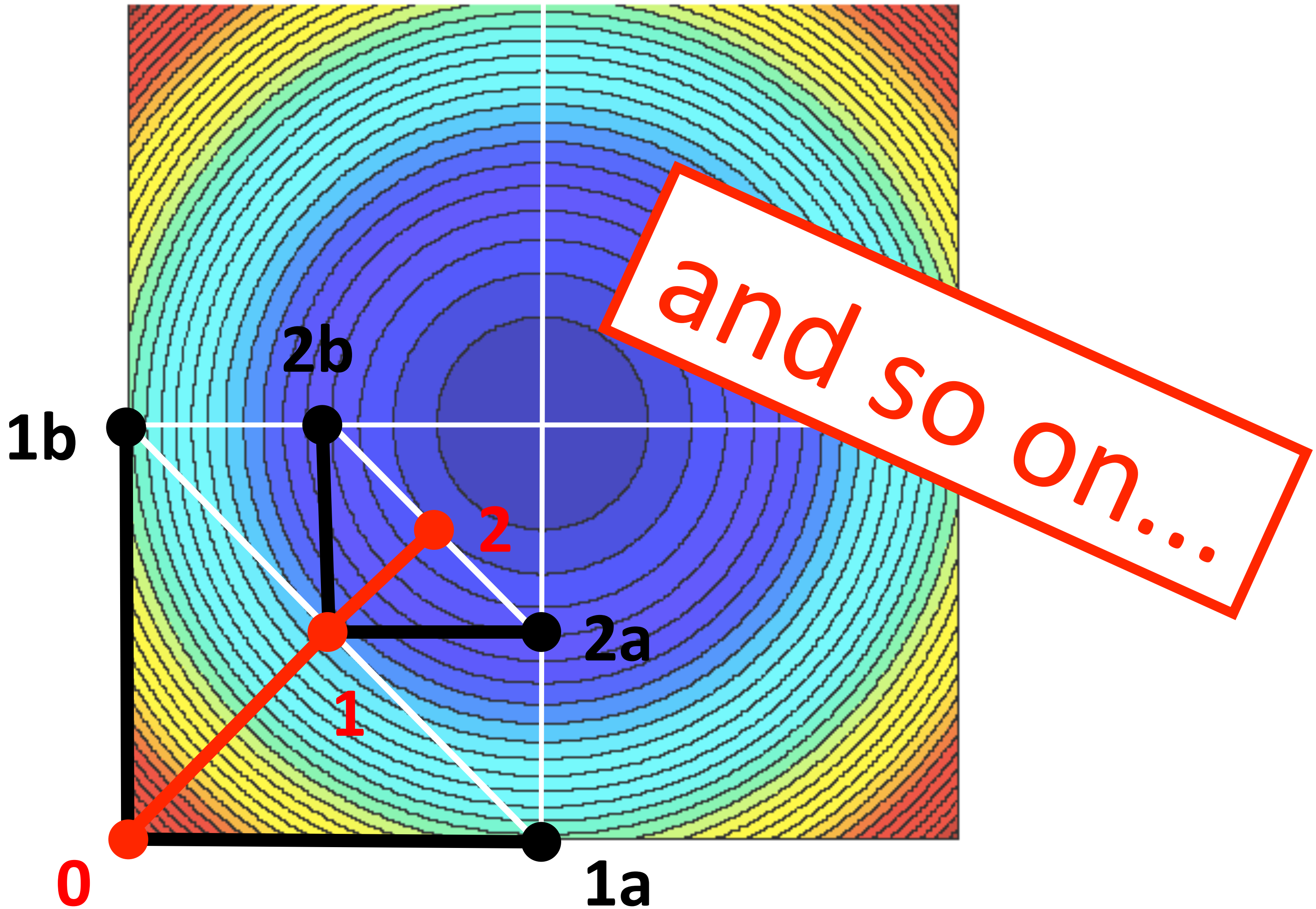


Idea: averaging updates may help



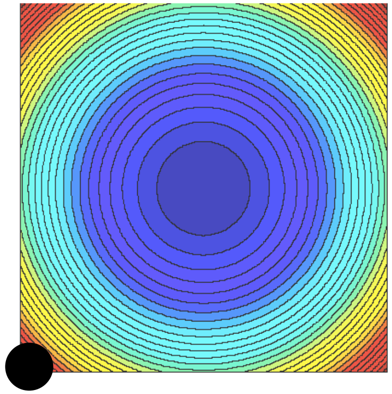
Averaging can be too conservative

$$f(x^1, x^2) = (x^1 - 1)^2 + (x^2 - 1)^2$$



Averaging may be too conservative

$$f(x) = (x^1 - 1)^2 + (x^2 - 1)^2 + \cdots + (x^n - 1)^2$$



$x_0 = 0 \quad f(x_0) = n$

BAD!!!

$$k \geq \frac{n}{2} \log\left(\frac{n}{\epsilon}\right)$$

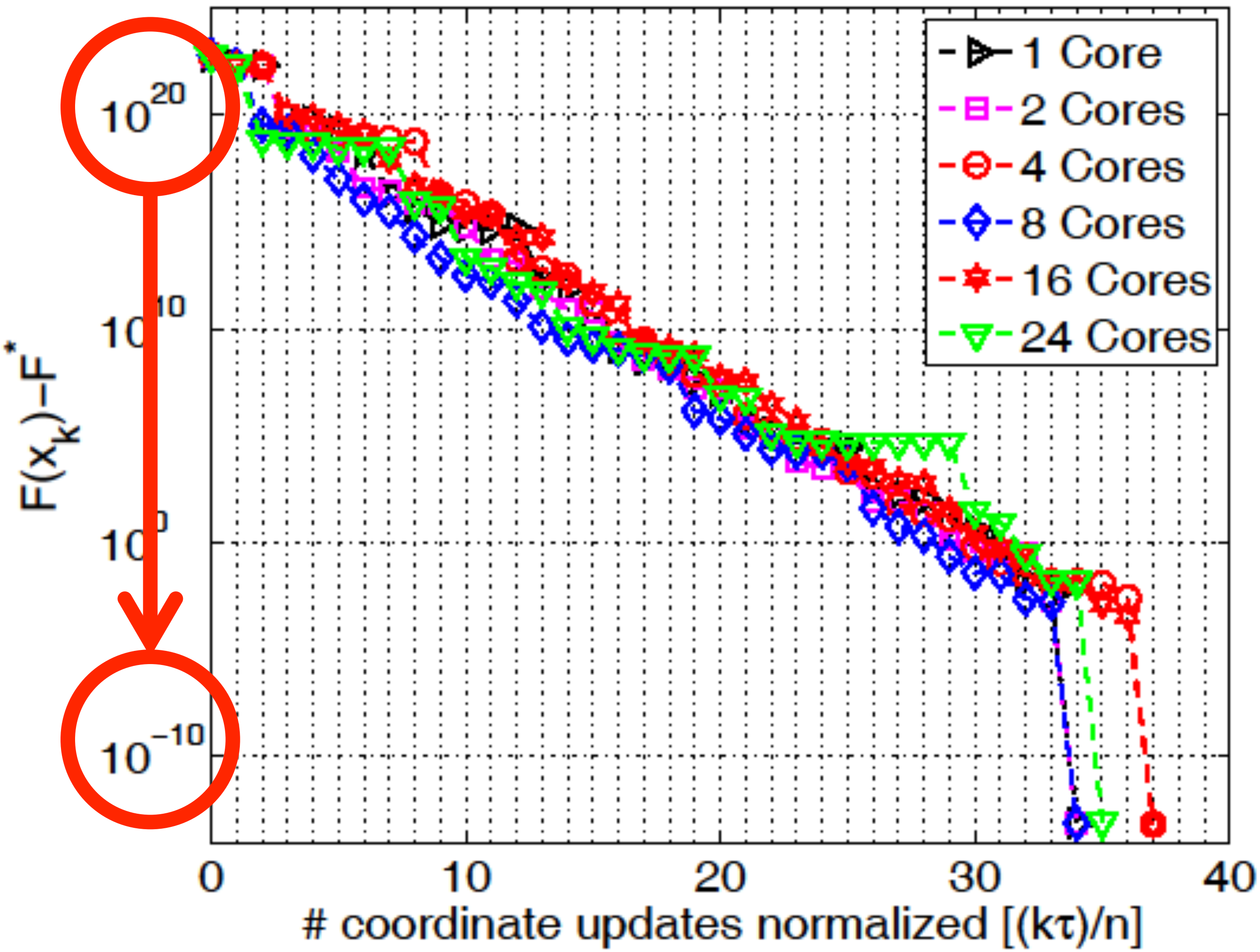
$$f(x_k) = n\left(1 - \frac{1}{n}\right)^{2k} \leq \epsilon$$

WANT

Experiment with a 1 billion-by-2 billion LASSO problem

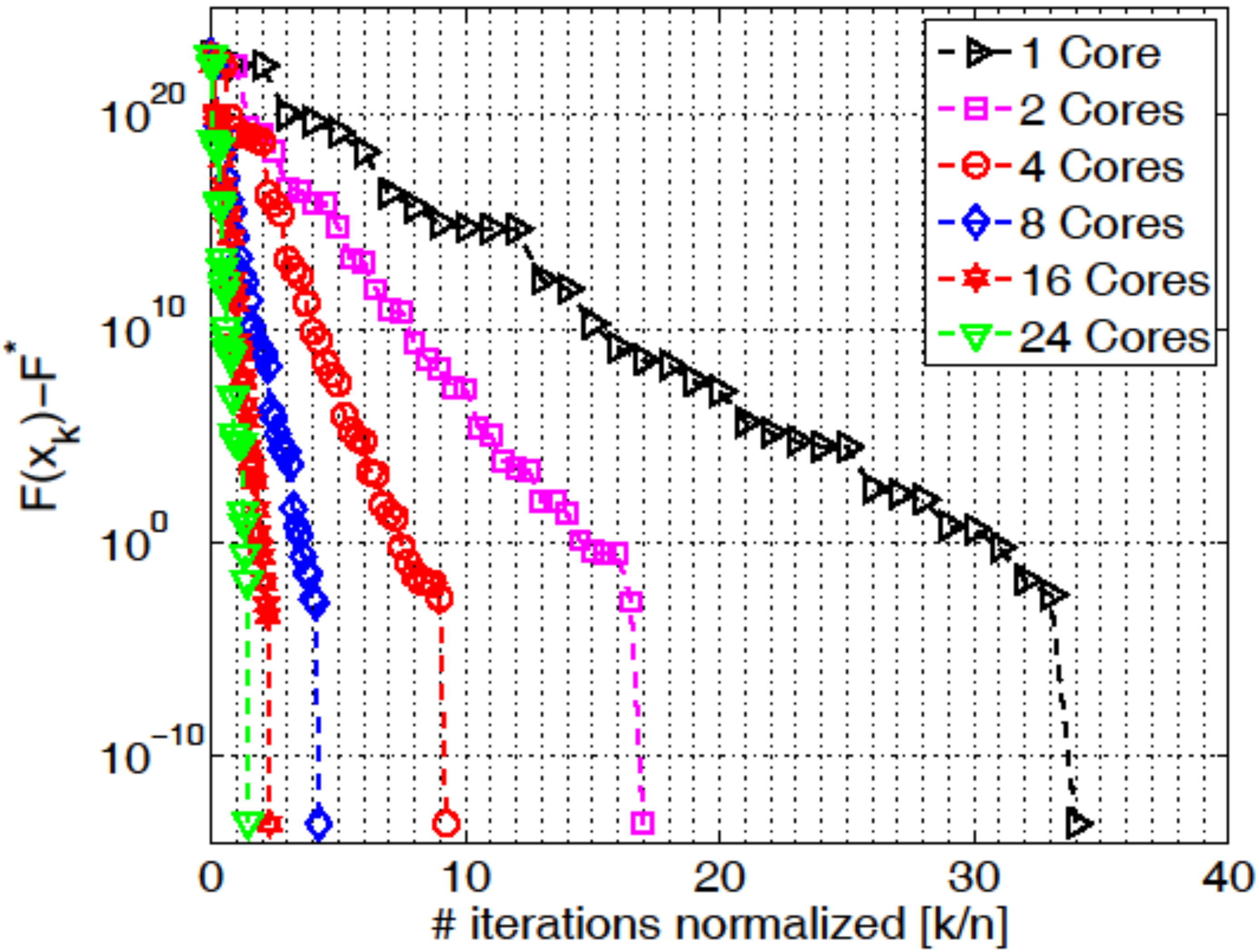
[R & Takáč 12]

Coordinate Updates



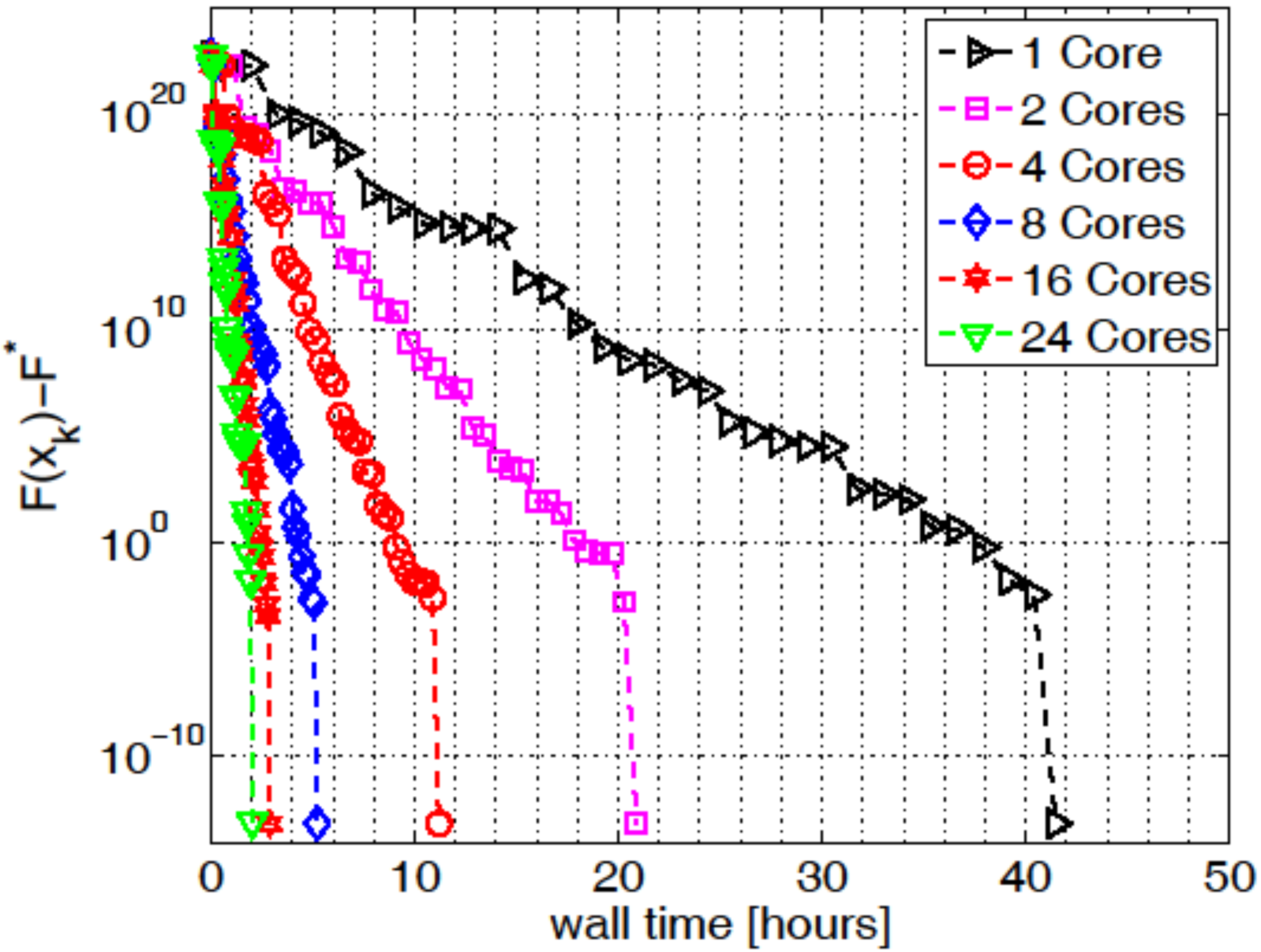
LASSO problem with $A \in \mathbb{R}^{m \times n}$, where $n = 10^9$ and $m = 2 \times 10^9$

Iterations



LASSO problem with $A \in \mathbb{R}^{m \times n}$, where $n = 10^9$ and $m = 2 \times 10^9$

Wall Time



LASSO problem with $A \in \mathbb{R}^{m \times n}$, where $n = 10^9$ and $m = 2 \times 10^9$

Minibatching for ERM

[Qu, R & Zhang 14]

Data sparsity

$$1 \leq \tilde{\omega} \leq n$$

"Fully sparse data"

A normalized measure of average sparsity of the data

"Fully dense data"

Quartz [Qu, R & Zhang 14]

Complexity of Quartz

Fully sparse data ($\tilde{\omega} = 1$)	$\frac{n}{\tau} + \frac{\max_i L_i}{\lambda \gamma \tau}$
Fully dense data ($\tilde{\omega} = n$)	$\frac{n}{\tau} + \frac{\max_i L_i}{\lambda \gamma}$
Any data ($1 \leq \tilde{\omega} \leq n$)	$\frac{n}{\tau} + \frac{\left(1 + \frac{(\tilde{\omega}-1)(\tau-1)}{n-1}\right) \max_i L_i}{\lambda \gamma \tau}$

$\equiv T(\tau)$

Speedup

Assume the data is normalized: $L_i \equiv \lambda_{\max}(A_i^\top A_i) \leq 1$

Then:

$$T(\tau) = \frac{\left(1 + \frac{(\tilde{\omega}-1)(\tau-1)}{(n-1)(1+\lambda\gamma n)}\right)}{\tau} \times T(1)$$

Linear speedup up to a certain data-independent minibatch size:

$$\tau \leq 2 + \lambda\gamma n \quad \Rightarrow \quad T(\tau) \leq \frac{2}{\tau} \times T(1)$$

Further data-dependent speedup, up to the extreme case:

$$\tilde{\omega} = \mathcal{O}(\lambda\gamma n) \quad \Rightarrow \quad T(\tau) = \mathcal{O}\left(\frac{T(1)}{\tau}\right)$$

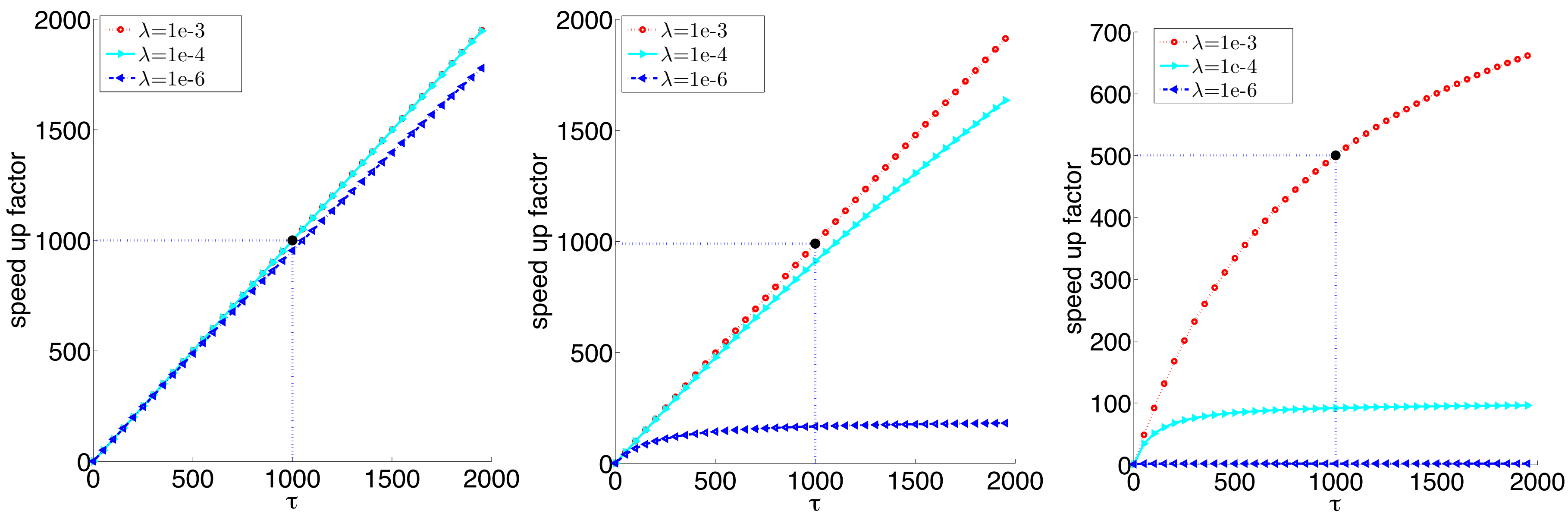
Quartz: Parallelization Speedup

examples: $n = 10^6$

Smoothness of loss functions: $\gamma = 1$

Low regularization: $\lambda = 1/n$

High regularization: $\lambda = 1/\sqrt{n}$



Sparse Data $\tilde{\omega} = 10^2$

Denser Data $\tilde{\omega} = 10^4$

Fully Dense Data $\tilde{\omega} = 10^6$

Primal-dual methods with tau-nice sampling

$L_i = 1$

Algorithm	Iteration complexity	g
SDCA [S-Shwartz & Zhang 12]	$n + \frac{1}{\lambda\gamma}$	$\frac{1}{2}\|\cdot\|^2$
ASDCA [S-Shwartz & Zhang 13a]	$4 \times \max\left\{\frac{n}{\tau}, \sqrt{\frac{n}{\lambda\gamma\tau}}, \frac{1}{\lambda\gamma\tau}, \frac{n^{\frac{1}{3}}}{(\lambda\gamma\tau)^{\frac{2}{3}}}\right\}$	$\frac{1}{2}\|\cdot\|^2$
SPDC [Zhang & Xiao 14]	$\frac{n}{\tau} + \sqrt{\frac{n}{\lambda\gamma\tau}}$	general
Quartz	$\frac{n}{\tau} + \left(1 + \frac{(\tilde{\omega}-1)(\tau-1)}{n-1}\right)\frac{1}{\lambda\gamma\tau}$	general

For sufficiently sparse data, Quartz wins even when compared against accelerated methods

	Algorithm	$\gamma\lambda n = \Theta(\frac{1}{\tau})$	$\gamma\lambda n = \Theta(1)$	$\gamma\lambda n = \Theta(\tau)$	$\gamma\lambda n = \Theta(\sqrt{n})$
		$\kappa = n\tau$	$\kappa = n$	$\kappa = n/\tau$	$\kappa = \sqrt{n}$
	SDCA	$n\tau$	n	n	n
Accelerated	ASDCA	n	$\frac{n}{\sqrt{\tau}}$	$\frac{n}{\tau}$	$\frac{n}{\tau} + \frac{n^{3/4}}{\sqrt{\tau}}$
Accelerated	SPDC	n	$\frac{n}{\sqrt{\tau}}$	$\frac{n}{\tau}$	$\frac{n}{\tau} + \frac{n^{3/4}}{\sqrt{\tau}}$
	Quartz	$n + \tilde{\omega}\tau$	$\frac{n}{\tau} + \tilde{\omega}$	$\frac{n}{\tau}$	$\frac{n}{\tau} + \frac{\tilde{\omega}}{\sqrt{n}}$

Trick 5
Distributed Implementation

HYDRA

Distributed Quartz: Perform the Dual Updates in a Distributed Manner

Quartz STEP 2: DUAL UPDATE

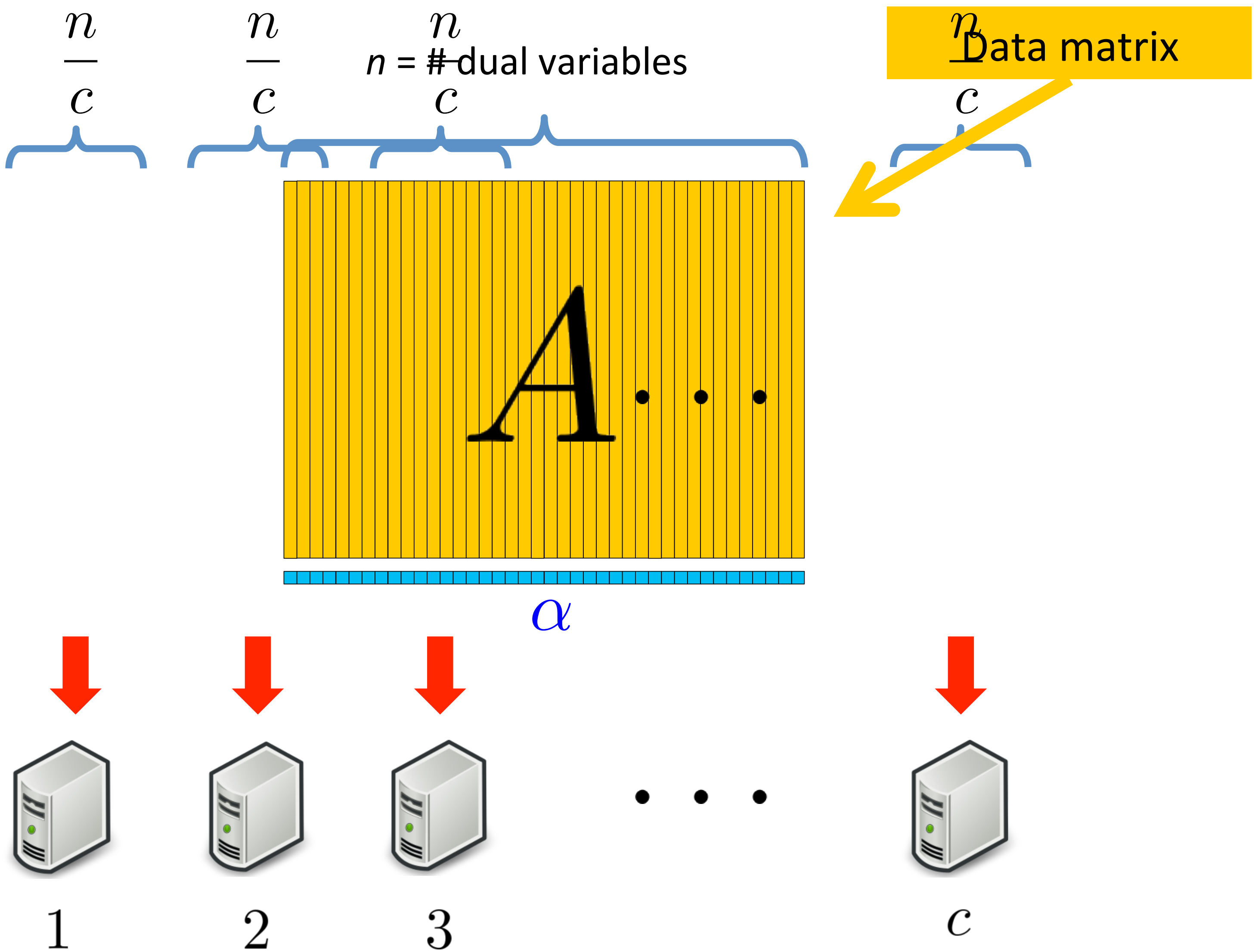
Choose a random set S_t of dual variables

For $i \in S_t$ do

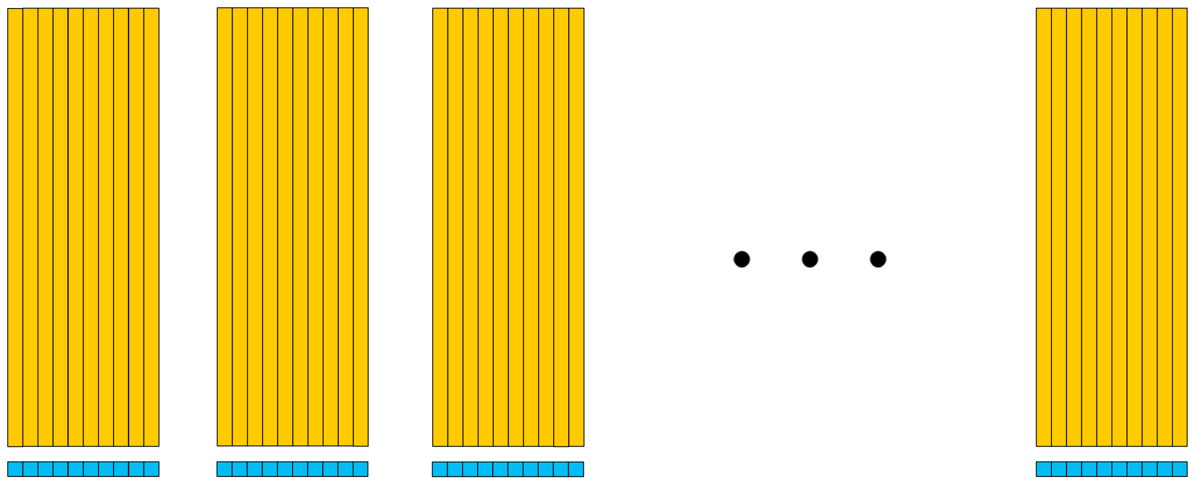
$$\alpha_i^{t+1} \leftarrow \left(1 - \frac{\theta}{p_i}\right) \alpha_i^t + \frac{\theta}{p_i} \left(-\nabla \phi_i (A_i^\top w^{t+1})\right)$$

Data required to compute the update

Distribution of Data

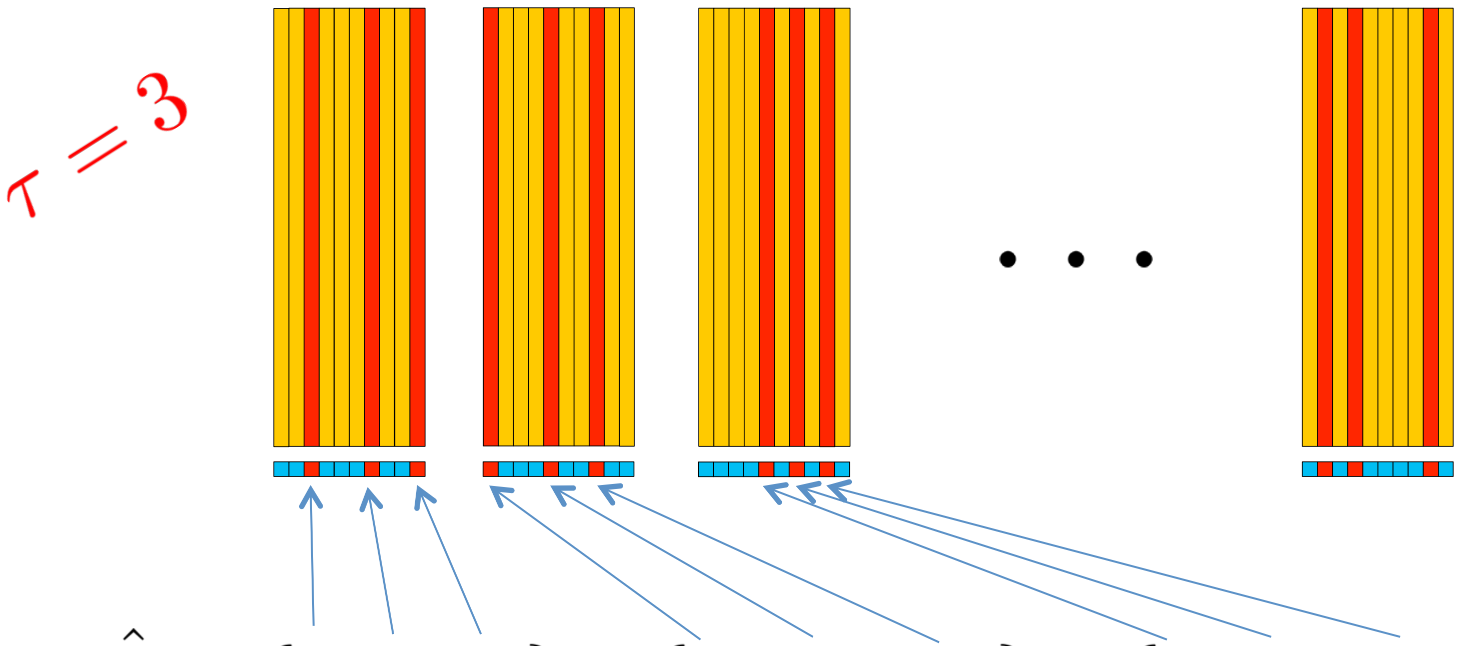


Distributed sampling



Distributed sampling

Each node independently picks τ dual variables from those it owns, uniformly at random



$$\hat{S} = \{3, 7, 10\} \cup \{11, 15, 18\} \cup \{25, 27, 29\} \cup \cdots$$

Random set of dual variables

Also see: CoCoA+ [Ma, Smith, Jaggi et al 15]

Complexity of Distributed Quartz

$$\frac{n}{c\tau} + \frac{\text{Something that looks complicated}}{\lambda\gamma c\tau}$$

$$\frac{n}{c\tau} + \max_i \frac{\lambda_{\max}\left(\sum_{j=1}^{d}\left(1 + \frac{(\tau-1)(\omega_j-1)}{\max\{n/c-1,1\}} + \left(\frac{\tau c}{n} - \frac{\tau-1}{\max\{n/c-1,1\}}\right)\frac{\omega_j'-1}{\omega_j'}\omega_j\right)A_{ji}^\top A_{ji}\right)}{\lambda\gamma c\tau}$$

Experiment

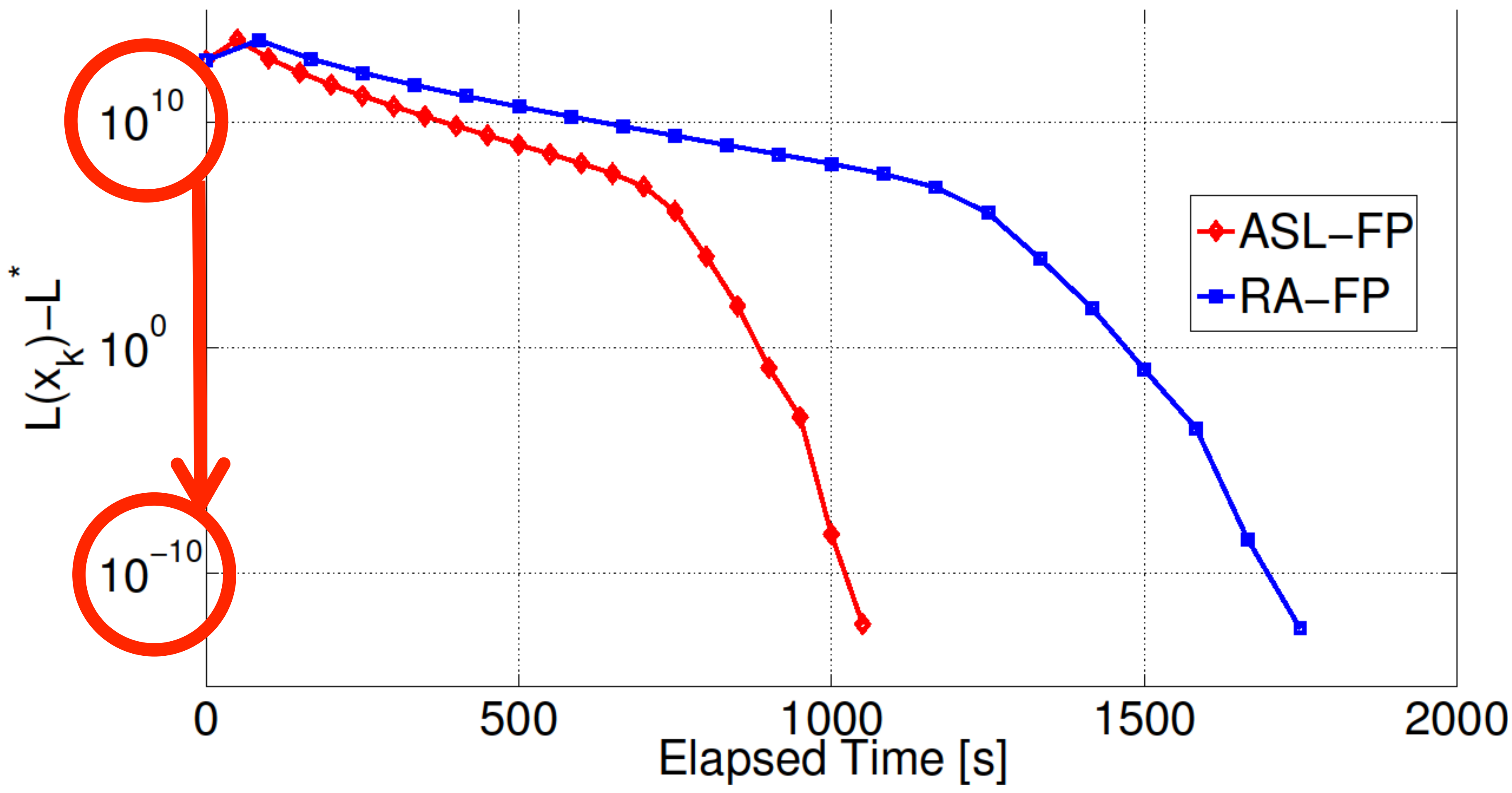
Machine: 128 nodes of Hector Supercomputer (4096 cores)

Problem: LASSO, n = 1 billion, d = 0.5 billion, 3 TB

Algorithm: HYDRA with $c = 512$

[R & Takáč 13a]

LASSO: 3TB data + 128 nodes



Experiment

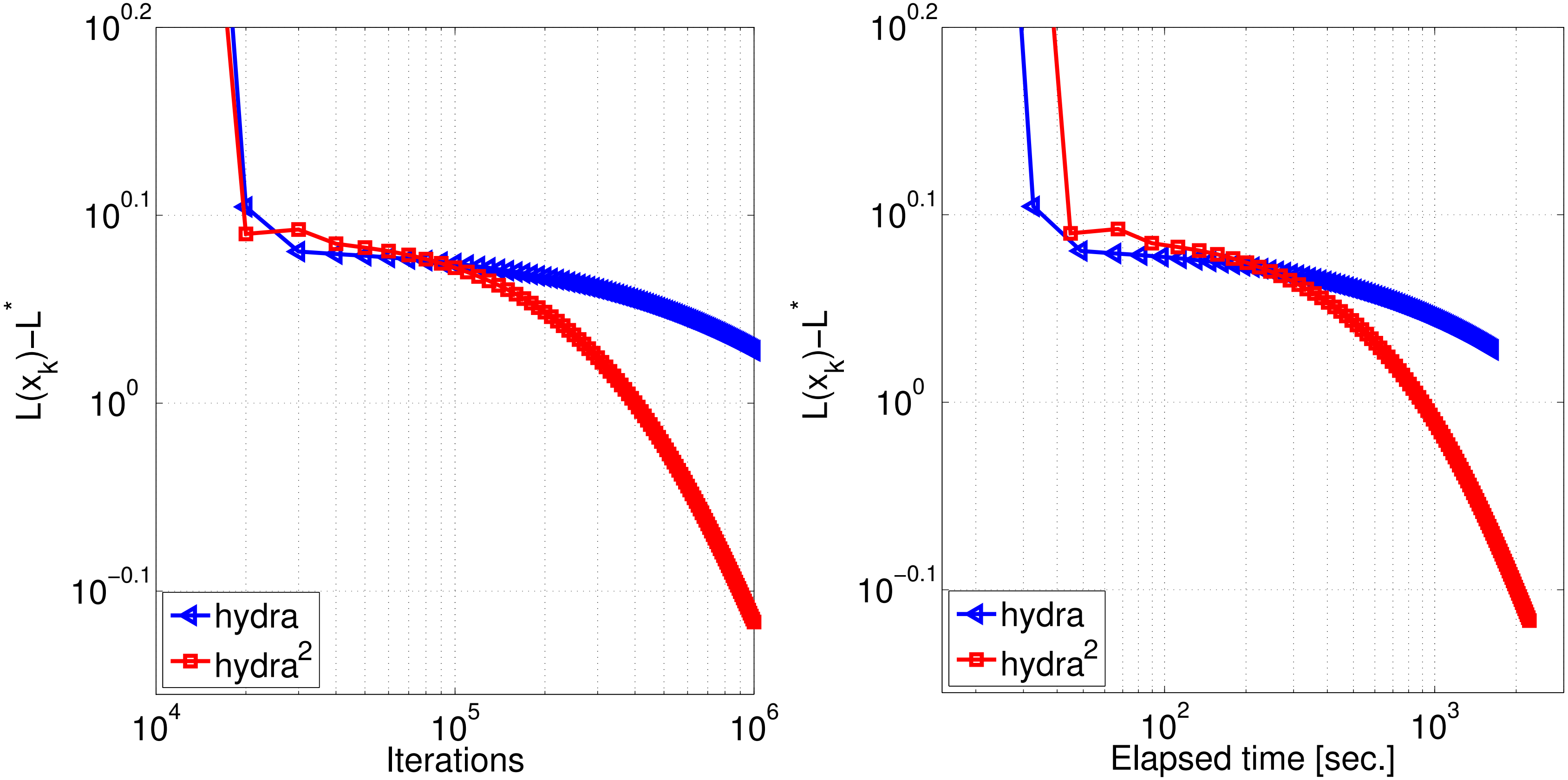
Machine: 128 nodes of Archer Supercomputer

Problem: LASSO, n = 5 million, d = 50 billion, 5 TB
(60,000 nnz per row of A)

Algorithm HYDRA^2 with c = 256

[Fercoq et al 14]

LASSO: 5TB data (d = 50b) + 128 nodes



THE END