Primal-dual subgradient methods for huge-scale problems

Yurii Nesterov, CORE/INMA (UCL)

July 10, 2013 (ROKS, Heverlee)

2nd part: joint work with S.Shpirko (IITP, Moscow)

Outline

Nonlinear Optimization: problems sizes

Class	Operations	Dimension	Iter.Cost	Memory	
Small-size	All	$10^0 - 10^2$	$n^4 \to n^3$	Kilobyte:	10^3
Medium-size	A^{-1}	$10^3 - 10^4$	$n^3 \to n^2$	Megabyte:	10^6

Nonlinear Optimization: problems sizes

Class	Operations	Dimension	Iter.Cost	Memory	
Small-size	All	$10^0 - 10^2$	$n^4 \to n^3$	Kilobyte:	10^3
Medium-size	A^{-1}	$10^3 - 10^4$	$n^3 \to n^2$	Megabyte:	10^6
Large-scale	Ax	$10^5 - 10^7$	$n^2 \to n$	Gigabyte:	10^9

Nonlinear Optimization: problems sizes

Class	Operations	Dimension	Iter.Cost	Memory	
Small-size	All	$10^0 - 10^2$	$n^4 \to n^3$	Kilobyte:	10^3
Medium-size	A^{-1}	$10^3 - 10^4$	$n^3 \to n^2$	Megabyte:	10^6
Large-scale	Ax	$10^5 - 10^7$	$n^2 \to n$	Gigabyte:	10^9
Huge-scale	$x + y$	$10^8 - 10^{12}$	$n \to \log n$	Terabyte:	10^{12}

Nonlinear Optimization: problems sizes

Class	Operations	Dimension	Iter.Cost	Memory	
Small-size	All	$10^0 - 10^2$	$n^4 \to n^3$	Kilobyte:	10^3
Medium-size	A^{-1}	$10^3 - 10^4$	$n^3 \to n^2$	Megabyte:	10^6
Large-scale	Ax	$10^5 - 10^7$	$n^2 \to n$	Gigabyte:	10^9
Huge-scale	$x + y$	$10^8 - 10^{12}$	$n \to \log n$	Terabyte:	10^{12}

Sources of Huge-Scale problems

Nonlinear Optimization: problems sizes

Class	Operations	Dimension	Iter.Cost	Memory	
Small-size	All	$10^0 - 10^2$	$n^4 \to n^3$	Kilobyte:	10^3
Medium-size	A^{-1}	$10^3 - 10^4$	$n^3 \to n^2$	Megabyte:	10^6
Large-scale	Ax	$10^5 - 10^7$	$n^2 \to n$	Gigabyte:	10^9
Huge-scale	$x + y$	$10^8 - 10^{12}$	$n \to \log n$	Terabyte:	10^{12}

Sources of Huge-Scale problems

- Internet (New)
- Telecommunications (New)

Nonlinear Optimization: problems sizes

Class	Operations	Dimension	Iter.Cost	Memory	
Small-size	All	$10^0 - 10^2$	$n^4 \to n^3$	Kilobyte:	10^3
Medium-size	A^{-1}	$10^3 - 10^4$	$n^3 \to n^2$	Megabyte:	10^6
Large-scale	Ax	$10^5 - 10^7$	$n^2 \to n$	Gigabyte:	10^9
Huge-scale	$x + y$	$10^8 - 10^{12}$	$n \to \log n$	Terabyte:	10^{12}

Sources of Huge-Scale problems

- Internet (New)
- Telecommunications (New)
- Finite-element schemes (Old)
- PDE, Weather prediction (Old)

Nonlinear Optimization: problems sizes

Class	Operations	Dimension	Iter.Cost	Memory	
Small-size	All	$10^0 - 10^2$	$n^4 \to n^3$	Kilobyte:	10^3
Medium-size	A^{-1}	$10^3 - 10^4$	$n^3 \to n^2$	Megabyte:	10^6
Large-scale	Ax	$10^5 - 10^7$	$n^2 \to n$	Gigabyte:	10^9
Huge-scale	$x + y$	$10^8 - 10^{12}$	$n \to \log n$	Terabyte:	10^{12}

Sources of Huge-Scale problems

- Internet (New)
- Telecommunications (New)
- Finite-element schemes (Old)
- PDE, Weather prediction (Old)

Main hope: Sparsity.

Sparse problems

Sparse problems

Problem: $\min_{x \in Q} f(x)$, where Q is closed and convex in R^N,

Sparse problems

Problem: $\min\limits_{x \in Q} f(x)$, where Q is closed and convex in R^N, and

- $f(x) = \Psi(Ax)$, where Ψ is a simple *convex function*:

$$\Psi(y_1) \geq \Psi(y_2) + \langle \Psi'(y_2), y_1 - y_2 \rangle, \quad y_1, y_2 \in R^M,$$

Sparse problems

Problem: $\min\limits_{x \in Q} f(x)$, where Q is closed and convex in R^N, and

- $f(x) = \Psi(Ax)$, where Ψ is a simple *convex function*:

$$\Psi(y_1) \geq \Psi(y_2) + \langle \Psi'(y_2), y_1 - y_2 \rangle, \quad y_1, y_2 \in R^M,$$

- $A : R^N \to R^M$ is a *sparse* matrix.

Sparse problems

Problem: $\min\limits_{x \in Q} f(x)$, where Q is closed and convex in R^N, and

- $f(x) = \Psi(Ax)$, where Ψ is a simple *convex function*:

$$\Psi(y_1) \geq \Psi(y_2) + \langle \Psi'(y_2), y_1 - y_2 \rangle, \quad y_1, y_2 \in R^M,$$

- $A : R^N \to R^M$ is a *sparse* matrix.

Let $p(x) = \#$ of nonzeros in x.

Sparse problems

Problem: $\min\limits_{x \in Q} f(x)$, where Q is closed and convex in R^N, and

- $f(x) = \Psi(Ax)$, where Ψ is a simple *convex function*:

$$\Psi(y_1) \geq \Psi(y_2) + \langle \Psi'(y_2), y_1 - y_2 \rangle, \quad y_1, y_2 \in R^M,$$

- $A : R^N \to R^M$ is a *sparse* matrix.

Let $p(x) = \#$ of nonzeros in x. Sparsity coefficient: $\gamma(A) \stackrel{\mathrm{def}}{=} \frac{p(A)}{MN}$.

Sparse problems

Problem: $\min\limits_{x \in Q} f(x)$, where Q is closed and convex in R^N, and

- $f(x) = \Psi(Ax)$, where Ψ is a simple *convex function*:

$$\Psi(y_1) \geq \Psi(y_2) + \langle \Psi'(y_2), y_1 - y_2 \rangle, \quad y_1, y_2 \in R^M,$$

- $A : R^N \to R^M$ is a *sparse* matrix.

Let $p(x) = \#$ of nonzeros in x. Sparsity coefficient: $\gamma(A) \stackrel{\mathrm{def}}{=} \frac{p(A)}{MN}$.

Example 1: Matrix-vector multiplication

Sparse problems

Problem: $\min\limits_{x \in Q} f(x)$, where Q is closed and convex in R^N, and

- $f(x) = \Psi(Ax)$, where Ψ is a simple *convex function*:

$$\Psi(y_1) \geq \Psi(y_2) + \langle \Psi'(y_2), y_1 - y_2 \rangle, \quad y_1, y_2 \in R^M,$$

- $A : R^N \to R^M$ is a *sparse* matrix.

Let $p(x) = \#$ of nonzeros in x. Sparsity coefficient: $\gamma(A) \stackrel{\mathrm{def}}{=} \frac{p(A)}{MN}$.

Example 1: Matrix-vector multiplication

- Computation of vector Ax needs $p(A)$ operations.

Sparse problems

Problem: $\min\limits_{x \in Q} f(x)$, where Q is closed and convex in R^N, and

- $f(x) = \Psi(Ax)$, where Ψ is a simple *convex function*:

$$\Psi(y_1) \geq \Psi(y_2) + \langle \Psi'(y_2), y_1 - y_2 \rangle, \quad y_1, y_2 \in R^M,$$

- $A : R^N \to R^M$ is a *sparse* matrix.

Let $p(x) = \#$ of nonzeros in x. Sparsity coefficient: $\gamma(A) \stackrel{\mathrm{def}}{=} \frac{p(A)}{MN}$.

Example 1: Matrix-vector multiplication

- Computation of vector Ax needs $p(A)$ operations.
- Initial complexity MN is reduced in $\gamma(A)$ times.

Example: Gradient Method

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

- Projection of simple set Q needs $O(N)$ operations.

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

- Projection of simple set Q needs $O(N)$ operations.
- Displacement $x_k \to x_k - hf'(x_k)$ needs $O(N)$ operations.

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

- Projection of simple set Q needs $O(N)$ operations.
- Displacement $x_k \to x_k - hf'(x_k)$ needs $O(N)$ operations.
- $f'(x) = A^T \Psi'(Ax)$.

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

- Projection of simple set Q needs $O(N)$ operations.
- Displacement $x_k \to x_k - hf'(x_k)$ needs $O(N)$ operations.
- $f'(x) = A^T \Psi'(Ax)$. If Ψ is simple, then the main efforts are spent for two matrix-vector multiplications: $2p(A)$.

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

- Projection of simple set Q needs $O(N)$ operations.
- Displacement $x_k \to x_k - hf'(x_k)$ needs $O(N)$ operations.
- $f'(x) = A^T \Psi'(Ax)$. If Ψ is simple, then the main efforts are spent for two matrix-vector multiplications: $2p(A)$.

Conclusion:

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

- Projection of simple set Q needs $O(N)$ operations.
- Displacement $x_k \to x_k - hf'(x_k)$ needs $O(N)$ operations.
- $f'(x) = A^T \Psi'(Ax)$. If Ψ is simple, then the main efforts are spent for two matrix-vector multiplications: $2p(A)$.

Conclusion: As compared with *full* matrices, we accelerate in $\gamma(A)$ times.

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

- Projection of simple set Q needs $O(N)$ operations.
- Displacement $x_k \to x_k - hf'(x_k)$ needs $O(N)$ operations.
- $f'(x) = A^T \Psi'(Ax)$. If Ψ is simple, then the main efforts are spent for two matrix-vector multiplications: $2p(A)$.

Conclusion: As compared with *full* matrices, we accelerate in $\gamma(A)$ times.

Note: For Large- and Huge-scale problems, we often have

$$\gamma(A) \approx 10^{-4} \ldots 10^{-6}.$$

Example: Gradient Method

$$x_0 \in Q, \quad x_{k+1} = \pi_Q(x_k - hf'(x_k)), \quad k \geq 0.$$

Main computational expenses

- Projection of simple set Q needs $O(N)$ operations.
- Displacement $x_k \to x_k - hf'(x_k)$ needs $O(N)$ operations.
- $f'(x) = A^T \Psi'(Ax)$. If Ψ is simple, then the main efforts are spent for two matrix-vector multiplications: $2p(A)$.

Conclusion: As compared with *full* matrices, we accelerate in $\gamma(A)$ times.

Note: For Large- and Huge-scale problems, we often have

$$\gamma(A) \approx 10^{-4} \ldots 10^{-6}.$$

Can we get more?

Sparse updating strategy

Sparse updating strategy

Main idea

Sparse updating strategy

Main idea

- After update $x_+ = x + d$

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+$

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$.

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\mathrm{def}}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j)$,

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\mathrm{def}}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\mathrm{def}}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\kappa_A(d) \;=\; M \sum_{j \in \sigma(d)} \gamma(Ae_j)$$

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\mathrm{def}}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\kappa_A(d) \;=\; M \sum_{j \in \sigma(d)} \gamma(Ae_j) \;=\; \gamma(d) \cdot \frac{1}{p(d)} \sum_{j \in \sigma(d)} \gamma(Ae_j) \cdot MN$$

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\mathrm{def}}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\begin{aligned} \kappa_A(d) &= M \sum_{j \in \sigma(d)} \gamma(Ae_j) = \gamma(d) \cdot \tfrac{1}{p(d)} \sum_{j \in \sigma(d)} \gamma(Ae_j) \cdot MN \\ &\leq \gamma(d) \max_{1 \leq j \leq m} \gamma(Ae_j) \cdot MN. \end{aligned}$$

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\rm def}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\rm def}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\begin{aligned} \kappa_A(d) &= M \sum_{j \in \sigma(d)} \gamma(Ae_j) = \gamma(d) \cdot \tfrac{1}{p(d)} \sum_{j \in \sigma(d)} \gamma(Ae_j) \cdot MN \\ &\leq \gamma(d) \max_{1 \leq j \leq m} \gamma(Ae_j) \cdot MN. \end{aligned}$$

If $\gamma(d) \leq c\gamma(A)$, $\gamma(A_j) \leq c\gamma(A)$,

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\rm def}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\rm def}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\begin{aligned} \kappa_A(d) &= M \sum_{j \in \sigma(d)} \gamma(Ae_j) = \gamma(d) \cdot \tfrac{1}{p(d)} \sum_{j \in \sigma(d)} \gamma(Ae_j) \cdot MN \\ &\leq \gamma(d) \max_{1 \leq j \leq m} \gamma(Ae_j) \cdot MN. \end{aligned}$$

If $\gamma(d) \leq c\gamma(A)$, $\gamma(A_j) \leq c\gamma(A)$, $\Rightarrow$ $\boxed{\kappa_A(d) \leq c^2 \cdot \gamma^2(A) \cdot MN}$

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\rm def}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\rm def}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\begin{aligned} \kappa_A(d) &= M \sum_{j \in \sigma(d)} \gamma(Ae_j) = \gamma(d) \cdot \tfrac{1}{p(d)} \sum_{j \in \sigma(d)} \gamma(Ae_j) \cdot MN \\ &\leq \gamma(d) \max_{1 \leq j \leq m} \gamma(Ae_j) \cdot MN. \end{aligned}$$

If $\gamma(d) \leq c\gamma(A)$, $\gamma(A_j) \leq c\gamma(A)$, $\Rightarrow$ $\boxed{\kappa_A(d) \leq c^2 \cdot \gamma^2(A) \cdot MN}$

Expected acceleration:

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\rm def}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\rm def}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\begin{aligned} \kappa_A(d) &= M \sum_{j \in \sigma(d)} \gamma(Ae_j) = \gamma(d) \cdot \tfrac{1}{p(d)} \sum_{j \in \sigma(d)} \gamma(Ae_j) \cdot MN \\ &\leq \gamma(d) \max_{1 \leq j \leq m} \gamma(Ae_j) \cdot MN. \end{aligned}$$

If $\gamma(d) \leq c\gamma(A)$, $\gamma(A_j) \leq c\gamma(A)$, $\Rightarrow$ $\boxed{\kappa_A(d) \leq c^2 \cdot \gamma^2(A) \cdot MN}$

Expected acceleration: $(10^{-6})^2 = 10^{-12}$

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\rm def}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\rm def}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\begin{aligned} \kappa_A(d) &= M \sum_{j \in \sigma(d)} \gamma(Ae_j) = \gamma(d) \cdot \tfrac{1}{p(d)} \sum_{j \in \sigma(d)} \gamma(Ae_j) \cdot MN \\ &\leq \gamma(d) \max_{1 \leq j \leq m} \gamma(Ae_j) \cdot MN. \end{aligned}$$

If $\gamma(d) \leq c\gamma(A)$, $\gamma(A_j) \leq c\gamma(A)$, $\Rightarrow$ $\boxed{\kappa_A(d) \leq c^2 \cdot \gamma^2(A) \cdot MN}$

Expected acceleration: $(10^{-6})^2 = 10^{-12} \Rightarrow 1$ sec

Sparse updating strategy

Main idea

- After update $x_+ = x + d$ we have $y_+ \stackrel{\mathrm{def}}{=} Ax_+ = \underbrace{Ax}_{y} + Ad$.
- What happens if d is *sparse*?

Denote $\sigma(d) = \{j : d^{(j)} \neq 0\}$. Then $y_+ = y + \sum\limits_{j \in \sigma(d)} d^{(j)} \cdot Ae_j$.

Its complexity, $\quad \kappa_A(d) \stackrel{\mathrm{def}}{=} \sum\limits_{j \in \sigma(d)} p(Ae_j), \quad$ can be VERY small!

$$\begin{aligned} \kappa_A(d) &= M \sum_{j \in \sigma(d)} \gamma(Ae_j) = \gamma(d) \cdot \tfrac{1}{p(d)} \sum_{j \in \sigma(d)} \gamma(Ae_j) \cdot MN \\ &\leq \gamma(d) \max_{1 \leq j \leq m} \gamma(Ae_j) \cdot MN. \end{aligned}$$

If $\gamma(d) \leq c\gamma(A)$, $\gamma(A_j) \leq c\gamma(A)$, $\Rightarrow$ $\boxed{\kappa_A(d) \leq c^2 \cdot \gamma^2(A) \cdot MN}$

Expected acceleration: $(10^{-6})^2 = 10^{-12} \Rightarrow 1$ sec $\approx 32\,000$ years!

When it can work?

When it can work?

- Simple methods:

When it can work?

- Simple methods: No full-vector operations!

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems:

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

1. Quadratic function $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$.

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

1. Quadratic function $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$. The gradient
$$f'(x) = Ax - b, \quad x \in R^N,$$

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

1. Quadratic function $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$. The gradient
$$f'(x) = Ax - b, \quad x \in R^N,$$
is *not* sparse even if A is sparse.

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

1. Quadratic function $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$. The gradient
$$f'(x) = Ax - b, \quad x \in R^N,$$
is *not* sparse even if A is sparse.
2. Piece-wise linear function $g(x) = \max\limits_{1 \le i \le m} [\langle a_i, x\rangle - b^{(i)}]$.

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

1. Quadratic function $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$. The gradient
$$f'(x) = Ax - b, \quad x \in R^N,$$
is *not* sparse even if A is sparse.
2. Piece-wise linear function $g(x) = \max\limits_{1 \le i \le m} [\langle a_i, x\rangle - b^{(i)}]$. Its *subgradient* $f'(x) = a_{i(x)}$, $i(x) : \; f(x) = \langle a_{i(x)}, x\rangle - b^{(i(x))}$,

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

1. Quadratic function $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$. The gradient
$$f'(x) = Ax - b, \quad x \in R^N,$$
is *not* sparse even if A is sparse.
2. Piece-wise linear function $g(x) = \max\limits_{1 \le i \le m} [\langle a_i, x\rangle - b^{(i)}]$. Its *subgradient* $f'(x) = a_{i(x)}$, $i(x) : \; f(x) = \langle a_{i(x)}, x\rangle - b^{(i(x))}$, can be sparse is a_i is sparse!

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

1. Quadratic function $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$. The gradient
$$f'(x) = Ax - b, \quad x \in R^N,$$
is *not* sparse even if A is sparse.
2. Piece-wise linear function $g(x) = \max\limits_{1 \le i \le m} [\langle a_i, x\rangle - b^{(i)}]$. Its *subgradient* $f'(x) = a_{i(x)}$, $i(x) : \; f(x) = \langle a_{i(x)}, x\rangle - b^{(i(x))}$, can be sparse is a_i is sparse!

But:

When it can work?

- Simple methods: No full-vector operations! (Is it possible?)
- Simple problems: Functions with *sparse* gradients.

Let us try:

1. Quadratic function $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$. The gradient
$$f'(x) = Ax - b, \quad x \in R^N,$$
is *not* sparse even if A is sparse.
2. Piece-wise linear function $g(x) = \max\limits_{1 \le i \le m} [\langle a_i, x\rangle - b^{(i)}]$. Its *subgradient* $f'(x) = a_{i(x)}$, $i(x) : \; f(x) = \langle a_{i(x)}, x\rangle - b^{(i(x))}$, can be sparse is a_i is sparse!

But: We need a fast procedure for updating *max-type operations*.

Fast updates in short computational trees

Fast updates in short computational trees

Def: Function $f(x)$, $x \in R^n$, is *short-tree representable*, if it can be computed by a short binary tree with the height $\approx \ln n$.

Fast updates in short computational trees

Def: Function $f(x)$, $x \in R^n$, is *short-tree representable*, if it can be computed by a short binary tree with the height $\approx \ln n$.

Let $n = 2^k$ and the tree has $k+1$ levels: $v_{0,i} = x^{(i)}$, $i = 1, \ldots, n$.

Fast updates in short computational trees

Def: Function $f(x)$, $x \in R^n$, is *short-tree representable*, if it can be computed by a short binary tree with the height $\approx \ln n$.

Let $n = 2^k$ and the tree has $k+1$ levels: $v_{0,i} = x^{(i)}$, $i = 1, \ldots, n$. Size of the next level halves the size of the previous one:

$$v_{i+1,j} \quad = \quad \psi_{i+1,j}(v_{i,2j-1}, v_{i,2j}), \quad j = 1, \ldots, 2^{k-i-1}, \; i = 0, \ldots, k-1,$$

where $\psi_{i,j}$ are some bivariate functions.

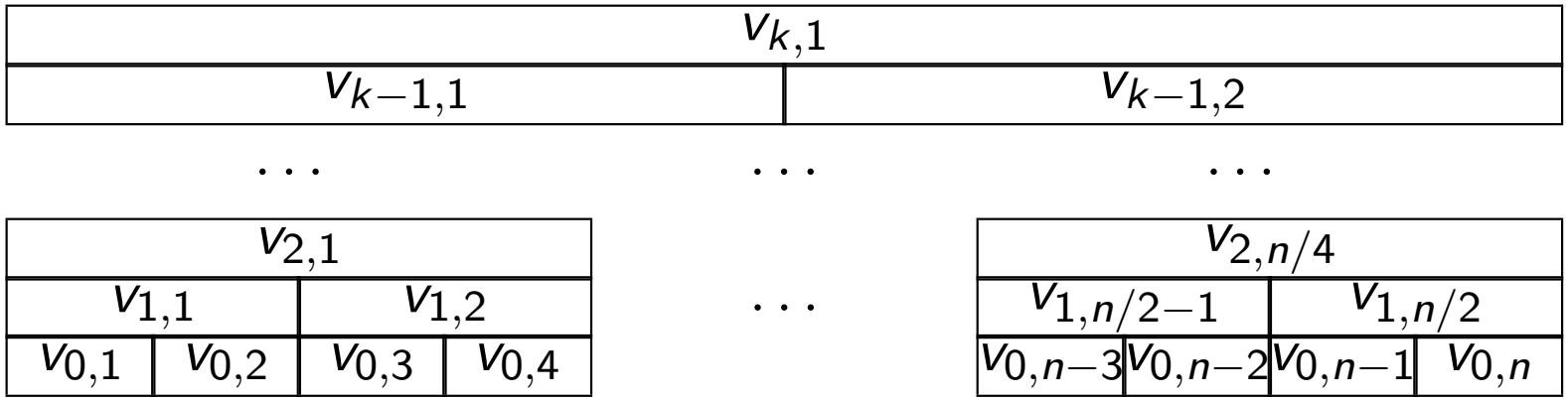
Fast updates in short computational trees

Def: Function $f(x)$, $x \in R^n$, is *short-tree representable*, if it can be computed by a short binary tree with the height $\approx \ln n$.

Let $n = 2^k$ and the tree has $k+1$ levels: $v_{0,i} = x^{(i)}$, $i = 1, \ldots, n$. Size of the next level halves the size of the previous one:

$$v_{i+1,j} \quad = \quad \psi_{i+1,j}(v_{i,2j-1}, v_{i,2j}), \quad j = 1, \ldots, 2^{k-i-1}, \; i = 0, \ldots, k-1,$$

where $\psi_{i,j}$ are some bivariate functions.



Main advantages

Main advantages

- Important examples (symmetric functions)

Main advantages

- Important examples (symmetric functions)

$$f(x) \;=\; \|x\|_p, \quad p \geq 1, \quad \psi_{i,j}(t_1, t_2) \;\equiv\; [\, |t_1|^p + |t_2|^p \,]^{1/p},$$

Main advantages

- Important examples (symmetric functions)

$$
\begin{aligned}
f(x) &= \|x\|_p, \quad p \geq 1, \quad \psi_{i,j}(t_1, t_2) \equiv [\, |t_1|^p + |t_2|^p \,]^{1/p}, \\
f(x) &= \ln\left(\sum_{i=1}^{n} e^{x^{(i)}}\right), \quad \psi_{i,j}(t_1, t_2) \equiv \ln\left(e^{t_1} + e^{t_2}\right),
\end{aligned}
$$

Main advantages

- Important examples (symmetric functions)

$$
\begin{array}{rcll}
f(x) & = & \|x\|_p, \quad p \geq 1, & \psi_{i,j}(t_1, t_2) \equiv [\, |t_1|^p + |t_2|^p \,]^{1/p}, \\
f(x) & = & \ln \left(\sum\limits_{i=1}^{n} e^{x^{(i)}} \right), & \psi_{i,j}(t_1, t_2) \equiv \ln \left(e^{t_1} + e^{t_2} \right), \\
f(x) & = & \max\limits_{1 \leq i \leq n} x^{(i)}, & \psi_{i,j}(t_1, t_2) \equiv \max \{ t_1, t_2 \}.
\end{array}
$$

Main advantages

- Important examples (symmetric functions)

$$
\begin{array}{rcll}
f(x) & = & \|x\|_p, \quad p \geq 1, & \psi_{i,j}(t_1, t_2) \equiv [\, |t_1|^p + |t_2|^p \,]^{1/p}, \\
f(x) & = & \ln \left(\sum\limits_{i=1}^{n} e^{x^{(i)}} \right), & \psi_{i,j}(t_1, t_2) \equiv \ln \left(e^{t_1} + e^{t_2} \right), \\
f(x) & = & \max\limits_{1 \leq i \leq n} x^{(i)}, & \psi_{i,j}(t_1, t_2) \equiv \max \{ t_1, t_2 \}.
\end{array}
$$

- The binary tree requires only $n - 1$ auxiliary cells.

Main advantages

- Important examples (symmetric functions)

$$
\begin{array}{rcll}
f(x) & = & \|x\|_p, \quad p \geq 1, & \psi_{i,j}(t_1, t_2) \equiv [\, |t_1|^p + |t_2|^p \,]^{1/p}, \\
f(x) & = & \ln \left(\sum\limits_{i=1}^{n} e^{x^{(i)}} \right), & \psi_{i,j}(t_1, t_2) \equiv \ln \left(e^{t_1} + e^{t_2} \right), \\
f(x) & = & \max\limits_{1 \leq i \leq n} x^{(i)}, & \psi_{i,j}(t_1, t_2) \equiv \max \{ t_1, t_2 \}.
\end{array}
$$

- The binary tree requires only $n - 1$ auxiliary cells.
- Its value needs $n - 1$ applications of $\psi_{i,j}(\cdot, \cdot)$ ($\equiv$ *operations*).

Main advantages

- Important examples (symmetric functions)

$$
\begin{array}{rcll}
f(x) & = & \|x\|_p, \quad p \geq 1, & \psi_{i,j}(t_1, t_2) \equiv [\, |t_1|^p + |t_2|^p \,]^{1/p}, \\
f(x) & = & \ln \left(\sum\limits_{i=1}^{n} e^{x^{(i)}} \right), & \psi_{i,j}(t_1, t_2) \equiv \ln \left(e^{t_1} + e^{t_2} \right), \\
f(x) & = & \max\limits_{1 \leq i \leq n} x^{(i)}, & \psi_{i,j}(t_1, t_2) \equiv \max \{ t_1, t_2 \}.
\end{array}
$$

- The binary tree requires only $n - 1$ auxiliary cells.
- Its value needs $n - 1$ applications of $\psi_{i,j}(\cdot, \cdot)$ ($\equiv$ *operations*).
- If x_+ differs from x in one entry only, then for re-computing $f(x_+)$ we need only $k \equiv \log_2 n$ operations.

Main advantages

- Important examples (symmetric functions)

$$
\begin{array}{rcll}
f(x) & = & \|x\|_p, \quad p \geq 1, & \psi_{i,j}(t_1, t_2) \equiv [\, |t_1|^p + |t_2|^p \,]^{1/p}, \\
f(x) & = & \ln \left(\sum\limits_{i=1}^{n} e^{x^{(i)}} \right), & \psi_{i,j}(t_1, t_2) \equiv \ln \left(e^{t_1} + e^{t_2} \right), \\
f(x) & = & \max\limits_{1 \leq i \leq n} x^{(i)}, & \psi_{i,j}(t_1, t_2) \equiv \max \{ t_1, t_2 \}.
\end{array}
$$

- The binary tree requires only $n - 1$ auxiliary cells.
- Its value needs $n - 1$ applications of $\psi_{i,j}(\cdot, \cdot)$ ($\equiv$ *operations*).
- If x_+ differs from x in one entry only, then for re-computing $f(x_+)$ we need only $k \equiv \log_2 n$ operations.

Thus, we can have pure subgradient minimization schemes with

Main advantages

- Important examples (symmetric functions)

$$
\begin{array}{rcll}
f(x) & = & \|x\|_p, \quad p \geq 1, & \psi_{i,j}(t_1, t_2) \equiv [\, |t_1|^p + |t_2|^p \,]^{1/p}, \\
f(x) & = & \ln \left(\sum\limits_{i=1}^{n} e^{x^{(i)}} \right), & \psi_{i,j}(t_1, t_2) \equiv \ln \left(e^{t_1} + e^{t_2} \right), \\
f(x) & = & \max\limits_{1 \leq i \leq n} x^{(i)}, & \psi_{i,j}(t_1, t_2) \equiv \max \{ t_1, t_2 \}.
\end{array}
$$

- The binary tree requires only $n - 1$ auxiliary cells.
- Its value needs $n - 1$ applications of $\psi_{i,j}(\cdot, \cdot)$ ($\equiv$ *operations*).
- If x_+ differs from x in one entry only, then for re-computing $f(x_+)$ we need only $k \equiv \log_2 n$ operations.

Thus, we can have pure subgradient minimization schemes with

Sublinear Iteration Cost

.

Simple subgradient methods

Simple subgradient methods

I. Problem: $f^* \stackrel{\mathrm{def}}{=} \min\limits_{x \in Q} f(x)$, where

Simple subgradient methods

I. Problem: $f^* \stackrel{\mathrm{def}}{=} \min\limits_{x \in Q} f(x)$, where

- Q is a closed and convex and $\|f'(x)\| \leq L(f)$, $x \in Q$,

Simple subgradient methods

I. Problem: $f^* \stackrel{\mathrm{def}}{=} \min\limits_{x \in Q} f(x)$, where

- Q is a closed and convex and $\|f'(x)\| \leq L(f)$, $x \in Q$,
- the optimal value f^* is known.

Simple subgradient methods

I. Problem: $f^* \stackrel{\mathrm{def}}{=} \min\limits_{x \in Q} f(x)$, where

- Q is a closed and convex and $\|f'(x)\| \leq L(f)$, $x \in Q$,
- the optimal value f^* is known.

Consider the following optimization scheme (B.Polyak, 1967):

Simple subgradient methods

I. Problem: $f^* \stackrel{\mathrm{def}}{=} \min\limits_{x \in Q} f(x)$, where

- Q is a closed and convex and $\|f'(x)\| \leq L(f)$, $x \in Q$,
- the optimal value f^* is known.

Consider the following optimization scheme (B.Polyak, 1967):

$$x_0 \in Q, \quad x_{k+1} = \pi_Q \left(x_k - \frac{f(x_k) - f^*}{\|f'(x_k)\|^2} f'(x_k) \right), \quad k \geq 0.$$

Simple subgradient methods

I. Problem: $f^* \stackrel{\mathrm{def}}{=} \min\limits_{x \in Q} f(x)$, where

- Q is a closed and convex and $\|f'(x)\| \leq L(f)$, $x \in Q$,
- the optimal value f^* is known.

Consider the following optimization scheme (B.Polyak, 1967):

$$x_0 \in Q, \quad x_{k+1} = \pi_Q \left(x_k - \frac{f(x_k) - f^*}{\|f'(x_k)\|^2} f'(x_k) \right), \quad k \geq 0.$$

Denote $f_k^* = \min\limits_{0 \leq i \leq k} f(x_i)$.

Simple subgradient methods

I. Problem: $f^* \stackrel{\mathrm{def}}{=} \min\limits_{x \in Q} f(x)$, where

- Q is a closed and convex and $\|f'(x)\| \leq L(f)$, $x \in Q$,
- the optimal value f^* is known.

Consider the following optimization scheme (B.Polyak, 1967):

$$x_0 \in Q, \quad x_{k+1} = \pi_Q \left(x_k - \frac{f(x_k) - f^*}{\|f'(x_k)\|^2} f'(x_k) \right), \quad k \geq 0.$$

Denote $f_k^* = \min\limits_{0 \leq i \leq k} f(x_i)$. Then for any $k \geq 0$ we have:

$$f_k^* - f^* \leq \frac{L(f)\|x_0 - \pi_{X_*}(x_0)\|}{(k+1)^{1/2}}.$$

Constrained minimization (N.Shor (1964) & B.Polyak)

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$,

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method.

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

If $g(x_k) > h \, \| g'(x_k) \|$, then (A):

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

If $g(x_k) > h \, \| g'(x_k) \|$, then (A): $x_{k+1} = \pi_Q \left(x_k - \frac{g(x_k)}{\| g'(x_k) \|^2} \, g'(x_k) \right)$,

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

$$\text{If } g(x_k) > h \, \|g'(x_k)\|, \quad \text{then (A):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{g(x_k)}{\|g'(x_k)\|^2} \, g'(x_k) \right),$$

$$\text{else (B):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{h}{\|f'(x_k)\|} \, f'(x_k) \right).$$

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

$$\text{If } g(x_k) > h \, \|g'(x_k)\|, \quad \text{then (A):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{g(x_k)}{\|g'(x_k)\|^2} \, g'(x_k) \right),$$

$$\text{else (B):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{h}{\|f'(x_k)\|} \, f'(x_k) \right).$$

Let $\mathcal{F}_k \subseteq \{0, \ldots, k\}$ be the set (B)-iterations, and $f_k^* = \min\limits_{i \in \mathcal{F}_k} f(x_i)$.

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

$$\text{If } g(x_k) > h \, \|g'(x_k)\|, \quad \text{then (A):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{g(x_k)}{\|g'(x_k)\|^2} \, g'(x_k) \right),$$

$$\text{else (B):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{h}{\|f'(x_k)\|} \, f'(x_k) \right).$$

Let $\mathcal{F}_k \subseteq \{0, \ldots, k\}$ be the set (B)-iterations, and $f_k^* = \min\limits_{i \in \mathcal{F}_k} f(x_i)$.

Theorem:

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

$$\text{If } g(x_k) > h \, \|g'(x_k)\|, \quad \text{then (A):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{g(x_k)}{\|g'(x_k)\|^2} \, g'(x_k) \right),$$

$$\text{else (B):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{h}{\|f'(x_k)\|} \, f'(x_k) \right).$$

Let $\mathcal{F}_k \subseteq \{0, \ldots, k\}$ be the set (B)-iterations, and $f_k^* = \min\limits_{i \in \mathcal{F}_k} f(x_i)$.

Theorem: If $k > \|x_0 - x^*\|^2 / h^2$,

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

$$\text{If } g(x_k) > h \, \|g'(x_k)\|, \quad \text{then (A):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{g(x_k)}{\|g'(x_k)\|^2} \, g'(x_k) \right),$$

$$\text{else (B):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{h}{\|f'(x_k)\|} \, f'(x_k) \right).$$

Let $\mathcal{F}_k \subseteq \{0, \ldots, k\}$ be the set (B)-iterations, and $f_k^* = \min\limits_{i \in \mathcal{F}_k} f(x_i)$.

Theorem: If $k > \|x_0 - x^*\|^2 / h^2$, then $\mathcal{F}_k \neq \emptyset$

Constrained minimization (N.Shor (1964) & B.Polyak)

II. Problem: $\min\limits_{x \in Q} \{ f(x) : \; g(x) \leq 0 \}$, where

- Q is closed and convex,
- f, g have uniformly bounded subgradients.

Consider the following method. It has step-size parameter $h > 0$.

$$\text{If } g(x_k) > h \, \|g'(x_k)\|, \quad \text{then (A):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{g(x_k)}{\|g'(x_k)\|^2} \, g'(x_k) \right),$$

$$\text{else (B):} \quad x_{k+1} = \pi_Q \left(x_k - \frac{h}{\|f'(x_k)\|} \, f'(x_k) \right).$$

Let $\mathcal{F}_k \subseteq \{0, \ldots, k\}$ be the set (B)-iterations, and $f_k^* = \min\limits_{i \in \mathcal{F}_k} f(x_i)$.

Theorem: If $k > \|x_0 - x^*\|^2 / h^2$, then $\mathcal{F}_k \neq \emptyset$ and

$$f_k^* - f(x) \leq hL(f), \quad \max_{i \in \mathcal{F}_k} g(x_i) \; \leq \; hL(g).$$

Linear Conic Problems

Linear Conic Problems

Assume that the space of primal variables E is partitioned:

$$x^j \in E_j, \; j = 1, \ldots, n, \quad x = \left(x^1, \ldots, x^n\right) \in E,$$

Linear Conic Problems

Assume that the space of primal variables E is partitioned:

$$x^j \;\in\; E_j,\; j = 1, \ldots, n, \quad x \;=\; \left(x^1, \ldots, x^n\right) \in E,$$

Thus, $\dim E = \sum\limits_{j=1}^{n} \dim E_j$, and $\langle c, x \rangle \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} \langle c^j, x^j \rangle$ for any $c \in E^*$.

Linear Conic Problems

Assume that the space of primal variables E is partitioned:

$$x^j \in E_j,\ j = 1, \ldots, n, \quad x = \left(x^1, \ldots, x^n\right) \in E,$$

Thus, $\dim E = \sum\limits_{j=1}^{n} \dim E_j$, and $\langle c, x \rangle \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} \langle c^j, x^j \rangle$ for any $c \in E^*$.

Linear operator: $\quad A = (A_1, \ldots, A_n), \quad Ax \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} A_j x^j,\ x \in E.$

Linear Conic Problems

Assume that the space of primal variables E is partitioned:

$$x^j \in E_j,\ j = 1, \ldots, n, \quad x = \left(x^1, \ldots, x^n\right) \in E,$$

Thus, $\dim E = \sum\limits_{j=1}^{n} \dim E_j$, and $\langle c, x \rangle \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} \langle c^j, x^j \rangle$ for any $c \in E^*$.

Linear operator: $\quad A = (A_1, \ldots, A_n), \quad Ax \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} A_j x^j,\ x \in E.$

Primal cone: $x \in K = \bigotimes\limits_{j=1}^{n} K_j$, $K_j \subset E_j$ are closed convex pointed.

Linear Conic Problems

Assume that the space of primal variables E is partitioned:

$$x^j \in E_j,\ j = 1, \ldots, n, \quad x = \left(x^1, \ldots, x^n\right) \in E,$$

Thus, $\dim E = \sum\limits_{j=1}^{n} \dim E_j$, and $\langle c, x \rangle \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} \langle c^j, x^j \rangle$ for any $c \in E^*$.

Linear operator: $\quad A = (A_1, \ldots, A_n), \quad Ax \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} A_j x^j,\ x \in E.$

Primal cone: $x \in K = \bigotimes\limits_{j=1}^{n} K_j$, $K_j \subset E_j$ are closed convex pointed.

Thus, $K^* = \bigotimes\limits_{j=1}^{n} K_j^*$.

Linear Conic Problems

Assume that the space of primal variables E is partitioned:

$$x^j \in E_j,\ j = 1, \ldots, n, \quad x = \left(x^1, \ldots, x^n\right) \in E,$$

Thus, $\dim E = \sum\limits_{j=1}^{n} \dim E_j$, and $\langle c, x \rangle \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} \langle c^j, x^j \rangle$ for any $c \in E^*$.

Linear operator: $\quad A = (A_1, \ldots, A_n), \quad Ax \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} A_j x^j,\ x \in E.$

Primal cone: $x \in K = \bigotimes\limits_{j=1}^{n} K_j$, $K_j \subset E_j$ are closed convex pointed.

Thus, $K^* = \bigotimes\limits_{j=1}^{n} K_j^*$.

Primal problem: $\quad f_* \stackrel{\mathrm{def}}{=} \inf\limits_{x \in K} \{ \langle c, x \rangle : \ Ax = b \}, \quad b \in R^m.$

Linear Conic Problems

Assume that the space of primal variables E is partitioned:

$$x^j \in E_j,\ j = 1, \ldots, n, \quad x = \left(x^1, \ldots, x^n\right) \in E,$$

Thus, $\dim E = \sum\limits_{j=1}^{n} \dim E_j$, and $\langle c, x \rangle \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} \langle c^j, x^j \rangle$ for any $c \in E^*$.

Linear operator: $\quad A = (A_1, \ldots, A_n), \quad Ax \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} A_j x^j,\ x \in E.$

Primal cone: $x \in K = \bigotimes\limits_{j=1}^{n} K_j$, $K_j \subset E_j$ are closed convex pointed.

Thus, $K^* = \bigotimes\limits_{j=1}^{n} K_j^*$.

Primal problem: $\quad f_* \stackrel{\mathrm{def}}{=} \inf\limits_{x \in K} \{ \langle c, x \rangle : \ Ax = b \}, \quad b \in R^m.$

Dual problem: $\quad \sup\limits_{y \in R^m,\ s \in K^*} \{ \langle b, y \rangle : \ s + A^* y = c \}.$

Linear Conic Problems

Assume that the space of primal variables E is partitioned:

$$x^j \in E_j,\ j = 1, \dots, n, \quad x = \left(x^1, \dots, x^n\right) \in E,$$

Thus, $\dim E = \sum\limits_{j=1}^{n} \dim E_j$, and $\langle c, x \rangle \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} \langle c^j, x^j \rangle$ for any $c \in E^*$.

Linear operator: $\quad A = (A_1, \dots, A_n), \quad Ax \stackrel{\mathrm{def}}{=} \sum\limits_{j=1}^{n} A_j x^j,\ x \in E.$

Primal cone: $x \in K = \bigotimes\limits_{j=1}^{n} K_j$, $K_j \subset E_j$ are closed convex pointed.

Thus, $K^* = \bigotimes\limits_{j=1}^{n} K_j^*$.

Primal problem: $\quad f_* \stackrel{\mathrm{def}}{=} \inf\limits_{x \in K} \{ \langle c, x \rangle : \ Ax = b \}, \quad b \in R^m.$

Dual problem: $\quad \sup\limits_{y \in R^m,\ s \in K^*} \{ \langle b, y \rangle : \ s + A^* y = c \}.$

Assumption: *Dual Problem is solvable.* $\Rightarrow \langle s^*, x^* \rangle = 0.$

Functional constraints

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

We need to write them in a functional form.

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

We need to write them in a functional form.

In each cone K_j^* we fix a *scaling element* $d^j \in \operatorname{int} K_j^*$, $j = 1, \ldots, n$.

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

We need to write them in a functional form.

In each cone K_j^* we fix a *scaling element* $d^j \in \operatorname{int} K_j^*$, $j = 1, \ldots, n$.

For $u^j \in E_j^*$, define $\psi_j(u^j) \stackrel{\text{def}}{=} \min_{\tau} \{ \tau : \; \tau d^j - u^j \in K_j^* \}$.

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

We need to write them in a functional form.

In each cone K_j^* we fix a *scaling element* $d^j \in \operatorname{int} K_j^*$, $j = 1, \ldots, n$.

For $u^j \in E_j^*$, define $\psi_j(u^j) \stackrel{\text{def}}{=} \min_{\tau} \{ \tau : \; \tau d^j - u^j \in K_j^* \}$.

Primal form: $\psi_j(u^j) = \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \}$.

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

We need to write them in a functional form.

In each cone K_j^* we fix a *scaling element* $d^j \in \operatorname{int} K_j^*$, $j = 1, \ldots, n$.

For $u^j \in E_j^*$, define $\psi_j(u^j) \stackrel{\text{def}}{=} \min_{\tau} \{ \tau : \; \tau d^j - u^j \in K_j^* \}$.

Primal form: $\psi_j(u^j) = \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \}$.

Thus, $\partial \psi_j(u^j) = \operatorname{Arg} \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \} \ni x^j(u^j)$.

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

We need to write them in a functional form.

In each cone K_j^* we fix a *scaling element* $d^j \in \operatorname{int} K_j^*$, $j = 1, \ldots, n$.

For $u^j \in E_j^*$, define $\psi_j(u^j) \stackrel{\text{def}}{=} \min_{\tau} \{ \tau : \; \tau d^j - u^j \in K_j^* \}$.

Primal form: $\psi_j(u^j) = \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \}$.

Thus, $\partial \psi_j(u^j) = \operatorname{Arg} \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \} \ni x^j(u^j)$.

Note: $c^j - A_j^T y \in K_j^*$ iff $f_j(y) \stackrel{\text{def}}{=} \psi_j(A_j^T y - c^j) \leq 0$.

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

We need to write them in a functional form.

In each cone K_j^* we fix a *scaling element* $d^j \in \operatorname{int} K_j^*$, $j = 1, \ldots, n$.

For $u^j \in E_j^*$, define $\psi_j(u^j) \stackrel{\text{def}}{=} \min_{\tau} \{ \tau : \; \tau d^j - u^j \in K_j^* \}$.

Primal form: $\psi_j(u^j) = \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \}$.

Thus, $\partial \psi_j(u^j) = \operatorname{Arg} \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \} \ni x^j(u^j)$.

Note: $c^j - A_j^T y \in K_j^*$ iff $f_j(y) \stackrel{\text{def}}{=} \psi_j(A_j^T y - c^j) \leq 0$.

Subgradients: $f_j'(y) \stackrel{\text{def}}{=} A_j \, x^j(A_j^T y - c^j) \; \in \; \partial f_j(y) \; \subset \; R^m$.

Functional constraints

Note: Constraints in the dual problem are separable

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; s^j = c^j - A_j^T y \in K_j^*, \; j = 1, \ldots, n \right\}.$$

We need to write them in a functional form.

In each cone K_j^* we fix a *scaling element* $d^j \in \operatorname{int} K_j^*$, $j = 1, \ldots, n$.

For $u^j \in E_j^*$, define $\psi_j(u^j) \stackrel{\text{def}}{=} \min_{\tau} \{ \tau : \; \tau d^j - u^j \in K_j^* \}$.

Primal form: $\psi_j(u^j) = \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \}$.

Thus, $\partial \psi_j(u^j) = \operatorname{Arg} \max_{x^j \in K_j} \{ \langle u^j, x^j \rangle : \; \langle d^j, x^j \rangle = 1 \} \ni x^j(u^j)$.

Note: $c^j - A_j^T y \in K_j^*$ iff $f_j(y) \stackrel{\text{def}}{=} \psi_j(A_j^T y - c^j) \leq 0$.

Subgradients: $f_j'(y) \stackrel{\text{def}}{=} A_j \, x^j(A_j^T y - c^j) \; \in \; \partial f_j(y) \; \subset \; R^m$.

Scaling: $\| f_j'(y) \| \leq \sigma_j \; \stackrel{\text{def}}{=} \; \lambda_{\max}^{1/2} \left(A_j \nabla^2 F_j^*(d^j) A_j^T \right)$.

Examples

Examples

1. If $K_j = R^1_+$, then $A_j = Ae_j \in R^m$, where e_j is a basis vector in R^n.

Examples

1. If $K_j = R^1_+$, then $A_j = Ae_j \in R^m$, where e_j is a basis vector in R^n.

Let us take $F_j(z) = -\ln z$ and $d^j = 1$.

Examples

1. If $K_j = R^1_+$, then $A_j = Ae_j \in R^m$, where e_j is a basis vector in R^n.

Let us take $F_j(z) = -\ln z$ and $d^j = 1$.Then $\nabla^2 F_j(z^j) = 1$ and

$$\sigma_j^2 = \lambda_{\max}(A_j A_j^T) = \|A_j\|^2.$$

Examples

1. If $K_j = R^1_+$, then $A_j = Ae_j \in R^m$, where e_j is a basis vector in R^n.
Let us take $F_j(z) = -\ln z$ and $d^j = 1$.Then $\nabla^2 F_j(z^j) = 1$ and

$$\sigma_j^2 = \lambda_{\max}(A_j A_j^T) = \|A_j\|^2.$$

2. Let $K_j = \{S_j \succeq 0_{p\times p}\}$.

Examples

1. If $K_j = R^1_+$, then $A_j = Ae_j \in R^m$, where e_j is a basis vector in R^n.
Let us take $F_j(z) = -\ln z$ and $d^j = 1$.Then $\nabla^2 F_j(z^j) = 1$ and

$$\sigma_j^2 = \lambda_{\max}(A_j A_J^T) = \|A_j\|^2.$$

2. Let $K_j = \{S_j \succeq 0_{p\times p}\}$.We take $F_j(z) = -\ln\det z$, and $z^j = d^j = I_p$.

Examples

1. If $K_j = R^1_+$, then $A_j = Ae_j \in R^m$, where e_j is a basis vector in R^n.
Let us take $F_j(z) = -\ln z$ and $d^j = 1$.Then $\nabla^2 F_j(z^j) = 1$ and

$$\sigma_j^2 = \lambda_{\max}(A_j A_j^T) = \|A_j\|^2.$$

2. Let $K_j = \{S_j \succeq 0_{p\times p}\}$.We take $F_j(z) = -\ln\det z$, and $z^j = d^j = I_p$.

Then $A_j^*(y) = \sum\limits_{i=1}^m A_j^i y^i,\ y \in R^m$, where A_j^i are symmetric $p \times p$-matrices.

Examples

1. If $K_j = R^1_+$, then $A_j = Ae_j \in R^m$, where e_j is a basis vector in R^n.
Let us take $F_j(z) = -\ln z$ and $d^j = 1$.Then $\nabla^2 F_j(z^j) = 1$ and

$$\sigma_j^2 = \lambda_{\max}(A_j A_J^T) = \|A_j\|^2.$$

2. Let $K_j = \{S_j \succeq 0_{p\times p}\}$.We take $F_j(z) = -\ln\det z$, and $z^j = d^j = I_p$.

Then $A_j^*(y) = \sum\limits_{i=1}^m A_j^i y^i,\ y \in R^m$, where A_j^i are symmetric $p \times p$-matrices.Thus,

$$\sigma_j = \max_{\|y\|=1} \|\sum_{i=1}^m A_j^i y^i\|_F = \max_{\substack{\|y\|=1,\\ \|B\|_F=1}} \langle \sum_{i=1}^m A_j^i y^i, B\rangle = \max_{\|B\|_F=1} \left[\sum_{i=1}^m \langle A_j^i, B\rangle^2\right]^{1/2}.$$

Examples

1. If $K_j = R^1_+$, then $A_j = Ae_j \in R^m$, where e_j is a basis vector in R^n.
Let us take $F_j(z) = -\ln z$ and $d^j = 1$.Then $\nabla^2 F_j(z^j) = 1$ and

$$\sigma_j^2 = \lambda_{\max}(A_j A_j^T) = \|A_j\|^2.$$

2. Let $K_j = \{S_j \succeq 0_{p\times p}\}$.We take $F_j(z) = -\ln\det z$, and $z^j = d^j = I_p$.

Then $A_j^*(y) = \sum\limits_{i=1}^m A_j^i y^i,\ y \in R^m$, where A_j^i are symmetric $p \times p$-matrices.Thus,

$$\sigma_j = \max_{\|y\|=1} \|\sum_{i=1}^m A_j^i y^i\|_F = \max_{\substack{\|y\|=1,\\ \|B\|_F=1}} \langle \sum_{i=1}^m A_j^i y^i, B\rangle = \max_{\|B\|_F=1} \left[\sum_{i=1}^m \langle A_j^i, B\rangle^2\right]^{1/2}.$$

We assume that all σ_j, $j = 1, \ldots, n$, are computed in advance.

New Dual Problem

New Dual Problem

Denote $g_j(y) = \frac{1}{\sigma_j} f_j(y)$. Consider the problem:

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; g(y) \stackrel{\mathrm{def}}{=} \max_{1 \le j \le n} g_j(y) \; \le \; 0 \right\}.$$

New Dual Problem

Denote $g_j(y) = \frac{1}{\sigma_j} f_j(y)$. Consider the problem:

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; g(y) \stackrel{\mathrm{def}}{=} \max_{1 \le j \le n} g_j(y) \; \le \; 0 \right\}.$$

Denote by $j(y)$ the active index j such that $g_j(y) = g(y)$. Then

$$g'(y) = \frac{1}{\sigma_{j(y)}} A_{j(y)} x^{j(y)} \left(A_{j(y)}^T y - c^{j(y)} \right), \quad \|g'(y)\| \; \le \; 1.$$

New Dual Problem

Denote $g_j(y) = \frac{1}{\sigma_j} f_j(y)$. Consider the problem:

$$\sup_{y \in R^m,\, s \in E^*} \left\{ \langle b, y \rangle : \; g(y) \stackrel{\mathrm{def}}{=} \max_{1 \le j \le n} g_j(y) \; \le \; 0 \right\}.$$

Denote by $j(y)$ the active index j such that $g_j(y) = g(y)$. Then

$$g'(y) = \frac{1}{\sigma_{j(y)}} A_{j(y)} x^{j(y)} \left(A_{j(y)}^T y - c^{j(y)} \right), \quad \|g'(y)\| \; \le \; 1.$$

Maximization scheme: Choose $h > 0$. Define $y_0 = 0$.

For $k \ge 0$ **do:**

if $g(y_k) \le h$**, then (F):** $\quad y_{k+1} = y_k + h \cdot \frac{b}{\|b\|}$,

else (G): $\quad y_{k+1} = y_k - g(y_k) \cdot g'(y_k)$.

Primal and dual minimization sequences

Primal and dual minimization sequences

For $N \geq 0$, denote by $\mathcal{F}_N$ the set of iterations of type (F).
Let $\mathcal{G}_N \stackrel{\mathrm{def}}{=} \{0, \ldots, N\} \setminus \mathcal{F}_N$, $N_f \stackrel{\mathrm{def}}{=} |\mathcal{F}_N|$, and $N_g \stackrel{\mathrm{def}}{=} |\mathcal{G}_N|$.

Primal and dual minimization sequences

For $N \geq 0$, denote by $\mathcal{F}_N$ the set of iterations of type (F).

Let $\mathcal{G}_N \stackrel{\mathrm{def}}{=} \{0, \ldots, N\} \setminus \mathcal{F}_N$, $N_f \stackrel{\mathrm{def}}{=} |\mathcal{F}_N|$, and $N_g \stackrel{\mathrm{def}}{=} |\mathcal{G}_N|$.

For step (F), $c^j - A_j^* y_k + h\sigma_j d^j \in K_j^*$, $j = 1, \ldots, n, \quad k \in \mathcal{F}_N$.

Primal and dual minimization sequences

For $N \geq 0$, denote by $\mathcal{F}_N$ the set of iterations of type (F).

Let $\mathcal{G}_N \stackrel{\mathrm{def}}{=} \{0, \ldots, N\} \setminus \mathcal{F}_N$, $N_f \stackrel{\mathrm{def}}{=} |\mathcal{F}_N|$, and $N_g \stackrel{\mathrm{def}}{=} |\mathcal{G}_N|$.

For step (F), $c^j - A_j^* y_k + h\sigma_j d^j \in K_j^*$, $j = 1, \ldots, n, \quad k \in \mathcal{F}_N$.

Denote $e_j(x^j) \in E$: $e_j^i(x^j) = \begin{cases} x^j, & i = j, \\ 0, & \text{otherwise}, \end{cases} \qquad i = 1, \ldots, n.$

Primal and dual minimization sequences

For $N \geq 0$, denote by $\mathcal{F}_N$ the set of iterations of type (F).

Let $\mathcal{G}_N \stackrel{\mathrm{def}}{=} \{0, \ldots, N\} \setminus \mathcal{F}_N$, $N_f \stackrel{\mathrm{def}}{=} |\mathcal{F}_N|$, and $N_g \stackrel{\mathrm{def}}{=} |\mathcal{G}_N|$.

For step (F), $c^j - A_j^* y_k + h\sigma_j d^j \in K_j^*$, $j = 1, \ldots, n, \quad k \in \mathcal{F}_N$.

Denote $e_j(x^j) \in E$: $e_j^i(x^j) = \begin{cases} x^j, & i = j, \\ 0, & \text{otherwise}, \end{cases} \qquad i = 1, \ldots, n.$

Define the approximate primal-dual solutions as follows:

$$\bar{x}_N \stackrel{\mathrm{def}}{=} \frac{\|b\|}{hN_f} \sum_{k \in \mathcal{G}_N} \frac{g(y_k)}{\sigma_{j(y_k)}} e_{j(y_k)} \left(x^{j(y_k)} \left(A_{j(y_k)}^* y_k - c^{j(y_k)} \right) \right) \in K,$$

$$\bar{y}_N = \frac{1}{N_f} \sum_{k \in \mathcal{F}_N} y_k, \quad \bar{s}_N = c - A^T \bar{y}_N.$$

Primal and dual minimization sequences

For $N \geq 0$, denote by $\mathcal{F}_N$ the set of iterations of type (F).

Let $\mathcal{G}_N \stackrel{\mathrm{def}}{=} \{0, \ldots, N\} \setminus \mathcal{F}_N$, $N_f \stackrel{\mathrm{def}}{=} |\mathcal{F}_N|$, and $N_g \stackrel{\mathrm{def}}{=} |\mathcal{G}_N|$.

For step (F), $c^j - A_j^* y_k + h\sigma_j d^j \in K_j^*$, $j = 1, \ldots, n, \quad k \in \mathcal{F}_N$.

Denote $e_j(x^j) \in E$: $e_j^i(x^j) = \begin{cases} x^j, & i = j, \\ 0, & \text{otherwise}, \end{cases} \qquad i = 1, \ldots, n.$

Define the approximate primal-dual solutions as follows:

$$\bar{x}_N \stackrel{\mathrm{def}}{=} \frac{\|b\|}{hN_f} \sum_{k \in \mathcal{G}_N} \frac{g(y_k)}{\sigma_{j(y_k)}} e_{j(y_k)} \left(x^{j(y_k)} \left(A_{j(y_k)}^* y_k - c^{j(y_k)} \right) \right) \in K,$$

$$\bar{y}_N = \frac{1}{N_f} \sum_{k \in \mathcal{F}_N} y_k, \quad \bar{s}_N = c - A^T \bar{y}_N.$$

This choice is motivated by the following relations:

Primal and dual minimization sequences

For $N \geq 0$, denote by $\mathcal{F}_N$ the set of iterations of type (F).

Let $\mathcal{G}_N \stackrel{\mathrm{def}}{=} \{0, \ldots, N\} \setminus \mathcal{F}_N$, $N_f \stackrel{\mathrm{def}}{=} |\mathcal{F}_N|$, and $N_g \stackrel{\mathrm{def}}{=} |\mathcal{G}_N|$.

For step (F), $c^j - A_j^* y_k + h\sigma_j d^j \in K_j^*$, $j = 1, \ldots, n, \quad k \in \mathcal{F}_N$.

Denote $e_j(x^j) \in E$: $e_j^i(x^j) = \begin{cases} x^j, & i = j, \\ 0, & \text{otherwise}, \end{cases} \qquad i = 1, \ldots, n.$

Define the approximate primal-dual solutions as follows:

$$\bar{x}_N \stackrel{\mathrm{def}}{=} \frac{\|b\|}{hN_f} \sum_{k \in \mathcal{G}_N} \frac{g(y_k)}{\sigma_{j(y_k)}} e_{j(y_k)} \left(x^{j(y_k)} \left(A_{j(y_k)}^* y_k - c^{j(y_k)} \right) \right) \in K,$$

$$\bar{y}_N = \frac{1}{N_f} \sum_{k \in \mathcal{F}_N} y_k, \quad \bar{s}_N = c - A^T \bar{y}_N.$$

This choice is motivated by the following relations:

$$\bar{s}_N^j = c^j - \frac{1}{N_f} \sum_{k \in \mathcal{F}_N} A_j^* y_k \succeq_{K_j^*} -h\sigma_j d^j,$$

Primal and dual minimization sequences

For $N \geq 0$, denote by $\mathcal{F}_N$ the set of iterations of type (F).

Let $\mathcal{G}_N \stackrel{\mathrm{def}}{=} \{0, \ldots, N\} \setminus \mathcal{F}_N$, $N_f \stackrel{\mathrm{def}}{=} |\mathcal{F}_N|$, and $N_g \stackrel{\mathrm{def}}{=} |\mathcal{G}_N|$.

For step (F), $c^j - A_j^* y_k + h\sigma_j d^j \in K_j^*$, $j = 1, \ldots, n, \quad k \in \mathcal{F}_N$.

Denote $e_j(x^j) \in E$: $e_j^i(x^j) = \begin{cases} x^j, & i = j, \\ 0, & \text{otherwise}, \end{cases} \qquad i = 1, \ldots, n.$

Define the approximate primal-dual solutions as follows:

$$\bar{x}_N \stackrel{\mathrm{def}}{=} \frac{\|b\|}{hN_f} \sum_{k \in \mathcal{G}_N} \frac{g(y_k)}{\sigma_{j(y_k)}} e_{j(y_k)} \left(x^{j(y_k)} (A_{j(y_k)}^* y_k - c^{j(y_k)}) \right) \in K,$$

$$\bar{y}_N = \frac{1}{N_f} \sum_{k \in \mathcal{F}_N} y_k, \quad \bar{s}_N = c - A^T \bar{y}_N.$$

This choice is motivated by the following relations:

$$\bar{s}_N^j = c^j - \frac{1}{N_f} \sum_{k \in \mathcal{F}_N} A_j^* y_k \succeq_{K_j^*} -h\sigma_j d^j,$$

$$y_{N+1} = \frac{hN_f}{\|b\|} \cdot b - \sum_{k \in \mathcal{G}_N} \frac{g(y_k)}{\sigma_{j(y_k)}} A e_{j(y_k)} \left(x^{j(y_k)} (A_{j(y_k)}^* y_k - c^{j(y_k)}) \right).$$

Convergence

Convergence

Denote $\hat{d} \in K^*$: $\hat{d}^j = \sigma_j d^j, \quad j = 1, \ldots, n.$

Convergence

Denote $\hat{d} \in K^*$: $\hat{d}^j = \sigma_j d^j, \quad j = 1, \ldots, n.$

Theorem. Let $\hat{D} = 2\left(\frac{\langle \hat{d}, x^* \rangle}{\|b\|} + 1\right)$. For any $N \geq 0$ we have:

$$N_f \geq \frac{1}{\hat{D}}\left(N + 1 - \frac{\|y^*\|^2}{h^2}\right).$$

Convergence

Denote $\hat{d} \in K^*$: $\hat{d}^j = \sigma_j d^j, \quad j = 1, \ldots, n.$

Theorem. Let $\hat{D} = 2\left(\frac{\langle \hat{d}, x^* \rangle}{\|b\|} + 1\right)$. For any $N \geq 0$ we have:

$$N_f \geq \frac{1}{\hat{D}}\left(N + 1 - \frac{\|y^*\|^2}{h^2}\right).$$

If $N_f \geq 1$, then $\langle c, \bar{x}_N \rangle - \langle b, \bar{y}_N \rangle \leq \frac{1}{2} h \|b\|$.

Convergence

Denote $\hat{d} \in K^*$: $\hat{d}^j = \sigma_j d^j, \quad j = 1, \ldots, n.$

Theorem. Let $\hat{D} = 2\left(\frac{\langle \hat{d}, x^* \rangle}{\|b\|} + 1\right)$. For any $N \geq 0$ we have:

$$N_f \geq \frac{1}{\hat{D}}\left(N + 1 - \frac{\|y^*\|^2}{h^2}\right).$$

If $N_f \geq 1$, then $\langle c, \bar{x}_N \rangle - \langle b, \bar{y}_N \rangle \leq \frac{1}{2} h \|b\|$.

Finally, if $\quad N + 1 > \frac{\|y^*\|^2}{h^2}$, then

$$\langle x^*, \bar{s}_N \rangle + \langle \bar{x}_N, s^* \rangle \leq h\|b\|,$$

and the residual in the primal-dual system vanishes as $N \to \infty$:

$$\frac{1}{\|b\|}\|b - A\bar{x}_N\| \leq \sqrt{\frac{\hat{D}}{N_f}} + \frac{\|y^*\|}{hN_f}.$$

Example: Solving huge LP

Example: Solving huge LP

Let $K = R^n_+$. Then $\sigma_j = \|Ae_j\|$, $j = 1, \ldots, n$.

Example: Solving huge LP

Let $K = R^n_+$. Then $\sigma_j = \|Ae_j\|$, $j = 1, \ldots, n$.
Assume the data is uniformly sparse: for all i and j

$$p(c) \leq r, \quad p(A^T e_i) \leq r, \quad p(b) \leq q, \quad p(Ae_j) \leq q,$$

with $r \ll n$ and $q \ll m$.

Example: Solving huge LP

Let $K = R^n_+$. Then $\sigma_j = \|Ae_j\|$, $j = 1, \ldots, n$.
Assume the data is uniformly sparse: for all i and j

$$p(c) \leq r, \quad p(A^T e_i) \leq r, \quad p(b) \leq q, \quad p(Ae_j) \leq q,$$

with $r \ll n$ and $q \ll m$.

Preliminary work: $O(p(A))$ operations at most.

Example: Solving huge LP

Let $K = R^n_+$. Then $\sigma_j = \|Ae_j\|$, $j = 1, \ldots, n$.
Assume the data is uniformly sparse: for all i and j

$$p(c) \leq r, \quad p(A^T e_i) \leq r, \quad p(b) \leq q, \quad p(Ae_j) \leq q,$$

with $r \ll n$ and $q \ll m$.

Preliminary work: $O(p(A))$ operations at most.

One iteration:

Example: Solving huge LP

Let $K = R^n_+$. Then $\sigma_j = \|Ae_j\|$, $j = 1, \ldots, n$.
Assume the data is uniformly sparse: for all i and j

$$p(c) \leq r, \quad p(A^T e_i) \leq r, \quad p(b) \leq q, \quad p(Ae_j) \leq q,$$

with $r \ll n$ and $q \ll m$.

Preliminary work: $O(p(A))$ operations at most.

One iteration:

- Update y_k: $O(q)$ operations at most.

Example: Solving huge LP

Let $K = R^n_+$. Then $\sigma_j = \|Ae_j\|$, $j = 1, \ldots, n$.
Assume the data is uniformly sparse: for all i and j

$$p(c) \leq r, \quad p(A^T e_i) \leq r, \quad p(b) \leq q, \quad p(Ae_j) \leq q,$$

with $r \ll n$ and $q \ll m$.

Preliminary work: $O(p(A))$ operations at most.

One iteration:

- Update y_k: $O(q)$ operations at most.
- Update new slack s_{k+1}: $O(rq \log_2 n)$ operations.

Example: Solving huge LP

Let $K = R^n_+$. Then $\sigma_j = \|Ae_j\|$, $j = 1, \ldots, n$.
Assume the data is uniformly sparse: for all i and j

$$p(c) \leq r, \quad p(A^T e_i) \leq r, \quad p(b) \leq q, \quad p(Ae_j) \leq q,$$

with $r \ll n$ and $q \ll m$.

Preliminary work: $O(p(A))$ operations at most.

One iteration:

- Update y_k: $O(q)$ operations at most.
- Update new slack s_{k+1}: $O(rq \log_2 n)$ operations.
- Update the norm $\|y_k\|^2$: $O(q)$ operations.

Example: Solving huge LP

Let $K = R^n_+$. Then $\sigma_j = \|Ae_j\|$, $j = 1, \ldots, n$.
Assume the data is uniformly sparse: for all i and j

$$p(c) \leq r, \quad p(A^T e_i) \leq r, \quad p(b) \leq q, \quad p(Ae_j) \leq q,$$

with $r \ll n$ and $q \ll m$.

Preliminary work: $O(p(A))$ operations at most.

One iteration:

- Update y_k: $O(q)$ operations at most.
- Update new slack s_{k+1}: $O(rq \log_2 n)$ operations.
- Update the norm $\|y_k\|^2$: $O(q)$ operations.

Conclusion: cost of one iteration is $O(rq \log_2 n)$.

NB: Often r and q do not depend on n.

Computational experiments: Iteration Cost

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals. Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \leq i \leq M} \kappa_A(A^T e_i) = p^2$.

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals. Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \leq i \leq M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \leq \kappa(A) \log_2 N$

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals. Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \leq i \leq M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \leq \kappa(A) \log_2 N \approx p^2 \log_2 N$,

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals. Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \le i \le M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \le \kappa(A) \log_2 N \approx p^2 \log_2 N, \quad GM \approx pN.$

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals. Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \le i \le M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \le \kappa(A) \log_2 N \approx p^2 \log_2 N, \quad GM \approx pN.$

$(\log_2 \ 10^3 = 10,$

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals. Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \le i \le M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \le \kappa(A) \log_2 N \approx p^2 \log_2 N, \quad GM \approx pN.$

$(\log_2 10^3 = 10, \quad \log_2 10^6 = 20,$

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals. Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \le i \le M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \le \kappa(A) \log_2 N \approx p^2 \log_2 N, \quad GM \approx pN.$

$$(\log_2\ 10^3 = 10, \quad \log_2\ 10^6 = 20, \quad \log_2\ 10^9 = 30)$$

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals.
Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \le i \le M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \le \kappa(A) \log_2 N \approx p^2 \log_2 N, \quad GM \approx pN.$

$$(\log_2\ 10^3 = 10, \quad \log_2\ 10^6 = 20, \quad \log_2\ 10^9 = 30)$$

Time for 10^4 iterations ($p = 32$)

N	$\kappa(A)$	GM_s	GM
1024	1632	3.00	2.98
2048	1792	3.36	6.41
4096	1888	3.75	15.11
8192	1920	4.20	139.92
16384	1824	4.69	408.38

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals.
Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \le i \le M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \le \kappa(A) \log_2 N \approx p^2 \log_2 N, \quad GM \approx pN.$

$$(\log_2\ 10^3 = 10, \quad \log_2\ 10^6 = 20, \quad \log_2\ 10^9 = 30)$$

Time for 10^4 iterations ($p = 32$)

N	$\kappa(A)$	GM_s	GM
1024	1632	3.00	2.98
2048	1792	3.36	6.41
4096	1888	3.75	15.11
8192	1920	4.20	139.92
16384	1824	4.69	408.38

Time for 10^3 iterations ($p = 16$)

N	$\kappa(A)$	GM_s	GM
131072	576	0.19	213.9
262144	592	0.25	477.8
524288	592	0.32	1095.5
1048576	608	0.40	2590.8

Computational experiments: Iteration Cost

We compare Polyak's GM with sparse update (GM_s) with the standard one (GM).

Problem: Maximum of linear functions with p nonzero diagonals.
Thus, $\kappa(A) \stackrel{\mathrm{def}}{=} \max\limits_{1 \le i \le M} \kappa_A(A^T e_i) = p^2$.

Iteration Cost: $GM_s \le \kappa(A) \log_2 N \approx p^2 \log_2 N, \quad GM \approx pN.$

$$(\log_2 10^3 = 10, \quad \log_2 10^6 = 20, \quad \log_2 10^9 = 30)$$

Time for 10^4 iterations ($p = 32$)

N	$\kappa(A)$	GM_s	GM
1024	1632	3.00	2.98
2048	1792	3.36	6.41
4096	1888	3.75	15.11
8192	1920	4.20	139.92
16384	1824	4.69	408.38

Time for 10^3 iterations ($p = 16$)

N	$\kappa(A)$	GM_s	GM
131072	576	0.19	213.9
262144	592	0.25	477.8
524288	592	0.32	1095.5
1048576	608	0.40	2590.8

1 sec $\approx$ 100 min!

Convergence of GM_s: Large Scale

Convergence of GM_s: Large Scale

Let $N = 1048576$, $p = 8$, $\kappa(A) = 192$, and $L(f) = 0.21$.

Convergence of GM_s: Large Scale

Let $N = 1048576$, $p = 8$, $\kappa(A) = 192$, and $L(f) = 0.21$.

Iterations	$f - f^*$	Time (sec)
0	2.000000	0.00
$1.0 \cdot 10^5$	0.546662	7.69
$4.0 \cdot 10^5$	0.276866	30.74
$1.0 \cdot 10^6$	0.137822	76.86
$2.5 \cdot 10^6$	0.063099	192.14
$5.1 \cdot 10^6$	0.032092	391.97
$9.9 \cdot 10^6$	0.016162	760.88
$1.5 \cdot 10^7$	0.010009	1183.59

Convergence of GM_s: Large Scale

Let $N = 1048576$, $p = 8$, $\kappa(A) = 192$, and $L(f) = 0.21$.

Iterations	$f - f^*$	Time (sec)
0	2.000000	0.00
$1.0 \cdot 10^5$	0.546662	7.69
$4.0 \cdot 10^5$	0.276866	30.74
$1.0 \cdot 10^6$	0.137822	76.86
$2.5 \cdot 10^6$	0.063099	192.14
$5.1 \cdot 10^6$	0.032092	391.97
$9.9 \cdot 10^6$	0.016162	760.88
$1.5 \cdot 10^7$	0.010009	1183.59

Final point $\bar{x}_*$**:** $\|\bar{x}_*\|_\infty = 2.941497$, $R_0^2 \stackrel{\text{def}}{=} \|\bar{x}_* - e\|_2^2 = 1.2 \cdot 10^5$.

Convergence of GM_s: Large Scale

Let $N = 1048576$, $p = 8$, $\kappa(A) = 192$, and $L(f) = 0.21$.

Iterations	$f - f^*$	Time (sec)
0	2.000000	0.00
$1.0 \cdot 10^5$	0.546662	7.69
$4.0 \cdot 10^5$	0.276866	30.74
$1.0 \cdot 10^6$	0.137822	76.86
$2.5 \cdot 10^6$	0.063099	192.14
$5.1 \cdot 10^6$	0.032092	391.97
$9.9 \cdot 10^6$	0.016162	760.88
$1.5 \cdot 10^7$	0.010009	1183.59

Final point $\bar{x}_*$**:** $\|\bar{x}_*\|_\infty = 2.941497$, $R_0^2 \stackrel{\text{def}}{=} \|\bar{x}_* - e\|_2^2 = 1.2 \cdot 10^5$.

Theoretical bound: $\frac{L^2(f) R_0^2}{\epsilon^2} = 5.3 \cdot 10^7$.

Convergence of GM_s: Large Scale

Let $N = 1048576$, $p = 8$, $\kappa(A) = 192$, and $L(f) = 0.21$.

Iterations	$f - f^*$	Time (sec)
0	2.000000	0.00
$1.0 \cdot 10^5$	0.546662	7.69
$4.0 \cdot 10^5$	0.276866	30.74
$1.0 \cdot 10^6$	0.137822	76.86
$2.5 \cdot 10^6$	0.063099	192.14
$5.1 \cdot 10^6$	0.032092	391.97
$9.9 \cdot 10^6$	0.016162	760.88
$1.5 \cdot 10^7$	0.010009	1183.59

Final point $\bar{x}_*$**:** $\|\bar{x}_*\|_\infty = 2.941497$, $R_0^2 \stackrel{\text{def}}{=} \|\bar{x}_* - e\|_2^2 = 1.2 \cdot 10^5$.
Theoretical bound: $\frac{L^2(f) R_0^2}{\epsilon^2} = 5.3 \cdot 10^7$. **Time for GM:** ≈ 1 year!

Thank you for your attention!