Iteration Complexity of Randomized Block-Coordinate Descent Methods for Minimizing a Composite Function

Peter Richtárik

The University of Edinburgh

Based on:

P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. 2011.

P. Richtárik and M. Takáč. Efficient serial and parallel coordinate descent methods for huge scale truss topology design. 2011.

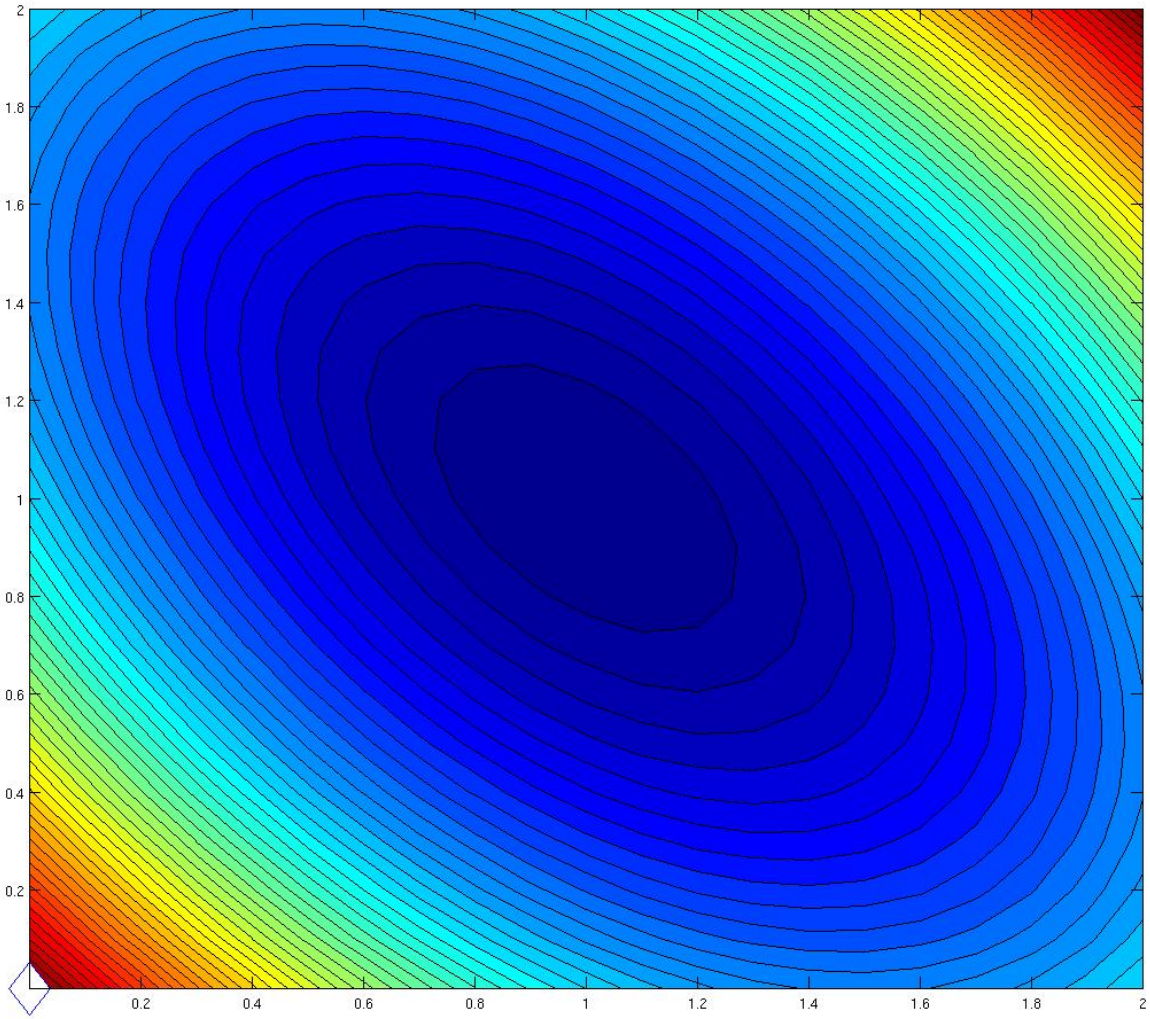
FoCM—Budapest—July 4–6, 2011

Part I.

Coordinate Descent

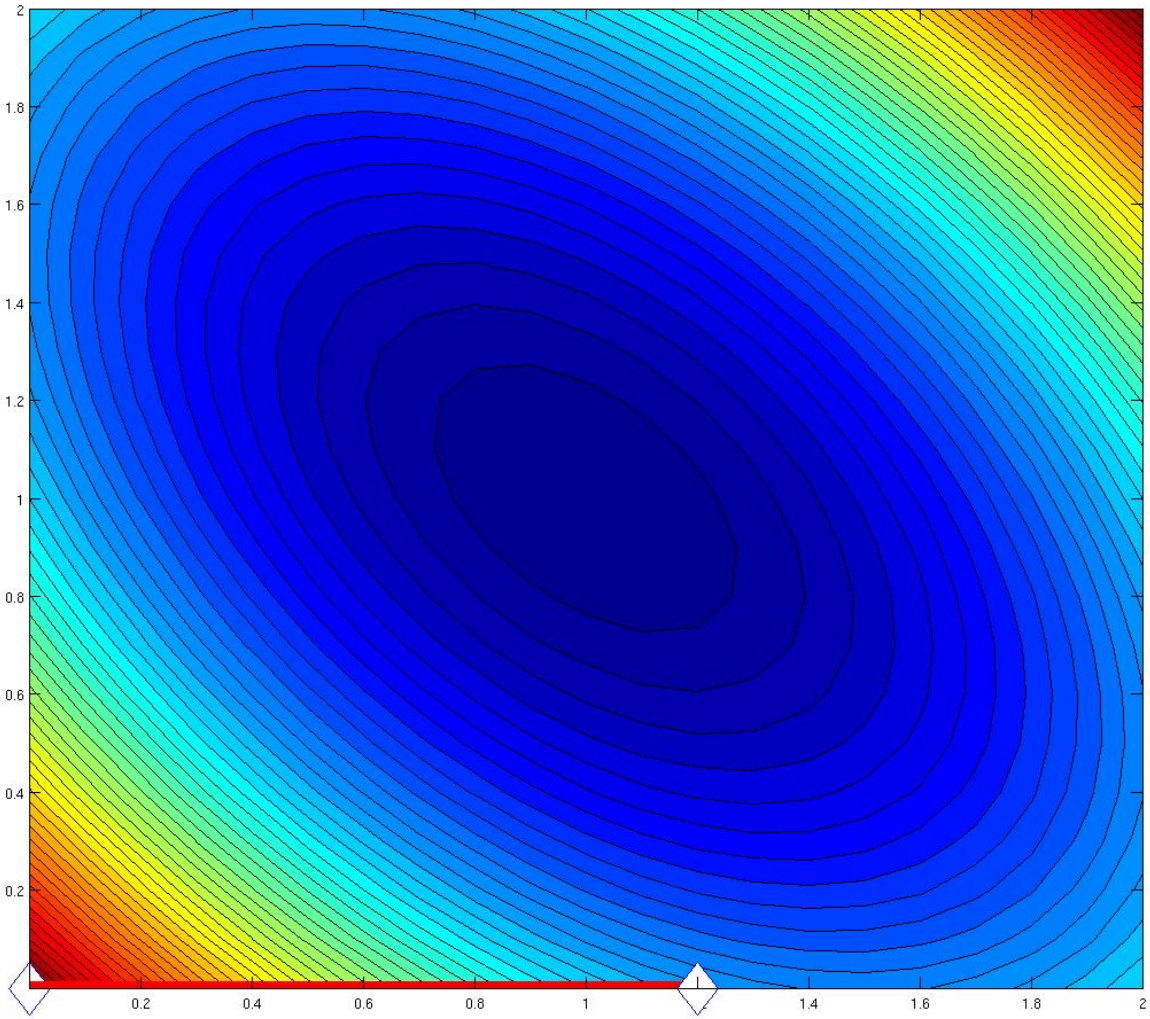
Coordinate Descent: a 2D example

$$f(x) = \frac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



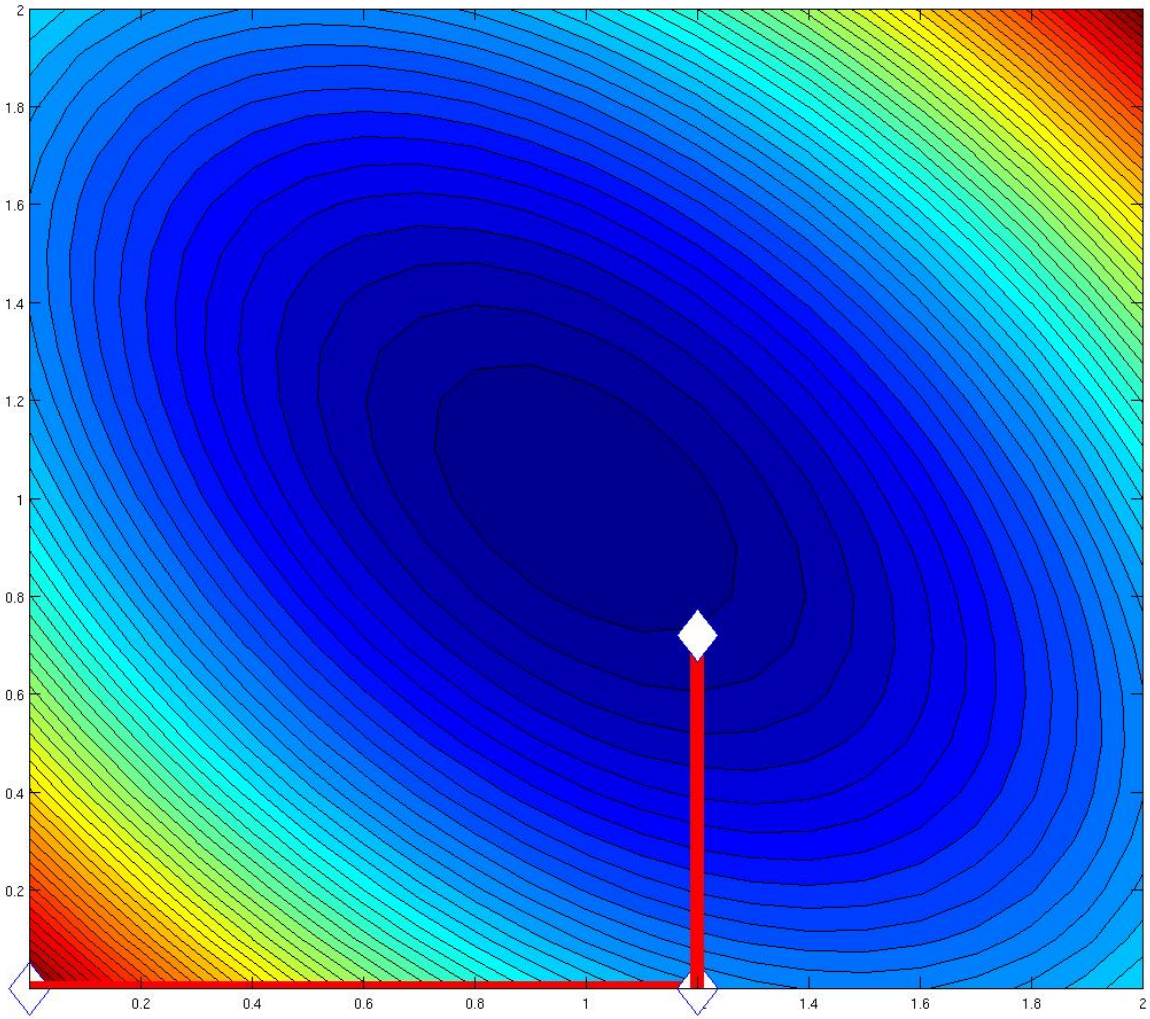
Coordinate Descent: a 2D example

$$f(x) = \tfrac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



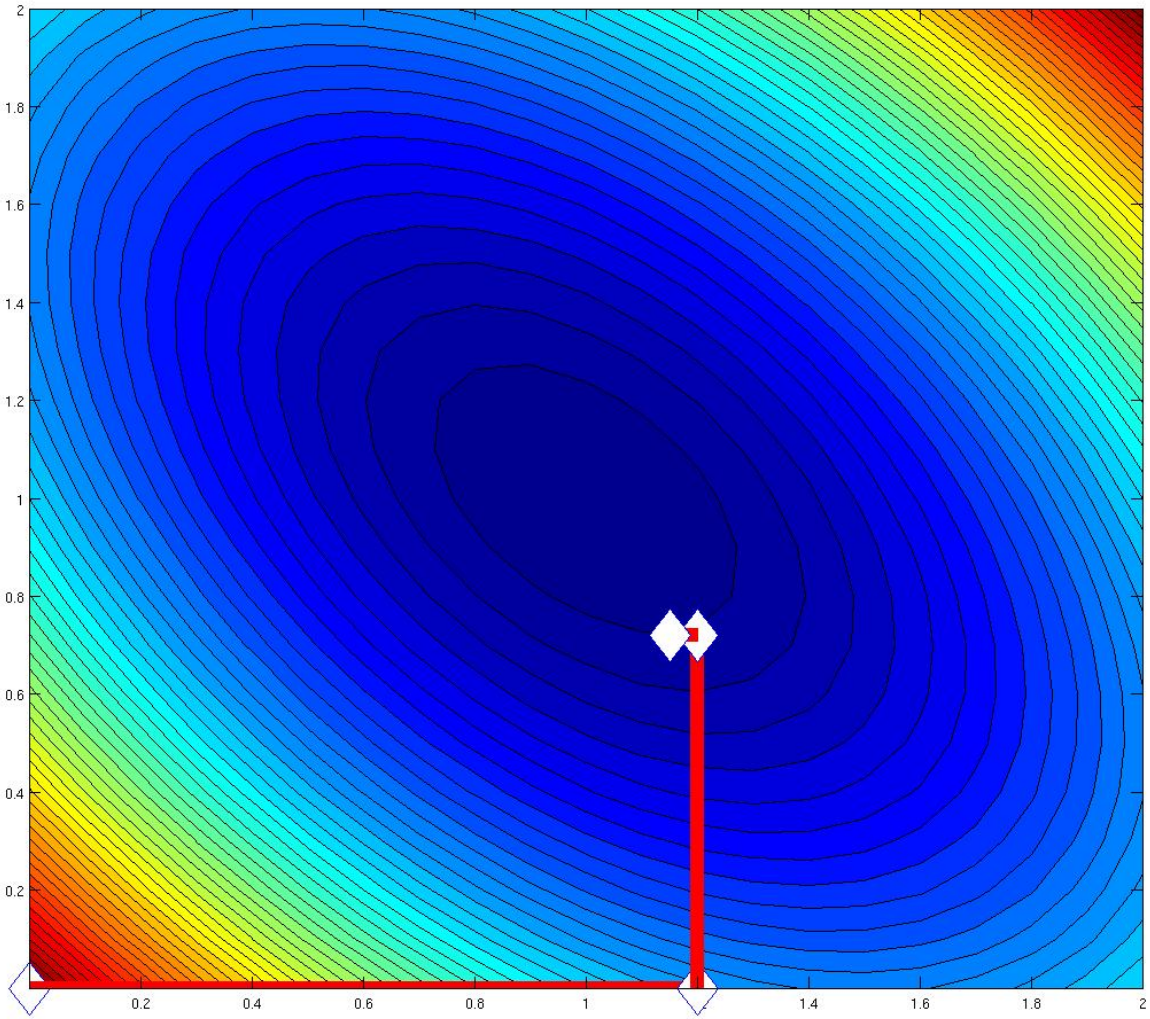
Coordinate Descent: a 2D example

$$f(x) = \frac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



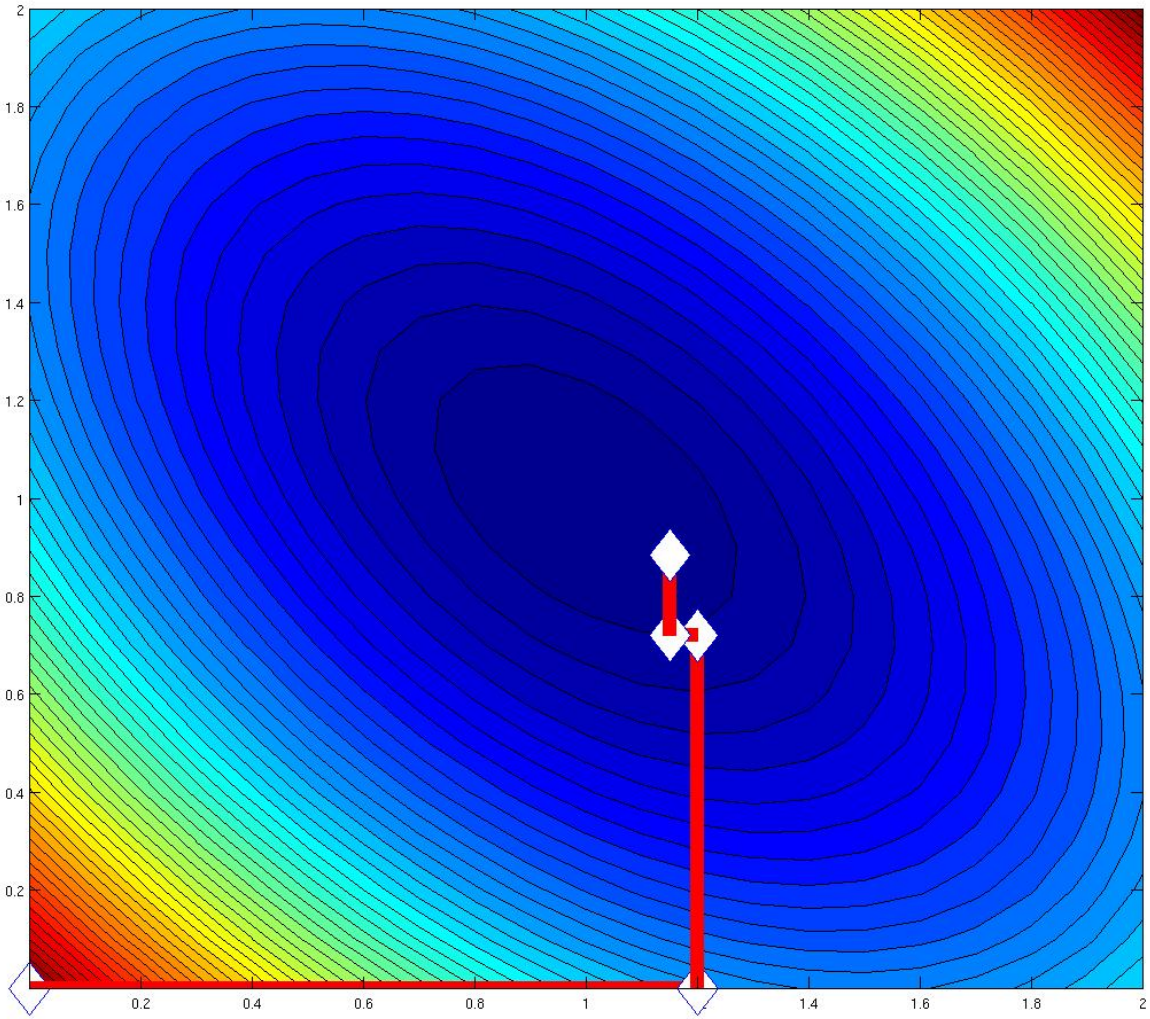
Coordinate Descent: a 2D example

$$f(x) = \tfrac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



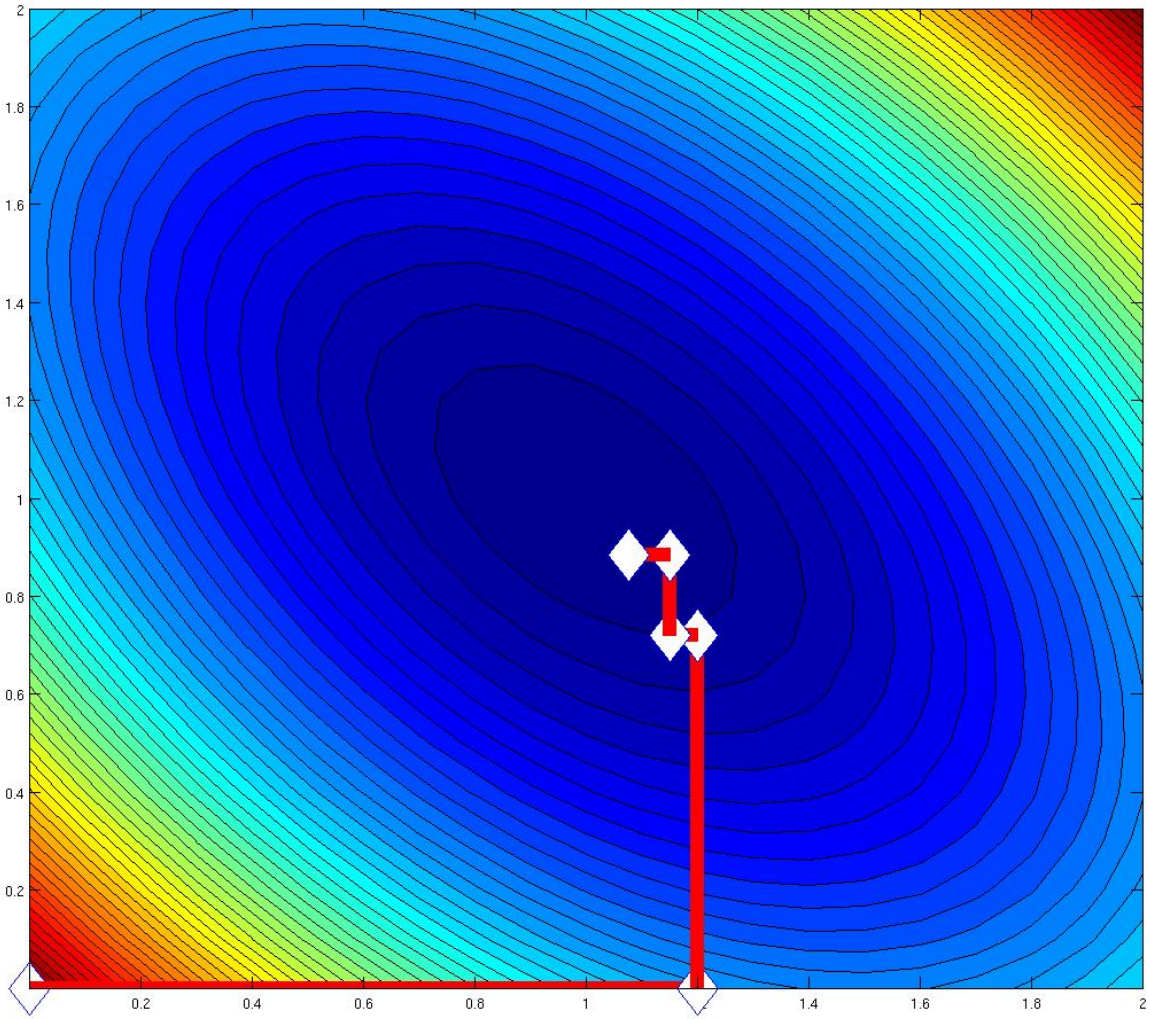
Coordinate Descent: a 2D example

$$f(x) = \tfrac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



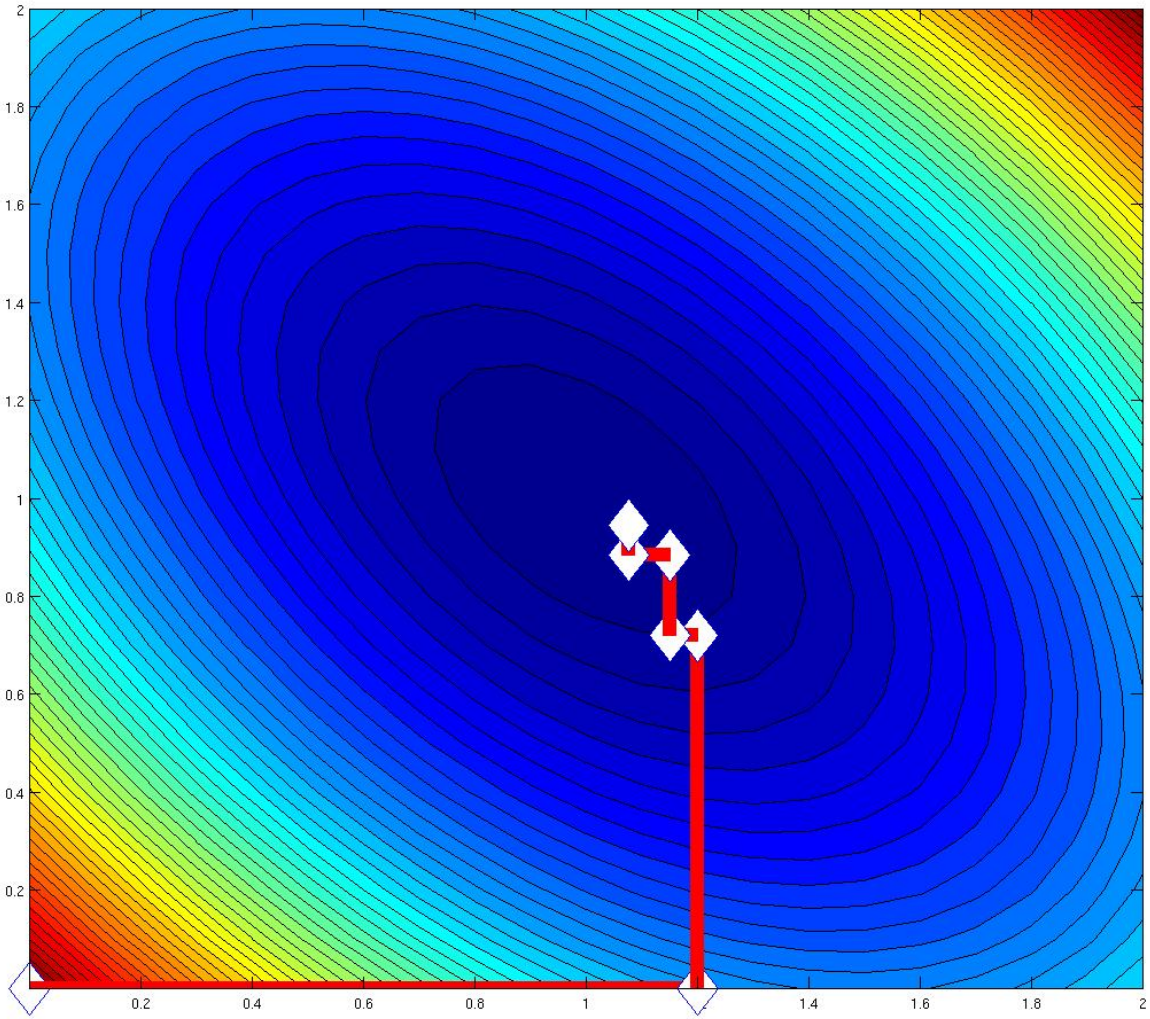
Coordinate Descent: a 2D example

$$f(x) = \frac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



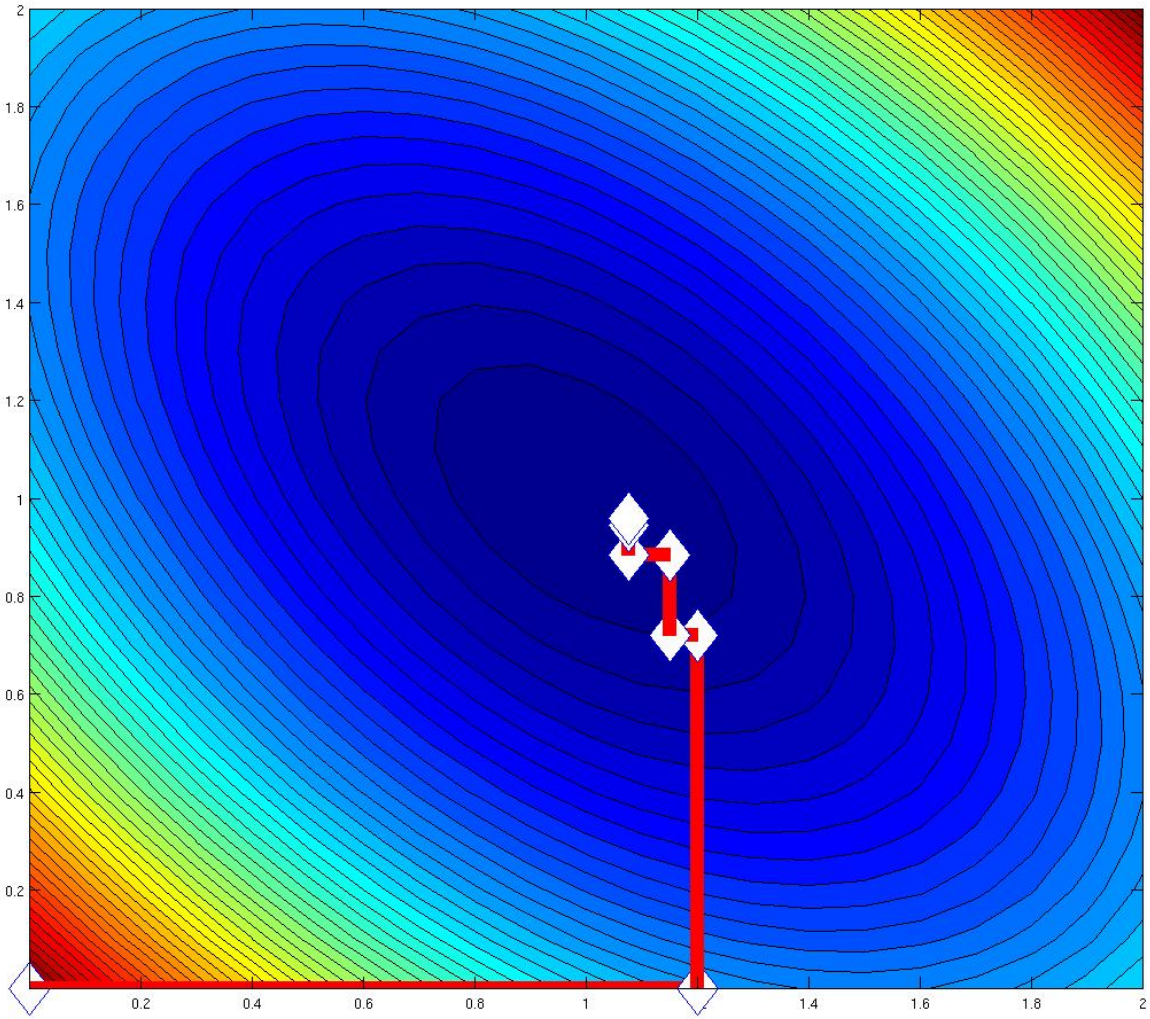
Coordinate Descent: a 2D example

$$f(x) = \tfrac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



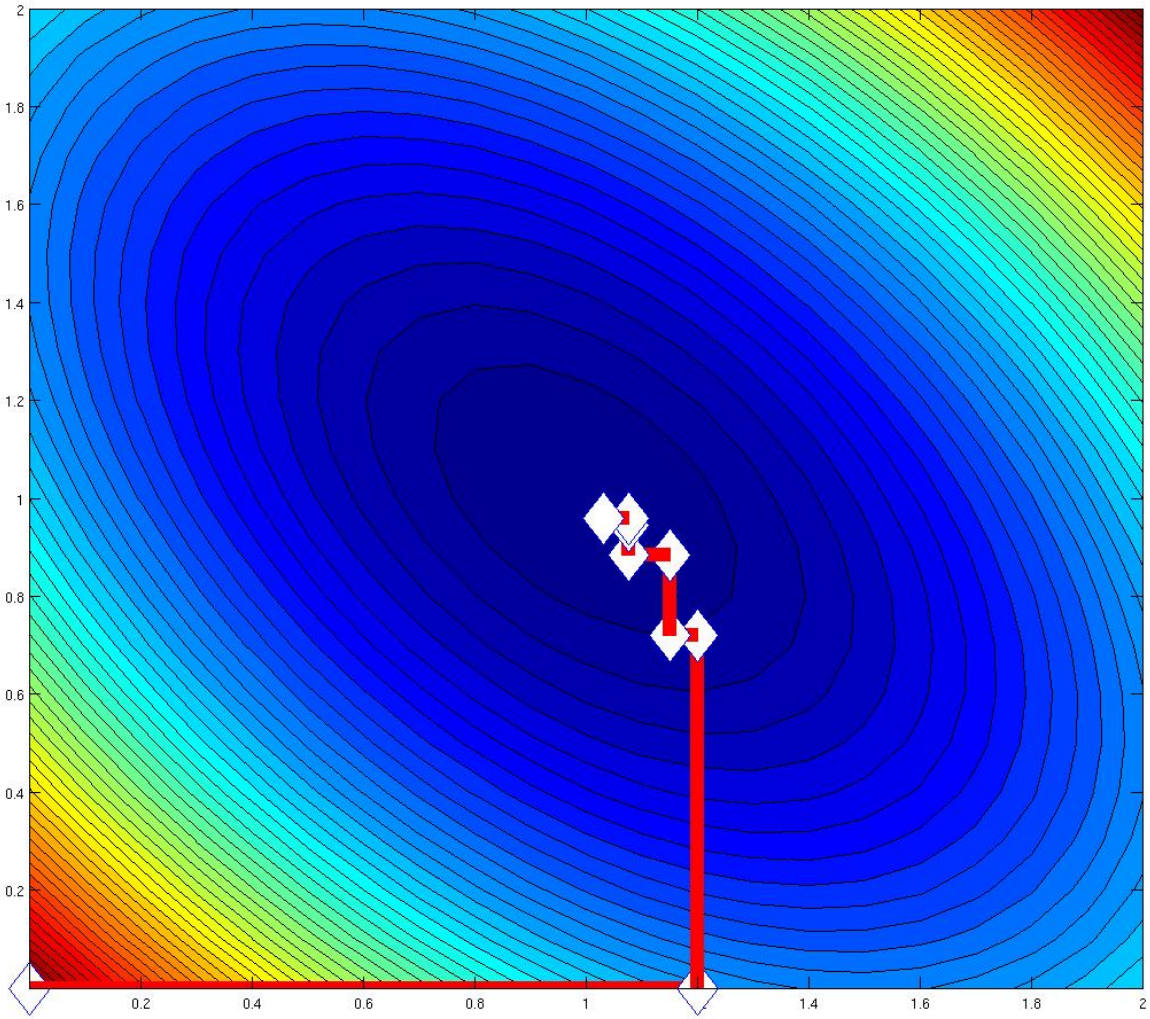
Coordinate Descent: a 2D example

$$f(x) = \tfrac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



Coordinate Descent: a 2D example

$$f(x) = \frac{1}{2} x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



The problem

We are interested in solving the problem

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

where

- f is convex and smooth
- Ψ is convex, simple and block-separable
- N is huge ($N \gg 10^6$)

$\Psi(x)$	meaning
0	smooth minimization
$\lambda\|x\|_1$	sparsity-inducing term
indicator function of a set	block-separable constraints

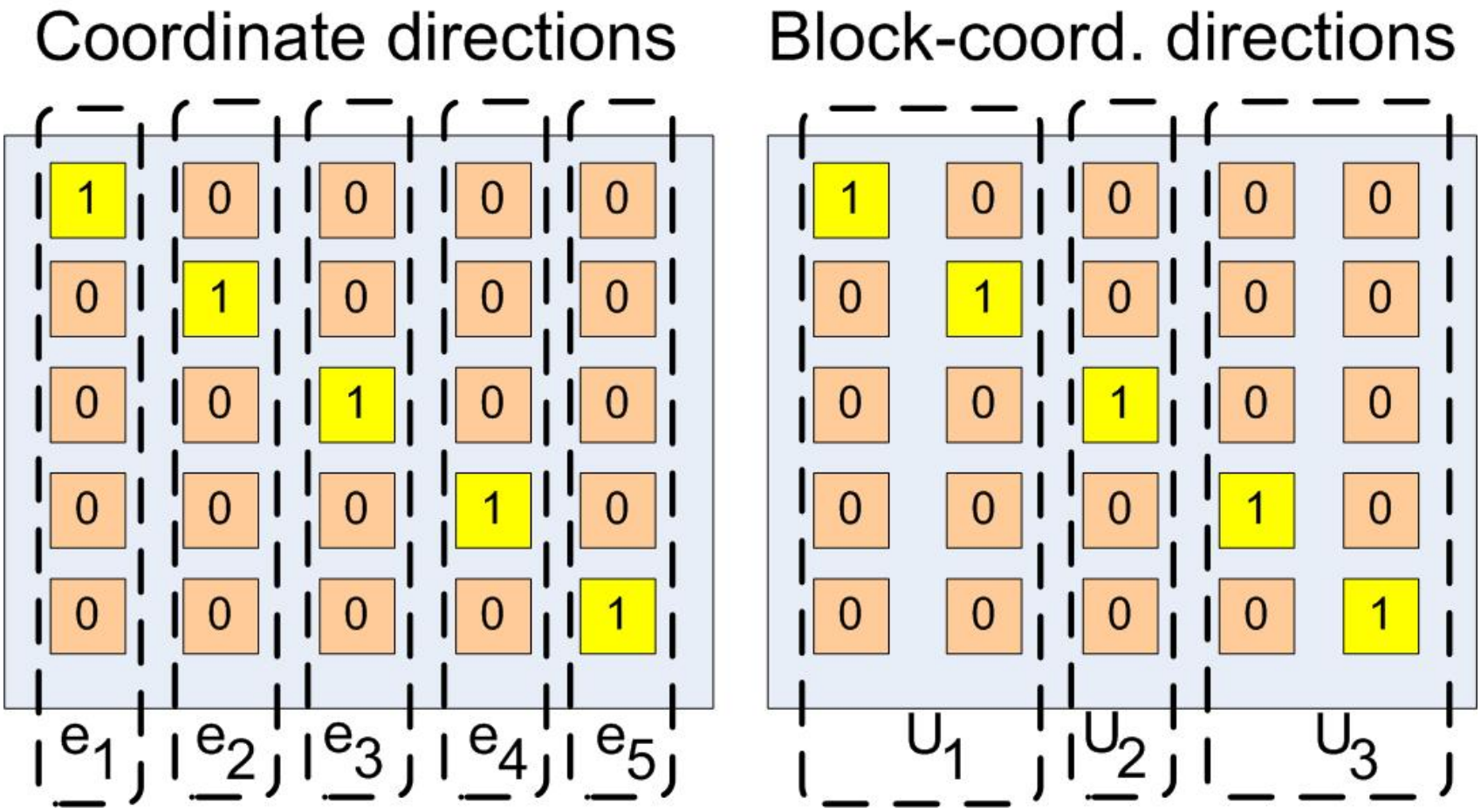
Applications

- Truss topology design
- Sparse regression, LASSO
- (Sparse) group LASSO
- Elastic Net ...
- L_1 regularized linear support vector machines (SVM) with various (smooth) loss functions

Part II.

Four Boring Slides

BS1: Block structure of $\mathbf{R}^N$



BS2: Block structure of $\mathbf{R}^N$

Decomposition of $\mathbf{R}^N$ into n blocks

- Let $I = [U_1, U_2, \ldots, U_n]$ be a decomposition of the $N \times N$ identity matrix I into n submatrices, with U_i being of size $N \times N_i$, where $\sum_i N_i = N$.
- Then any vector $x \in \mathbf{R}^N$ can be written uniquely as

$$x = \sum_{i=1}^{n} U_i x^{(i)}, \qquad \text{where} \qquad x^{(i)} = U_i^T x \in \mathbf{R}_i \equiv \mathbf{R}^{N_i}.$$

- For simplicity we will write $x = (x^{(1)}, \ldots, x^{(n)})$.

Block updates

For $x_k \in \mathbf{R}^N$ and $t \in \mathbf{R}_i$

$$x_{k+1} = x_k + U_i t \qquad \Longleftrightarrow \qquad x_{k+1}^{(j)} = \begin{cases} x_k^{(j)} + t, & \text{if } j = i, \\ x_k^{(j)}, & \text{otherwise.} \end{cases}$$

BS3: Block norms and induced norms

Block norms (norms in $\mathbf{R}_i$): We equip $\mathbf{R}_i$ with a pair of conjugate Euclidean norms:

$$\|t\|_{(i)} = \langle B_i t, t\rangle^{1/2}, \qquad \|t\|_{(i)}^* = \langle B_i^{-1} t, t\rangle^{1/2}, \qquad t \in \mathbf{R}_i,$$

where $B_i \in \mathbf{R}^{N_i \times N_i}$ is a positive definite matrix.

Induced norms (norms in $\mathbf{R}^N$): For fixed positive scalars $w_1, \ldots, w_n$ and $x \in \mathbf{R}^N$ we define

$$\|x\|_W = \left[\sum_{i=1}^n w_i \|x^{(i)}\|_{(i)}^2\right]^{1/2},$$

where $W = \mathrm{Diag}(w_1, \ldots, w_n)$. The conjugate norm is defined by

$$\|y\|_W^* = \max_{\|x\|_W \le 1} \langle y, x\rangle = \left[\sum_{i=1}^n w_i^{-1} (\|y^{(i)}\|_{(i)}^*)^2\right]^{1/2}.$$

BS4: Assumptions on f and Ψ

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

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

where

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

An important consequence of (1) is the following inequality

$$f(x + U_i t) \leq f(x) + \langle \nabla_i f(x), t \rangle + \frac{L_i}{2} \|t\|^2_{(i)}. \quad (2)$$

We denote $L = \text{Diag}(L_1, \ldots, L_n)$.

Ψ is block-separable:

$$\Psi(x) = \sum_{i=1}^{n} \Psi_i(x^{(i)}).$$

Part III.

UCDC: Uniform Coordinate Descent for Composite Functions

The UCDC Algorithm

An upper bound on $F(x + U_i t)$ is readily available:

$$F(x + U_i t) = f(x + U_i t) + \Psi(x + U_i t)$$
$$\overset{(2)}{\leq} f(x) + \underbrace{\langle \nabla_i f(x), t \rangle + \tfrac{L_i}{2} \|t\|_{(i)}^2 + \Psi_i(x^{(i)} + t)}_{\overset{\text{def}}{=} V_i(x,t)} + \sum_{j \neq i} \Psi_j(x^{(j)}).$$

The UCDC Algorithm

An upper bound on $F(x+U_i t)$ is readily available:

$$F(x+U_i t) = f(x+U_i t) + \Psi(x+U_i t)$$
$$\overset{(2)}{\leq} f(x) + \underbrace{\langle \nabla_i f(x), t\rangle + \tfrac{L_i}{2}\|t\|^2_{(i)} + \Psi_i(x^{(i)}+t)}_{\overset{\text{def}}{=} V_i(x,t)} + \sum_{j\neq i} \Psi_j(x^{(j)}).$$

Algorithm 1 UCDC(x_0)

for $k = 0, 1, 2, \ldots$
  Choose block coordinate $i \in \{1, 2, \ldots, n\}$ with probability $\frac{1}{n}$
  $T^{(i)}(x_k) = \arg\min\{ V_i(x_k, t) \ : \ t \in \mathbf{R}_i \equiv \mathbf{R}^{N_i}\}$
  $x_{k+1} = x_k + U_i T^{(i)}(x_k)$
end for

- In each step we minimize a function of N_i variables
- We assume Ψ is simple: $T^{(i)}(x_k)$ can be computed in closed form
- If $\Psi \equiv 0$, then $T^{(i)}(x_k) = -\frac{1}{L_i} B_i^{-1} \nabla_i f(x)$

The Main Result

Theorem 1

Choose initial point x_0 and confidence level $\rho \in (0,1)$. Further, let the target accuracy $\epsilon > 0$ and iteration counter k be chosen in any of the following two ways:

(i) $\epsilon < F(x_0) - F^*$ and

$$k \geq \frac{2n \max\{\mathcal{R}_L^2(x_0), F(x_0)-F^*\}}{\epsilon}\left(1 + \log\frac{1}{\rho}\right) + 2 - \frac{2n \max\{\mathcal{R}_L^2(x_0), F(x_0)-F^*\}}{F(x_0)-F^*},$$

(ii) $\epsilon < \min\{\mathcal{R}_L^2(x_0), F(x_0) - F^*\}$ and

$$k \geq \frac{2n\mathcal{R}_L^2(x_0)}{\epsilon} \log\frac{F(x_0)-F^*}{\epsilon\rho}.$$

If x_k is the random point generated by UCDC(x_0) as applied to F, then

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

Performance on Strongly Convex Functions

Theorem 2

Let F be strongly convex with respect to $\|\cdot\|_L$ with convexity parameter $\mu > 0$ and choose accuracy level $\epsilon > 0$, confidence level $0 < \rho < 1$, and

$$k \geq \frac{n}{1-\gamma_\mu} \log\left(\frac{F(x_0)-F^*}{\rho\epsilon}\right),$$

where γ_μ is given by

$$\gamma_\mu = \begin{cases} 1 - \frac{\mu}{4}, & \text{if } \mu \leq 2, \\ \frac{1}{\mu}, & \text{otherwise.} \end{cases} \tag{3}$$

If x_k is the random point generated by UCDC(x_0), then

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

Part IV.

RCDS: Randomized Coordinate Descent for Smooth Functions

RCDS: The Algorithm

In the smooth case we allow

- for arbitrary norms $\|\cdot\|_{(i)}$,
- for an arbitrary probability vector $p > 0$.

RCDS: The Algorithm

In the smooth case we allow

- for arbitrary norms $\|\cdot\|_{(i)}$,
- for an arbitrary probability vector $p > 0$.

Algorithm 2 RCDS(p, x_0)

for $k = 0, 1, 2, \ldots$
  Choose $i_k = i \in \{1, 2, \ldots, n\}$ with probability p_i
  $x_{k+1} = x_k - \frac{1}{L_i} U_i (\nabla_i f(x_k))^{\#}$
end for

where by $s^{\#}$ for $s \in \mathbf{R}_i$ we denote an optimal solution of the problem

$$\min_t \left\{ u(s) \stackrel{\text{def}}{=} -\langle s, t \rangle + \tfrac{1}{2}\|t\|_{(i)}^2 \right\}.$$

Performance on Smooth Functions

Theorem 3

Choose initial point x_0, target accuracy

$$0 < \epsilon < \min\{f(x_0) - f^*, 2\mathcal{R}^2_{LP^{-1}}(x_0)\},$$

target confidence $0 < \rho < 1$ and

$$k \geq \frac{2\mathcal{R}^2_{LP^{-1}}(x_0)}{\epsilon}\left(1 + \log\frac{1}{\rho}\right) - 2.$$

If x_k is the random point generated by RCDS(p, x_0) as applied to f, then

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

Part V.

Theoretical Comparison with Other Methods

Comparison with other coordinate descent methods for composite minimization

	Lip	Ψ	block	coordinate choice	$\mathbf{E}(F(x_k) < \epsilon)$	1 iter
[YT09]	$L(\nabla f)$	separable	Yes	greedy	$O(\frac{nL(\nabla f)\|x^* - x_0\|_2^2}{\epsilon})$	$O(n^2)$
[SST09]	$L_i = \beta$	$\|\cdot\|_1$	No	$\frac{1}{n}$	$O(\frac{n\beta\|x^* - x_0\|_2^2}{\epsilon})$	cheap
[ST10]	$L(\nabla f)$	$\|\cdot\|_1$	No	cyclic	$O(\frac{nL(\nabla f)\|x^* - x_0\|_2^2}{\epsilon})$	cheap
Alg 1	L_i	separable	Yes	$\frac{1}{n}$	$O(\frac{n\|x^* - x_0\|_L^2}{\epsilon})$	cheap

[YT09] S. Yun and P. Tseng. A block-coordinate gradient descent method for linearly constrained nonsmooth separable optimization. *Journal of Optimization Theory and Applications*, 140:513–535, 2009.

[SST09] S. Shalev-Shwartz and A. Tewari. Stochastic methods for ℓ_1 regularized loss minimization. In: ICML 2009.

[ST09] A. Saha and A. Tewari. On the finite time convergence of cyclic coordinate descent methods. *arXiv:1005.2146*, 2010.

Comparison with Nesterov's randomized coordinate descent

. . . for achieving $\mathbf{P}(F(x_k) - F^* \leq \epsilon) \geq 1 - \rho$ in the general case (F not strongly convex):

Alg	Ψ	p_i	norms	complexity	objective
[N10]	0	$\frac{L_i}{S}$	Euclid.	$\frac{8n\mathcal{R}_L^2(x_0)}{\epsilon} \log \frac{4(f(x_0)-f^*)}{\epsilon\rho}$	$f(x) + \frac{\epsilon\|x-x_0\|_L^2}{8\mathcal{R}_L^2(x_0)}$
[N10]	0	$\frac{1}{n}$	Euclid.	$(2n + \frac{8S\mathcal{R}_I^2(x_0)}{\epsilon}) \log \frac{4(f(x_0)-f^*)}{\epsilon\rho}$	$f(x) + \frac{\epsilon\|x-x_0\|_I^2}{8\mathcal{R}_I^2(x_0)}$
Alg1	0	> 0	general	$\frac{2\mathcal{R}^2_{LP^{-1}}(x_0)}{\epsilon}(1 + \log \frac{1}{\rho}) - 2$	f
Alg2	$\neq 0$	$\frac{1}{n}$	Euclid.	$\frac{2nM}{\epsilon}(1 + \log \frac{1}{\rho})$ $\frac{2n\mathcal{R}_L^2(x_0)}{\epsilon} \log \frac{F(x_0)-F^*}{\epsilon\rho}$	F

Symbols used in the table: $S = \sum L_i$, $M = \max\{\mathcal{R}_L^2(x_0), F(x_0) - F^*\}$.

[N10] Yu. Nesterov, Efficiency of coordinate descent methods on huge-scale optimization problems. *CORE Discussion Paper # 2010/2*, 2010.

Part VI.

Experiments with a Serial Code

Application 2: Sparse Regression

Problem formulation

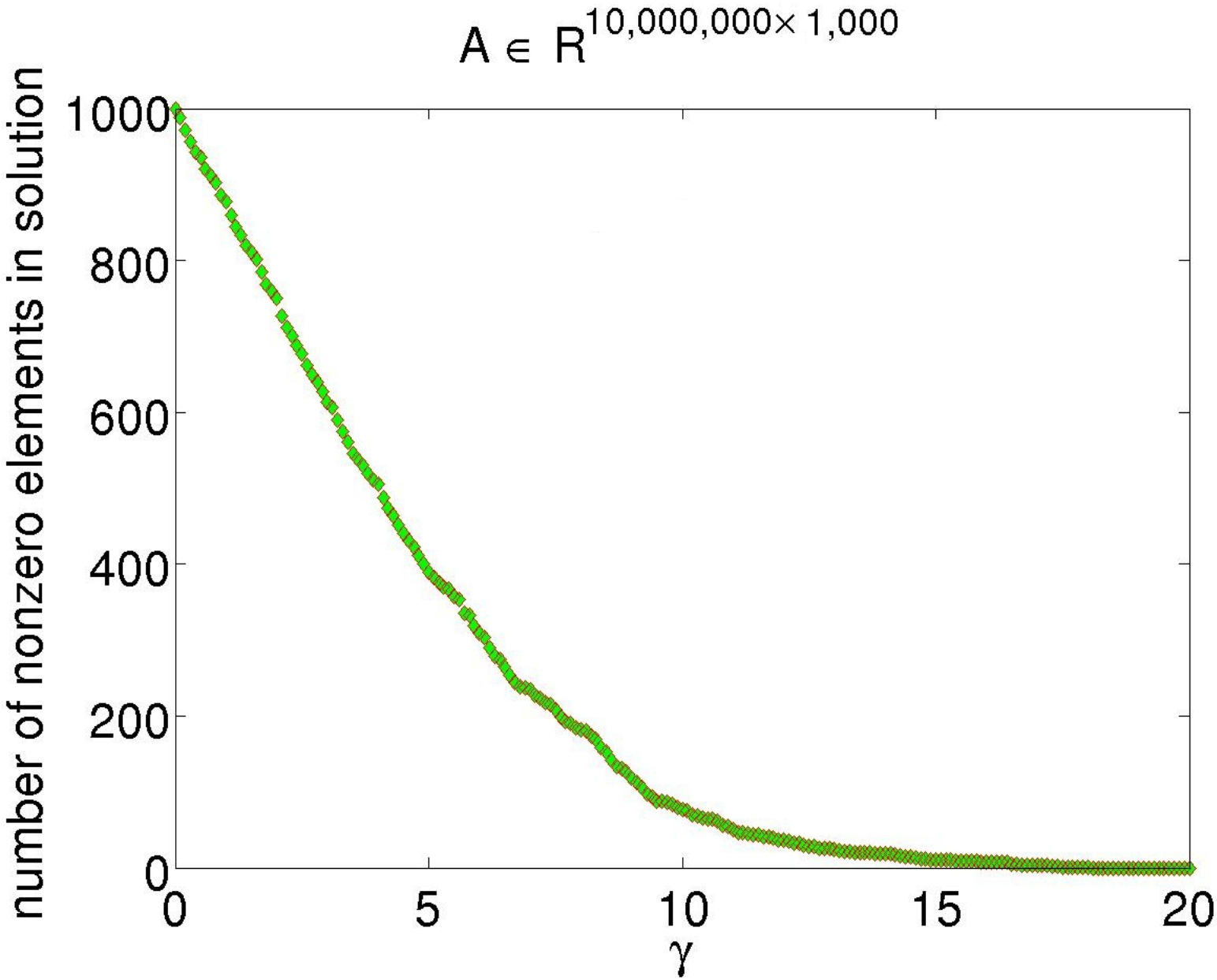
$$\min_{x \in \mathbf{R}^n} \tfrac{1}{2}\|Ax - b\|_2^2 + \gamma \|x\|_1$$

Lipschitz constants

$$L_i = a_i^T a_i \quad \text{(easy to compute)}$$

Note: Lipschitz constant of $\nabla f = \lambda_{\max}(A^T A)$ (hard to compute)

Increasing γ induces sparsity in the solution



Uniform coordinate descent for sparse regression

UCDC for Sparse Regression

Choose $x_0 = 0$ and set $g_0 = Ax_0 - b = -b$

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

Choose $i_k = i \in \{1, 2, \ldots, n\}$ with probability $\frac{1}{n}$

$\alpha = a_i^T g_k$,

$$T^{(i)}(x_k) = \begin{cases} -\frac{\alpha+\gamma}{L_i}, & \text{if } x_k^{(i)} - \frac{\alpha+\gamma}{L_i} > 0, \\ -\frac{\alpha-\gamma}{L_i}, & \text{if } x_k^{(i)} - \frac{\alpha-\gamma}{L_i} < 0, \\ -x_k^{(i)}, & \text{otherwise,} \end{cases}$$

$x_{k+1} = x_k + e_i T^{(i)}(x_k)$,

$g_{k+1} = g_k + a_i T^{(i)}(x_k)$.

end for

Instance 1: $A \in \mathbf{R}^{10^7 \times 10^6}$, $\|A\|_0 = 10^8$

k/n	$F(x_k) - F^*$	$\|x_k\|_0$	time [s]
0.0010	$< 10^{16}$	857	0.01
12.72	$< 10^{11}$	999,246	54.48
15.23	$< 10^{10}$	997,944	65.19
18.61	$< 10^{9}$	990,923	79.69
20.61	$< 10^{8}$	978,761	88.25
23.38	$< 10^{7}$	926,167	100.08
25.91	$< 10^{6}$	763,314	110.94
28.22	$< 10^{5}$	366,835	120.84
30.66	$< 10^{4}$	57,991	131.25
32.83	$< 10^{3}$	9,701	140.55
35.05	$< 10^{2}$	2,538	150.02
37.01	$< 10^{1}$	1,722	158.39
38.26	$< 10^{0}$	1,633	163.75
40.98	$< 10^{-1}$	1,604	175.38
42.71	$< 10^{-3}$	1,600	182.75
42.71	$< 10^{-4}$	1,600	182.77
43.28	$< 10^{-5}$	1,600	185.21
44.86	$< 10^{-6}$	1,600	191.94

Instance 2: $A \in \mathbf{R}^{10^9 \times 10^8}$, $\|A\|_0 = 2 \times 10^9$

k/n	$F(x_k) - F^*$	$\|x_k\|_0$	time [s]
0.01	$< 10^{18}$	18,486	1.32
9.35	$< 10^{14}$	99,837,255	1294.72
11.97	$< 10^{13}$	99,567,891	1657.32
14.78	$< 10^{12}$	98,630,735	2045.53
17.12	$< 10^{11}$	96,305,090	2370.07
20.09	$< 10^{10}$	86,242,708	2781.11
22.60	$< 10^{9}$	58,157,883	3128.49
24.97	$< 10^{8}$	19,926,459	3455.80
28.62	$< 10^{7}$	747,104	3960.96
31.47	$< 10^{6}$	266,180	4325.60
34.47	$< 10^{5}$	175,981	4693.44
36.84	$< 10^{4}$	163,297	5004.24
39.39	$< 10^{3}$	160,516	5347.71
41.08	$< 10^{2}$	160,138	5577.22
43.88	$< 10^{1}$	160,011	5941.72
45.94	$< 10^{0}$	160,002	6218.82
46.19	$< 10^{-1}$	160,001	6252.20
46.25	$< 10^{-2}$	160,000	6260.20
46.89	$< 10^{-3}$	160,000	6344.31
46.91	$< 10^{-4}$	160,000	6346.99
46.93	$< 10^{-5}$	160,000	6349.69

Instance 3: $A \in \mathbf{R}^{10^{10} \times 10^9}$, $\|A\|_0 = 2 \times 10^{10}$

k/n	$\frac{F(x_k)-F^*}{F(x_0)-F^*}$	$\|x_k\|_0$	time [hours]
0	10^0	0	0.00
1	10^{-1}	14,923,993	1.43
3	10^{-2}	22,688,665	4.25
16	10^{-3}	24,090,068	22.65

Speedup via q-shrinking

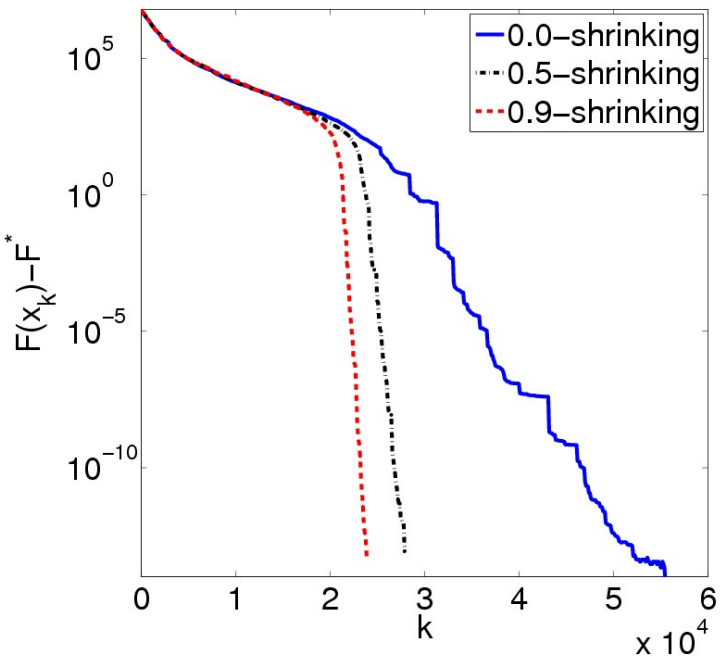
Instead of uniform probabilities, we use at step k

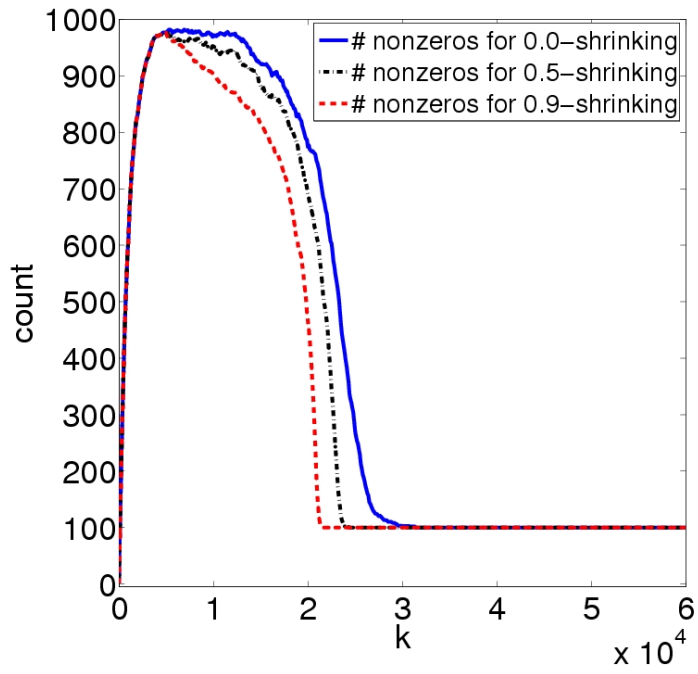
$$p_k^{(i)} = \begin{cases} \frac{1-q}{n}, & \text{if } x_k^{(i)} = 0, \\ \frac{1-q}{n} + \frac{q}{\|x_k\|_0}, & \text{otherwise.} \end{cases}$$

Speedup via q-shrinking

Instead of uniform probabilities, we use at step k

$$p_k^{(i)} = \begin{cases} \frac{1-q}{n}, & \text{if } x_k^{(i)} = 0, \\ \frac{1-q}{n} + \frac{q}{\|x_k\|_0}, & \text{otherwise.} \end{cases}$$





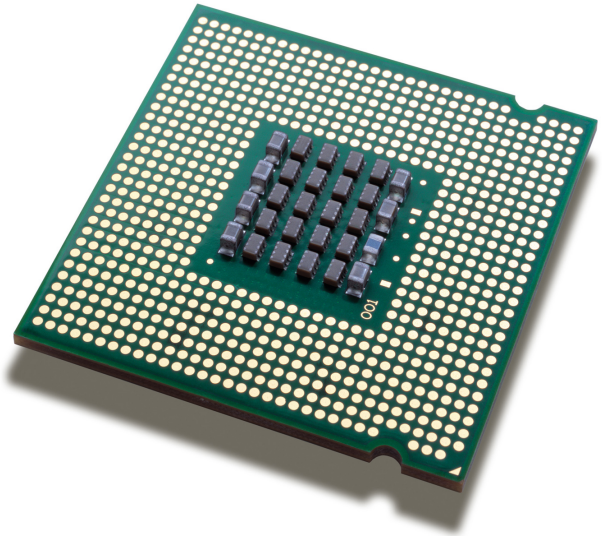
Compare/complement with: L. El Ghaoui, V. Viallon, T. Rabbani. Safe Feature Elimination in Sparse Supervised Learning, Technical Report No. UCB/EECS-2010-126, September 2010.

Part VII.

Acceleration on a Graphical Processing Unit (GPU)

CPU vs. GPU

CPU



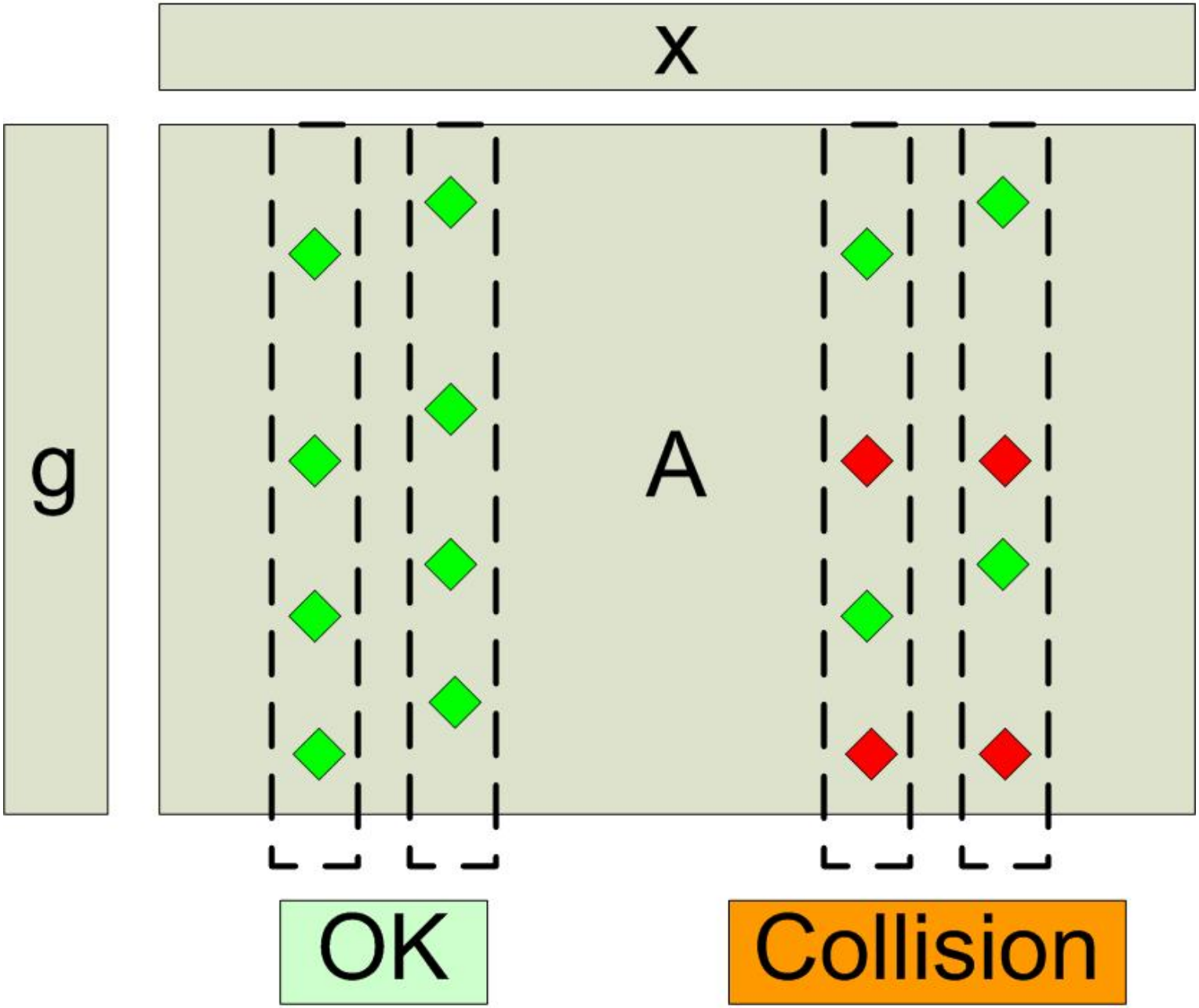
1 core
3.00 GHz
3 GFlops

GPU

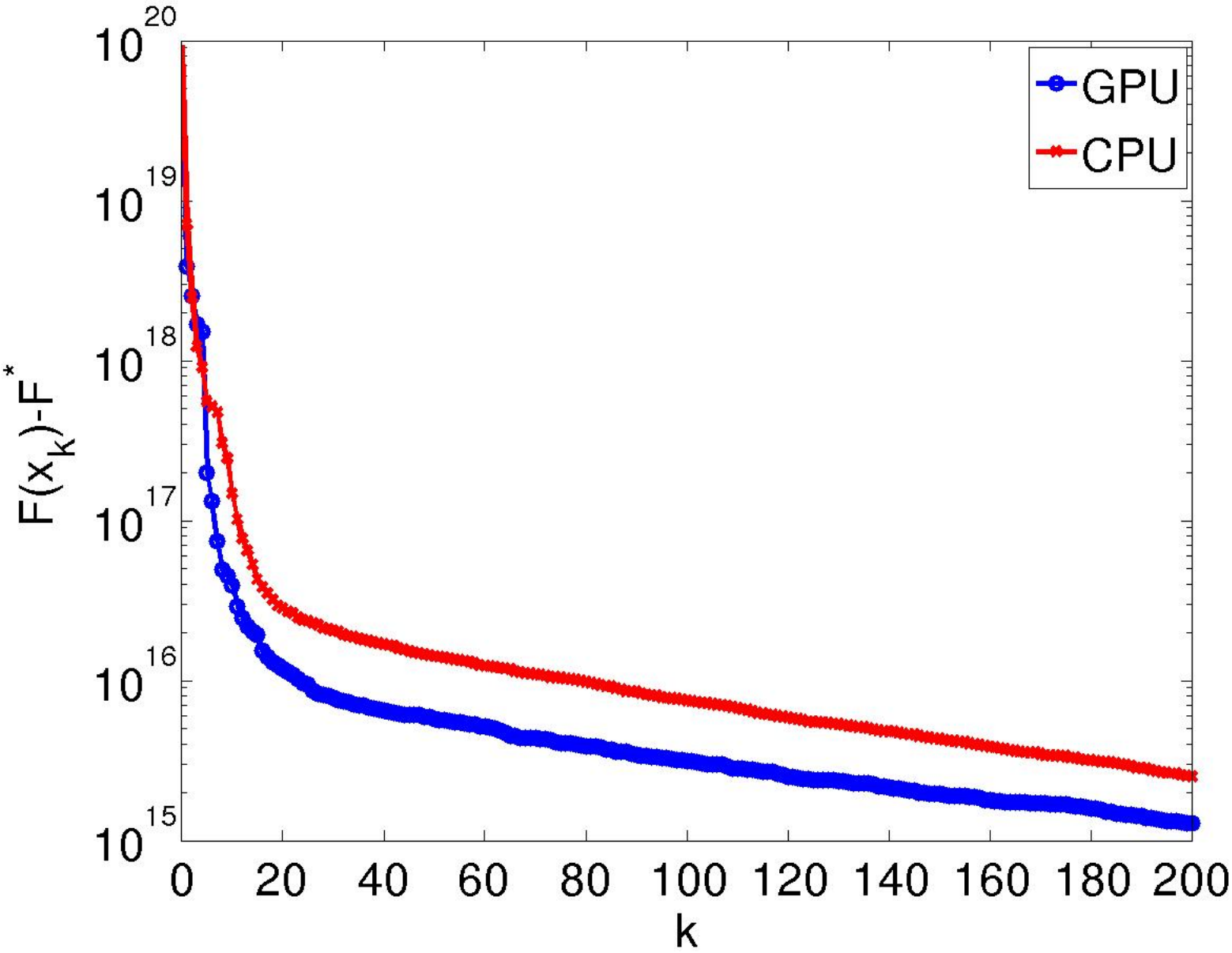
448 cores
1.15 GHz
1 TFlop

GPU (Graphical Processing Unit): performs a single operation in parallel on multiple data (vector addition).

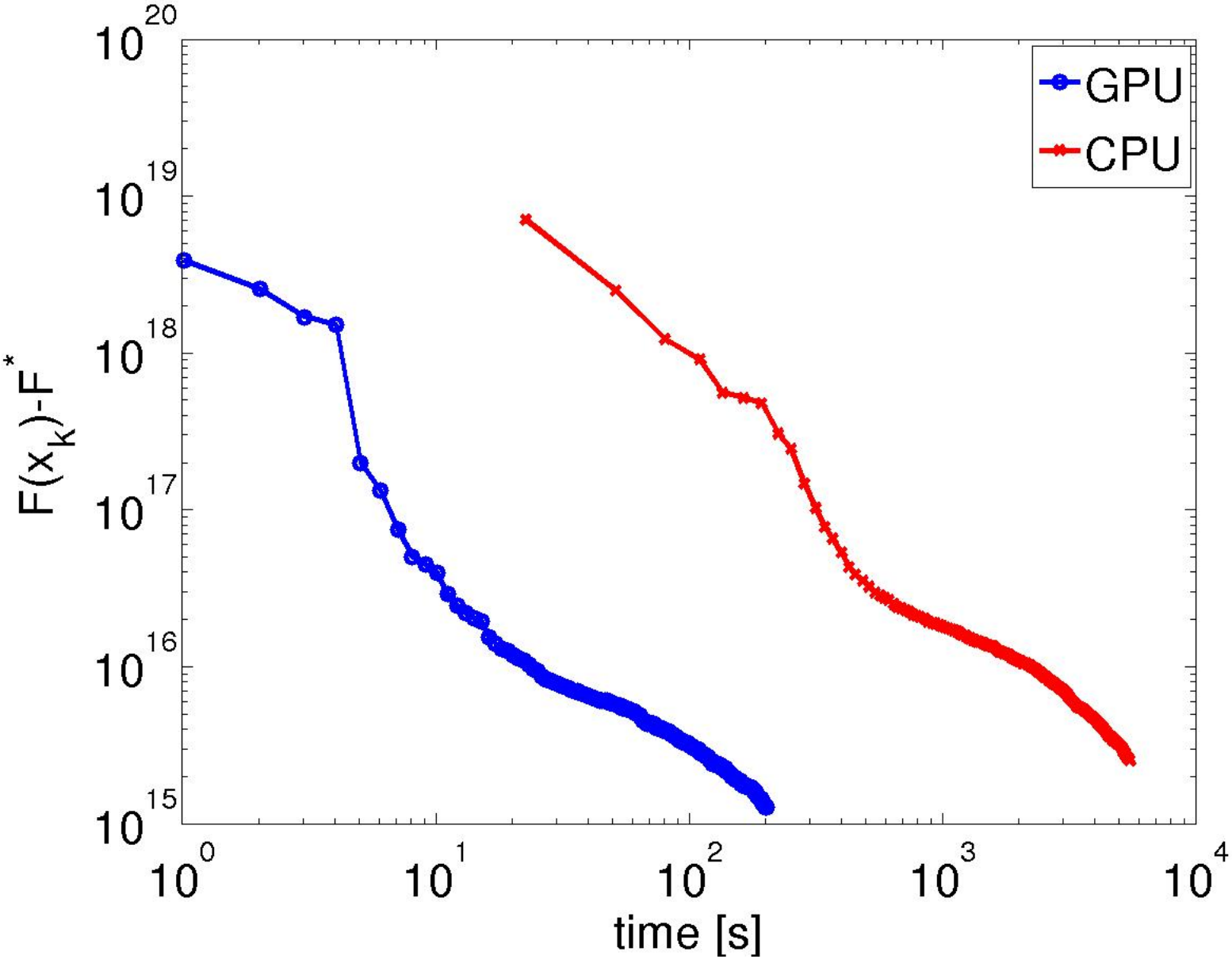
Huge and sparse data = small chance of collision



CPU (C) vs GPU (CUDA): step-by-step comparison

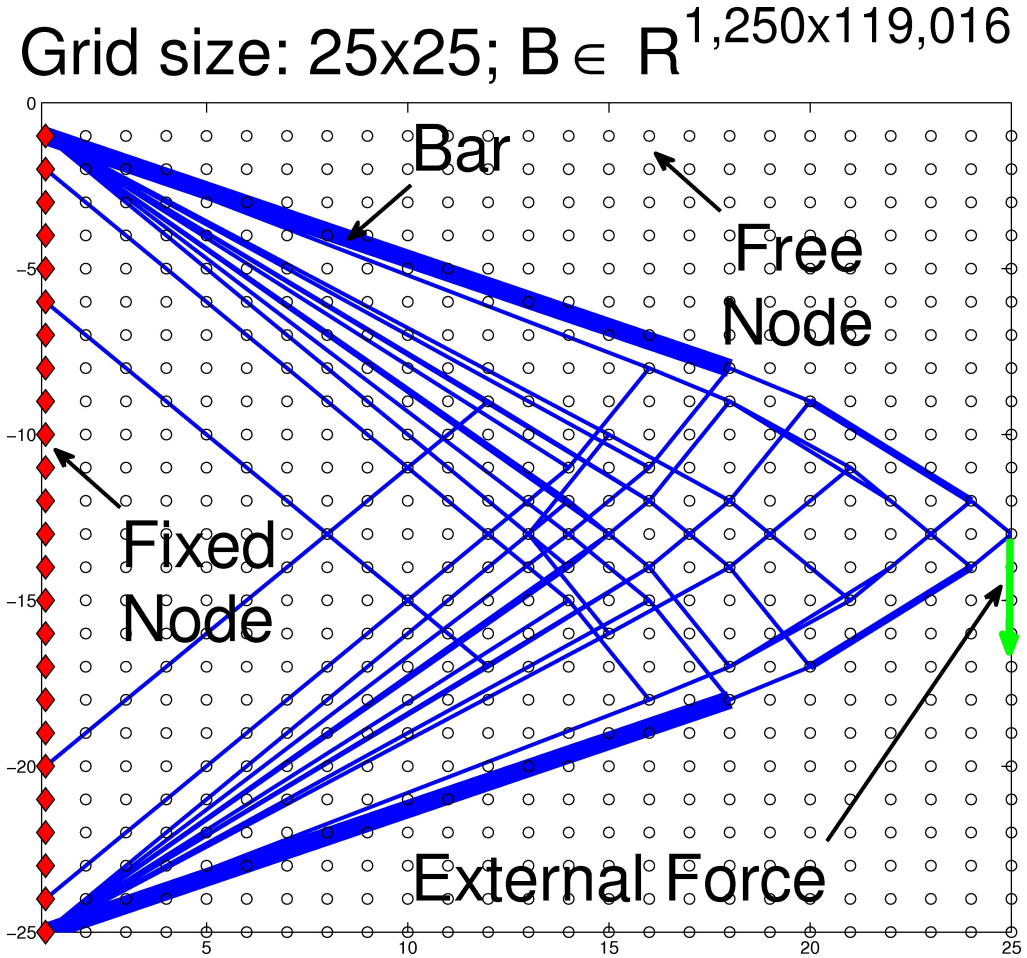


Acceleration by GPU

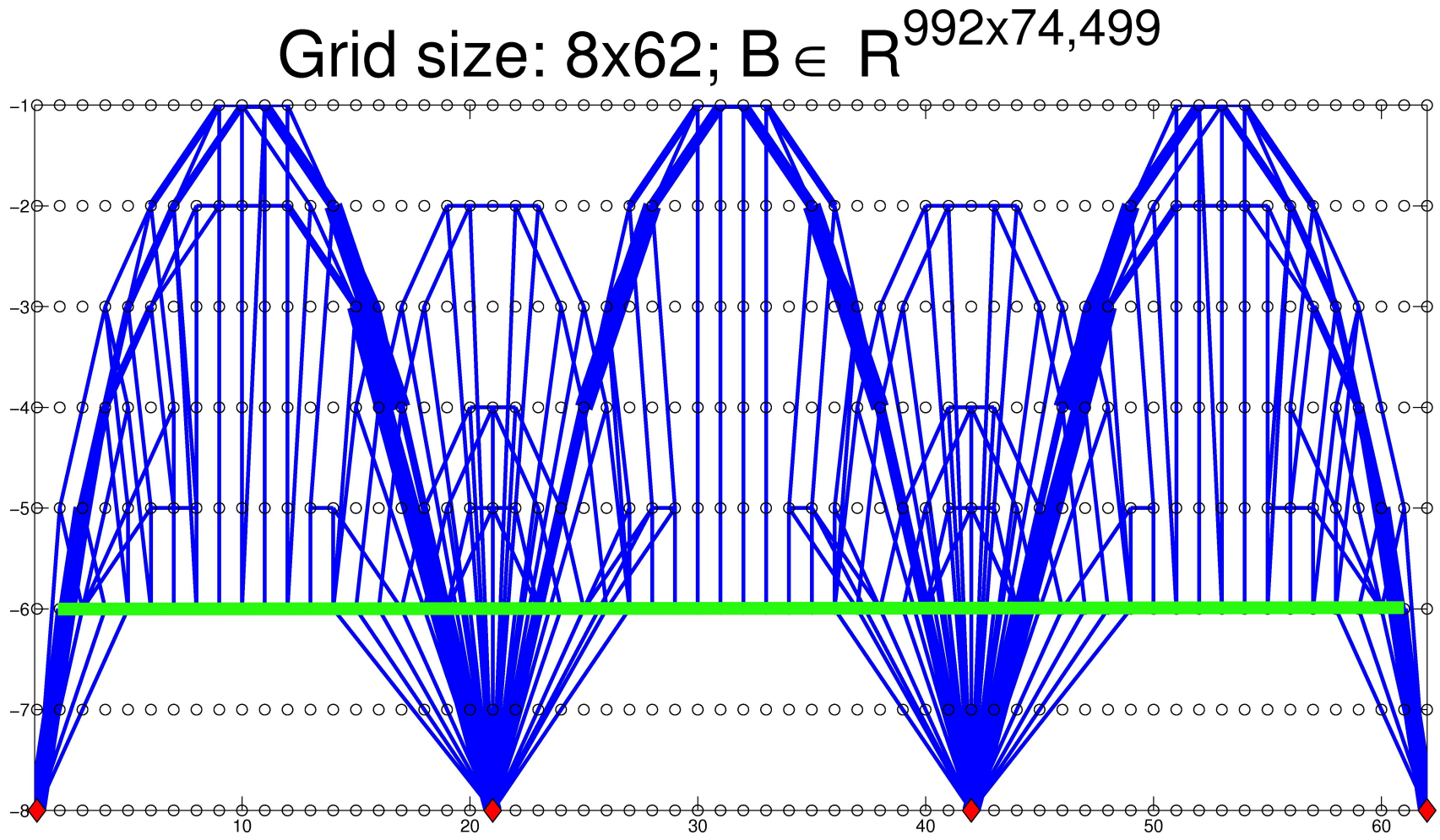


Truss Topology Design: The problem

$$f(x) = \frac{\gamma}{2}\|Bx - d\|_2^2, \quad \Psi(x) = \|x\|_1$$



Truss Topology Design: A Bridge



Truss Topology Design: Performance Test

$r \times c$	$2m$	N	$\|B\|_0 = 4n$
20x20	800	48,934	195,736
30x30	1800	246,690	986,760
40x40	3200	779,074	3,116,296
50x50	5000	1,901,930	7,607,720
80x80	12800	12,454,678	49,818,712
100x100	20000	30,398,894	121,595,576

$r \times c$	time (iter.) SeDuMi	it/sec SG	accel PG	it/sec SR	accel PR	it/sec NEST
20x20	61 (40)	5,101.13	0.62	2.4M	28.29	53.05
30x30	658 (10) NP	1,194.61	2.65	2.2M	26.13	9.01
40x40	8.6k (32) NP	380.36	7.95	1.9M	24.83	3.43
50x50	48.4k (4) NP	159.13	15.98	1.5M	27.15	1.41
80x80	X	25.63	42.34	1.5M	34.01	0.15
100x100	X	10.66	52.18	1.2M	30.86	0.05

Algorithms: **S**erial **G**reedy, **P**arallel **G**reedy, **S**erial **R**andom, **P**arallel **R**andom. For SeDuMi: NP = Numerical Problems, X = no iteration after 10 hours. NEST = Nesterov's accelerated method.