First-order methods
(Optml++ Meeting 2)

Suvrit Sra
Massachusetts Institute of Technology

OPTML++, Fall 2015

Outline

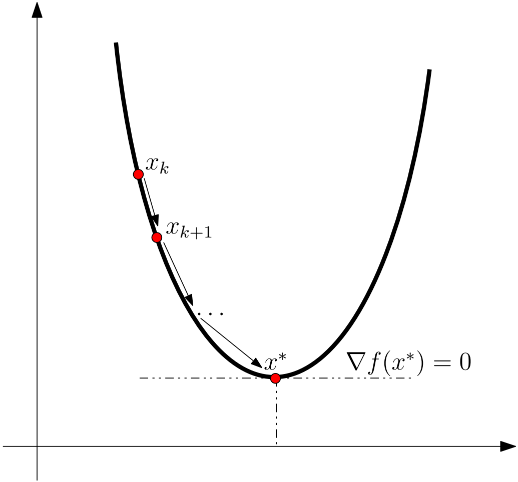
- Lect 1: Recap on convexity
- Lect 1: Recap on duality, optimality
- First-order optimization algorithms
- Proximal methods, operator splitting

Descent methods

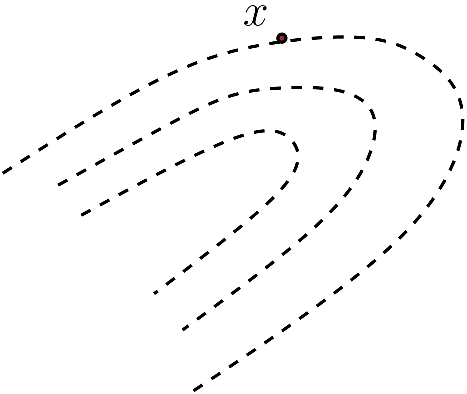
$$\min_x \quad f(x)$$

Descent methods

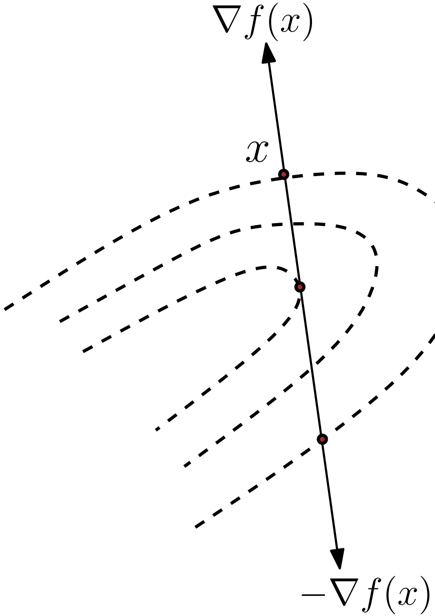
$$\min_x \quad f(x)$$



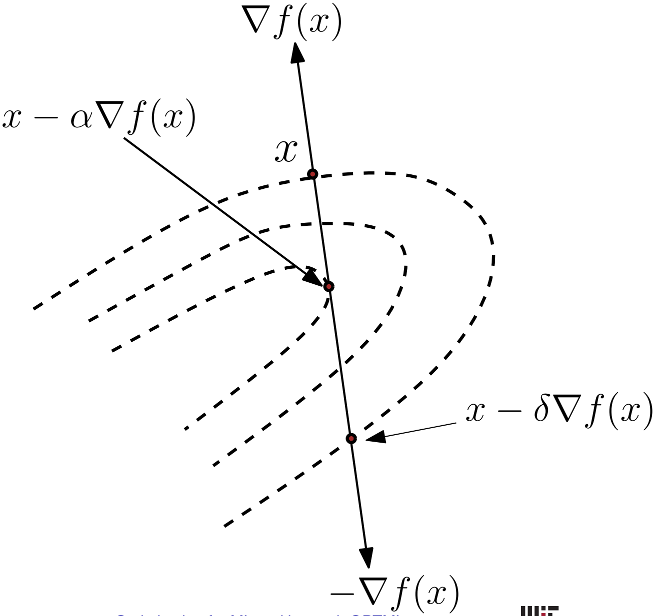
Descent methods



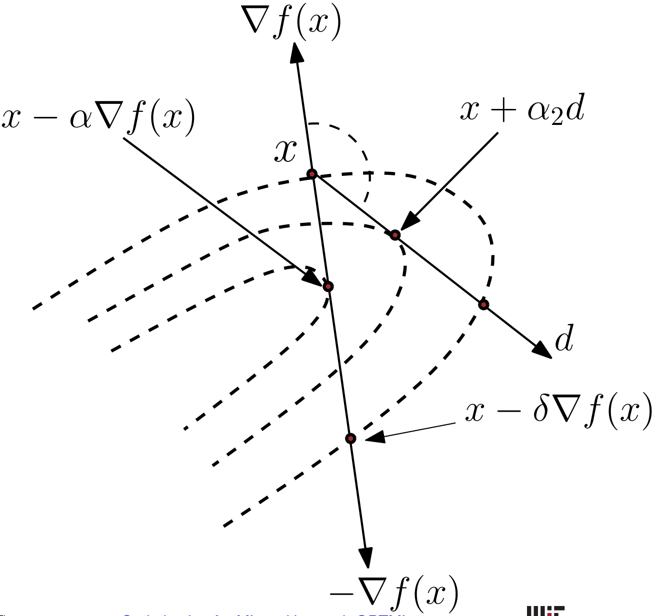
Descent methods



Descent methods



Descent methods



Algorithm

1. Start with some guess x^0;
2. For each $k = 0, 1, \ldots$
   - $x^{k+1} \leftarrow x^k + \alpha_k d^k$
   - Check when to stop (e.g., if $\nabla f(x^{k+1}) = 0$)

Gradient methods

$$x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots$$

- **stepsize** $\alpha_k \geq 0$, usually ensures $f(x^{k+1}) < f(x^k)$

Gradient methods

$$x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots$$

- **stepsize** $\alpha_k \geq 0$, usually ensures $f(x^{k+1}) < f(x^k)$
- **Descent direction** d^k satisfies

$$\langle \nabla f(x^k), d^k \rangle < 0$$

Gradient methods

$$x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots$$

- **stepsize** $\alpha_k \geq 0$, usually ensures $f(x^{k+1}) < f(x^k)$
- **Descent direction** d^k satisfies

$$\langle \nabla f(x^k), d^k \rangle < 0$$

Numerous ways to select α_k and d^k

Gradient methods

$$x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots$$

- **stepsize** $\alpha_k \geq 0$, usually ensures $f(x^{k+1}) < f(x^k)$
- **Descent direction** d^k satisfies

$$\langle \nabla f(x^k), d^k \rangle < 0$$

Numerous ways to select α_k and d^k

Usually methods **seek monotonic descent**

$$f(x^{k+1}) < f(x^k)$$

Gradient methods – direction

$$x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots$$

- Different choices of direction d^k
  - **Scaled gradient:** $d^k = -D^k \nabla f(x^k)$, $D^k \succ 0$
  - **Newton's method:** $(D^k = [\nabla^2 f(x^k)]^{-1})$
  - **Quasi-Newton:** $D^k \approx [\nabla^2 f(x^k)]^{-1}$
  - **Steepest descent:** $D^k = \boldsymbol{I}$
  - **Diagonally scaled:** D^k diagonal with $D^k_{ii} \approx \left(\frac{\partial^2 f(x^k)}{(\partial x_i)^2}\right)^{-1}$
  - **Discretized Newton:** $D^k = [H(x^k)]^{-1}$, H via finite-diff.

Gradient methods – direction

$$x^{k+1} = x^k + \alpha_k d^k, \quad k = 0, 1, \ldots$$

- Different choices of direction d^k
  - **Scaled gradient:** $d^k = -D^k \nabla f(x^k)$, $D^k \succ 0$
  - **Newton's method:** $(D^k = [\nabla^2 f(x^k)]^{-1})$
  - **Quasi-Newton:** $D^k \approx [\nabla^2 f(x^k)]^{-1}$
  - **Steepest descent:** $D^k = \boldsymbol{I}$
  - **Diagonally scaled:** D^k diagonal with $D^k_{ii} \approx \left(\frac{\partial^2 f(x^k)}{(\partial x_i)^2}\right)^{-1}$
  - **Discretized Newton:** $D^k = [H(x^k)]^{-1}$, H via finite-diff.
  - …

Exercise: Verify that $\langle \nabla f(x^k), d^k \rangle < 0$ for above choices

Gradient methods – stepsize

- **Exact:** $\alpha_k := \operatorname{argmin}_{\alpha \geq 0} f(x^k + \alpha d^k)$

Gradient methods – stepsize

- **Exact:** $\alpha_k := \underset{\alpha \geq 0}{\operatorname{argmin}} f(x^k + \alpha d^k)$
- **Limited min:** $\alpha_k = \underset{0 \leq \alpha \leq s}{\operatorname{argmin}} f(x^k + \alpha d^k)$

Gradient methods – stepsize

- **Exact:** $\alpha_k := \underset{\alpha \geq 0}{\text{argmin}}\, f(x^k + \alpha d^k)$
- **Limited min:** $\alpha_k = \underset{0 \leq \alpha \leq s}{\text{argmin}}\, f(x^k + \alpha d^k)$
- **Armijo-rule**. Given **fixed** scalars, s, β, σ with $0 < \beta < 1$ and $0 < \sigma < 1$ (chosen experimentally). Set

$$\alpha_k = \beta^{m_k} s,$$

where we **try** $\beta^m s$ for $m = 0, 1, \ldots$ until **sufficient descent**

$$f(x^k) - f(x + \beta^m s d^k) \geq -\sigma \beta^m s \langle \nabla f(x^k), d^k \rangle$$

Gradient methods – stepsize

- **Exact:** $\alpha_k := \operatorname{argmin}_{\alpha \geq 0} f(x^k + \alpha d^k)$
- **Limited min:** $\alpha_k = \operatorname{argmin}_{0 \leq \alpha \leq s} f(x^k + \alpha d^k)$
- **Armijo-rule**. Given **fixed** scalars, s, β, σ with $0 < \beta < 1$ and $0 < \sigma < 1$ (chosen experimentally). Set

$$\alpha_k = \beta^{m_k} s,$$

where we **try** $\beta^m s$ for $m = 0, 1, \ldots$ until **sufficient descent**

$$f(x^k) - f(x + \beta^m s d^k) \geq -\sigma \beta^m s \langle \nabla f(x^k), d^k \rangle$$

If $\langle \nabla f(x^k), d^k \rangle < 0$, stepsize guaranteed to exist

Gradient methods – stepsize

- **Exact:** $\alpha_k := \underset{\alpha \geq 0}{\operatorname{argmin}} f(x^k + \alpha d^k)$
- **Limited min:** $\alpha_k = \underset{0 \leq \alpha \leq s}{\operatorname{argmin}} f(x^k + \alpha d^k)$
- **Armijo-rule**. Given **fixed** scalars, s, β, σ with $0 < \beta < 1$ and $0 < \sigma < 1$ (chosen experimentally). Set

  $$\alpha_k = \beta^{m_k} s,$$

  where we **try** $\beta^m s$ for $m = 0, 1, \ldots$ until **sufficient descent**

  $$f(x^k) - f(x + \beta^m s d^k) \geq -\sigma \beta^m s \langle \nabla f(x^k), d^k \rangle$$

  If $\langle \nabla f(x^k), d^k \rangle < 0$, stepsize guaranteed to exist
  Usually, σ small $\in [10^{-5}, 0.1]$, while β from $1/2$ to $1/10$ depending on how confident we are about initial stepsize s.
- **Constant:** $\alpha_k = 1/L$ (for suitable value of L)
- **Diminishing:** $\alpha_k \to 0$ but $\sum_k \alpha_k = \infty$.

Gradient methods – nonmonotonic steps*

- Stepsize computation can be expensive
- Convergence analysis depends on monotonic descent

Gradient methods – nonmonotonic steps*

- Stepsize computation can be expensive
- Convergence analysis depends on monotonic descent
- Give up search for stepsizes
- Use closed-form formulae for stepsizes
- Don't insist on monotonic descent?
- (e.g., diminishing stepsizes do not give monotonic descent)

Gradient methods – nonmonotonic steps*

- Stepsize computation can be expensive
- Convergence analysis depends on monotonic descent
- Give up search for stepsizes
- Use closed-form formulae for stepsizes
- Don't insist on monotonic descent?
- (e.g., diminishing stepsizes do not give monotonic descent)

Barzilai & Borwein stepsizes

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k), \quad k = 0, 1, \ldots$$

Gradient methods – nonmonotonic steps*

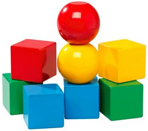
- Stepsize computation can be expensive
- Convergence analysis depends on monotonic descent
- Give up search for stepsizes
- Use closed-form formulae for stepsizes
- Don't insist on monotonic descent?
- (e.g., diminishing stepsizes do not give monotonic descent)

Barzilai & Borwein stepsizes

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k), \quad k = 0, 1, \ldots$$

$$\alpha_k = \frac{\langle u^k, v^k \rangle}{\|v^k\|^2}, \qquad \alpha_k = \frac{\|u^k\|^2}{\langle u^k, v^k \rangle}$$

$$u^k = x^k - x^{k-1}, \quad v^k = \nabla f(x^k) - \nabla f(x^{k-1})$$



Least-squares

Nonnegative least squares

$$\min \quad \tfrac{1}{2}\|Ax - b\|^2 + [\![x \geq 0]\!]$$

intensities, concentrations, frequencies, . . .

Applications

Machine learning
Statistics
Image Processing
Computer Vision
Medical Imaging
Astronomy

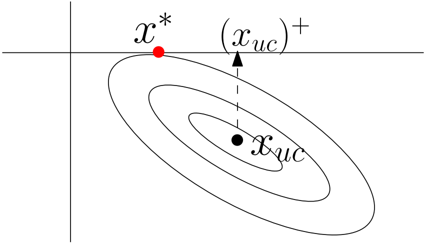
Physics
Bioinformatics
Remote Sensing
Engineering
Inverse problems
Finance

NNLS: $\|Ax - b\|^2$ s.t. $x \geq 0$

Unconstrained solution

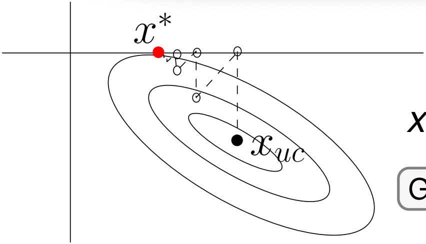
Solve $\quad \nabla f(x) = 0 \quad \Longrightarrow \quad x_{uc} = (A^T A)^{-1} A^T b$

Cannot just truncate $\quad x = (x_{uc})^+$



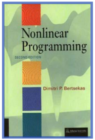
$x \geq 0$ makes problem trickier as **problem size** $\nearrow$

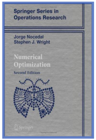
Solving NNLS scalably



$$x \leftarrow (x - \alpha \nabla f(x))^+$$

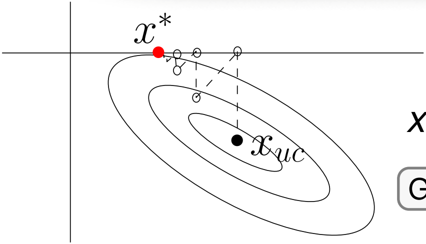
Good choice of α **crucial**





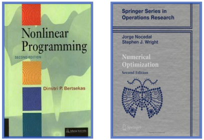
- Backtracking line-search
- Armijo
- and many others

Solving NNLS scalably



$$x \leftarrow (x - \alpha \nabla f(x))^+$$

Good choice of α **crucial**



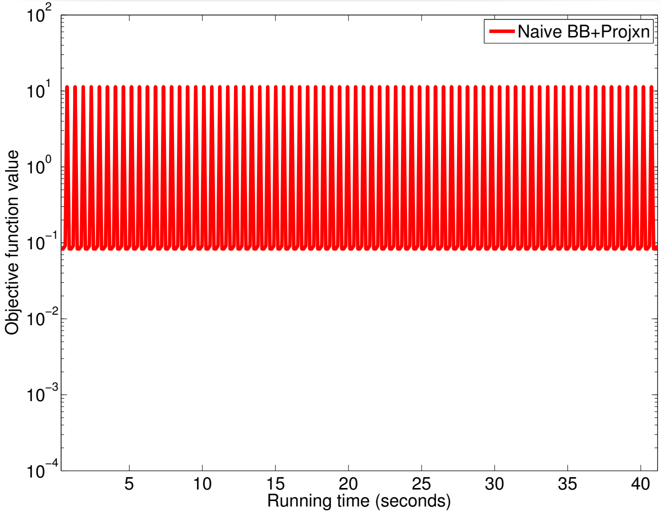
- Backtracking line-search
- Armijo
- and many others

Too slow!

NNLS: long studied problem

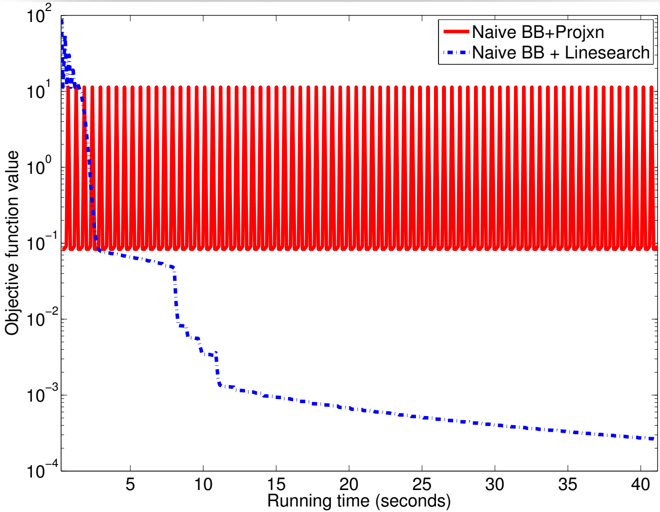
Method	Remarks	Scalability	Accuracy
NNLS (1976)	MATLAB default	poor	**high**
FNNLS (1989)	fast NNLS	poor	**high**
LBFGS-B (1997)	famous solver	fair	medium
TRON (1999)	TR newton	poor	**high**
SPG (2000)	spectral proj	fair+	medium
ASA (2006)	prev state-of-art	fair+	medium
SBB (2011)	subspace BB steps	**very good**	medium

Spectacular failure of projection



$$x' = (x - \alpha \nabla f(x))^+$$

Rescue: occasional line-search?



Mix BB-step with linesearch

Can we completely avoid linesearch?

Do not use all coordinates to compute α!

“Subspace-BB” (SBB)

Kim, Sra, Dhillon (OMS, 2011)

Identify **fixed** variables
(those likely to satisfy $x_i = 0$)
Compute α using **free** variables
Most crucial step!

Can we completely avoid linesearch?

Do not use all coordinates to compute α!

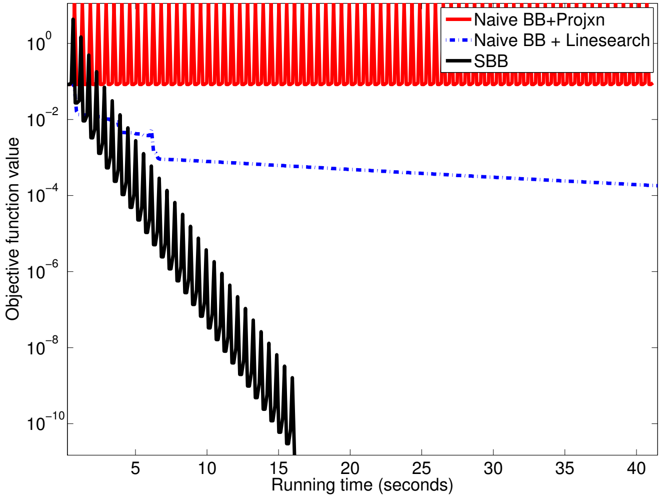
"Subspace-BB" (SBB)

Kim, Sra, Dhillon (OMS, 2011)

Identify **fixed** variables
(those likely to satisfy $x_i = 0$)
Compute α using **free** variables
Most crucial step!

SBB convergence theorem
Global rate – open problem
Empirically great!

SBB: simplicity and scalability



Numerical result

Algorithm	Time	$\|Ax - b\|^2$	Convg. tol.
LBFGS-B (FORTRAN)	19000s	20.2	**1.0E-03**
SPG (FORTRAN)	8600s	20.5	3.8E-01
ASA (C++)	1001s	24.5	4.8e-02

"medium" 20,000 $\times$ 1,350,000 matrix

Numerical result

Algorithm	Time	$\|Ax - b\|^2$	Convg. tol.
LBFGS-B (FORTRAN)	19000s	20.2	**1.0E-03**
SPG (FORTRAN)	8600s	20.5	3.8E-01
ASA (C++)	1001s	24.5	4.8e-02
SBB (MATLAB)	**201s**	**21.2**	**8.7E-03**

"medium" $20{,}000 \times 1{,}350{,}000$ matrix

Back to gradient-descent

Assumption: **Lipschitz continuous gradient**; denoted $f \in C_L^1$

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2$$

Back to gradient-descent

Assumption: Lipschitz continuous gradient; denoted $f \in C^1_L$

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2$$

- ♣ Gradient vectors of closeby points are close to each other
- ♣ Objective function has "bounded curvature"
- ♣ Speed at which gradient varies is bounded

Back to gradient-descent

Assumption: Lipschitz continuous gradient; denoted $f \in C_L^1$

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2$$

- ♣ Gradient vectors of closeby points are close to each other
- ♣ Objective function has "bounded curvature"
- ♣ Speed at which gradient varies is bounded

Lemma (Descent). Let $f \in C_L^1$. Then,

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \tfrac{L}{2}\|y - x\|_2^2$$

Theorem Let $f \in C_L^1$ and $\{x^k\}$ be sequence generated as above, with $\alpha_k = 1/L$. Then, $f(x^{k+1}) - f(x^*) = O(1/k)$.

Linear convergence

Assumption: **Strong convexity**; denote $f \in S^1_{L,\mu}$

$$f(x) \geq f(y) + \langle \nabla f(y), x - y \rangle + \frac{\mu}{2}\|x - y\|_2^2$$

- Setting $\alpha_k = 2/(\mu + L)$ yields linear rate $(\mu > 0)$

Strongly convex – linear rate

Theorem. If $f \in S^1_{L,\mu}$, $0 < \alpha < 2/(L+\mu)$, then the gradient method generates a sequence $\{x^k\}$ that satisfies

$$\|x^k - x^*\|_2^2 \leq \left(1 - \frac{2\alpha\mu L}{\mu + L}\right)^k \|x^0 - x^*\|_2.$$

Moreover, if $\alpha = 2/(L+\mu)$ then

$$f(x^k) - f^* \leq \frac{L}{2}\left(\frac{\kappa - 1}{\kappa + 1}\right)^{2k} \|x^0 - x^*\|_2^2,$$

where $\kappa = L/\mu$ is the **condition number**.

Gradient methods – lower bounds

$$x^{k+1} = x^k - \alpha_k \nabla f(x^k)$$

Theorem Lower bound I (Nesterov) For any $x^0 \in \mathbb{R}^n$, and $1 \leq k \leq \frac{1}{2}(n-1)$, there is a smooth f, s.t.

$$f(x^k) - f(x^*) \geq \frac{3L\|x^0 - x^*\|_2^2}{32(k+1)^2}$$

Gradient methods – lower bounds

$$x^{k+1} = x^k - \alpha_k \nabla f(x^k)$$

Theorem Lower bound I (Nesterov) For any $x^0 \in \mathbb{R}^n$, and $1 \leq k \leq \frac{1}{2}(n-1)$, there is a smooth f, s.t.

$$f(x^k) - f(x^*) \geq \frac{3L\|x^0 - x^*\|_2^2}{32(k+1)^2}$$

Theorem Lower bound II (Nesterov). For class of smooth, strongly convex, i.e., $S^\infty_{L,\mu}$ ($\mu > 0$, $\kappa > 1$)

$$f(x^k) - f(x^*) \geq \frac{\mu}{2}\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{2k} \|x^0 - x^*\|_2^2.$$