ORF522 – Linear and Nonlinear Optimization

21. Acceleration schemes

Bartolomeo Stellato — Fall 2022

Today's lecture
[Chapter 3, COAC][Section 2.2, ILCO][Chapter 1, FMO]

First-order methods acceleration

- Lower bounds
- Acceleration
- Interpretation and examples

Recap of nonlinear optimization

Lower bounds

Sublinear convergence rates

For a convex L-smooth function f we have

Gradient descent

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

Proximal gradient

$$x^{k+1} = \mathbf{prox}_{tg}(x^k - t\nabla f(x^k))$$

Convergence

$$f(x^k) - f(x^\star) \leq \frac{\|x^0 - x^\star\|_2^2}{2tk} \longrightarrow$$

distance $O(1/k)$

iterations $O(1/\epsilon)$

Can we do better? Is there a lower bound?

Lower bounds

First-order methods

Any algorithm that selects

$$x^{k+1} \in x_0 + \mathbf{span}\{\nabla f(x_0), \nabla f(x_1), \dots, \nabla f(x^k)\}$$

Theorem (Nesterov ’83)

For every integer $k \leq (n-1)/2$, there exist a convex L-smooth function f such that, for any first-order method

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

distance	$O(1/k^2)$
iterations	$O(1/\sqrt{\epsilon})$

Lower bound proof

minimize $\quad f(x) = \frac{L}{4}\left(\frac{1}{2}x^T A x - e_1^T x\right) \longrightarrow \nabla f(x) = \frac{L}{4}\left(Ax - e_1\right)$

Gil. Strang (MIT) "cupcake matrix"

$$A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}, \qquad e_1 = (1, 0, \ldots, 0)$$

- f is convex and L-smooth
- $x^\star$ is the **optimizer** with $x_i^\star = 1 - \frac{i}{n+1}$ (Solves $\nabla f(x^\star) = 0 \to Ax^\star = e_1$)
- $f(x^\star) = \frac{L}{4}\left(\frac{1}{2}e_1^T x^\star - e_1^T x^\star\right) = -\frac{L}{8}x_1^\star = -\frac{L}{8}\frac{n}{n+1}, \qquad \|x^\star\|^2 \leq \frac{n+1}{3}$

Lower bound proof

Iterates

If $x^0 = 0$ then $x^k \in \mathbf{span}\{\nabla f(x^0), \dots, \nabla f(x^{k-1})\} = \mathbf{span}\{e_1, \dots, e_k\}$

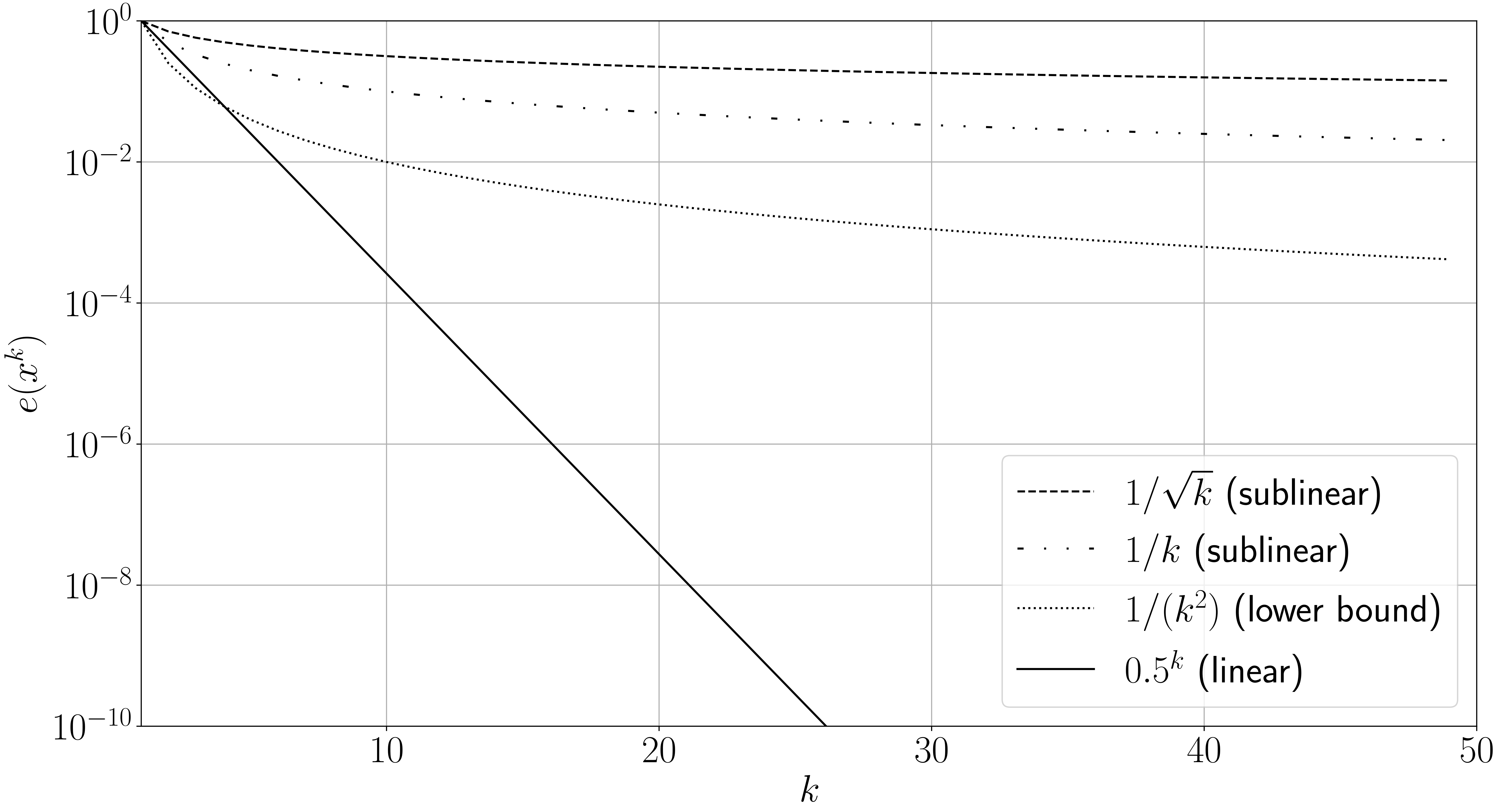
Upper bound

$$f(x^k) \geq \min_{x \in \mathbf{span}\{\nabla f(x^0), \dots, \nabla f(x^{k-1})\}} f(x) = \min_{x_{k+1} = \cdots = x_n = 0} f(x) = -\frac{L}{8} \frac{k}{k+1}$$

For $k \approx n/2$ or $n = 2k+1$,

$$\frac{f(x^k) - f(x^\star)}{\|x^0 - x^\star\|^2} \geq \frac{L}{8} \left(-\frac{k}{k+1} + \frac{2k+1}{2k+2} \right) / \left(\frac{2k+2}{3} \right) = \frac{3L}{32(k+1)^2} \qquad \blacksquare$$

Convergence rates



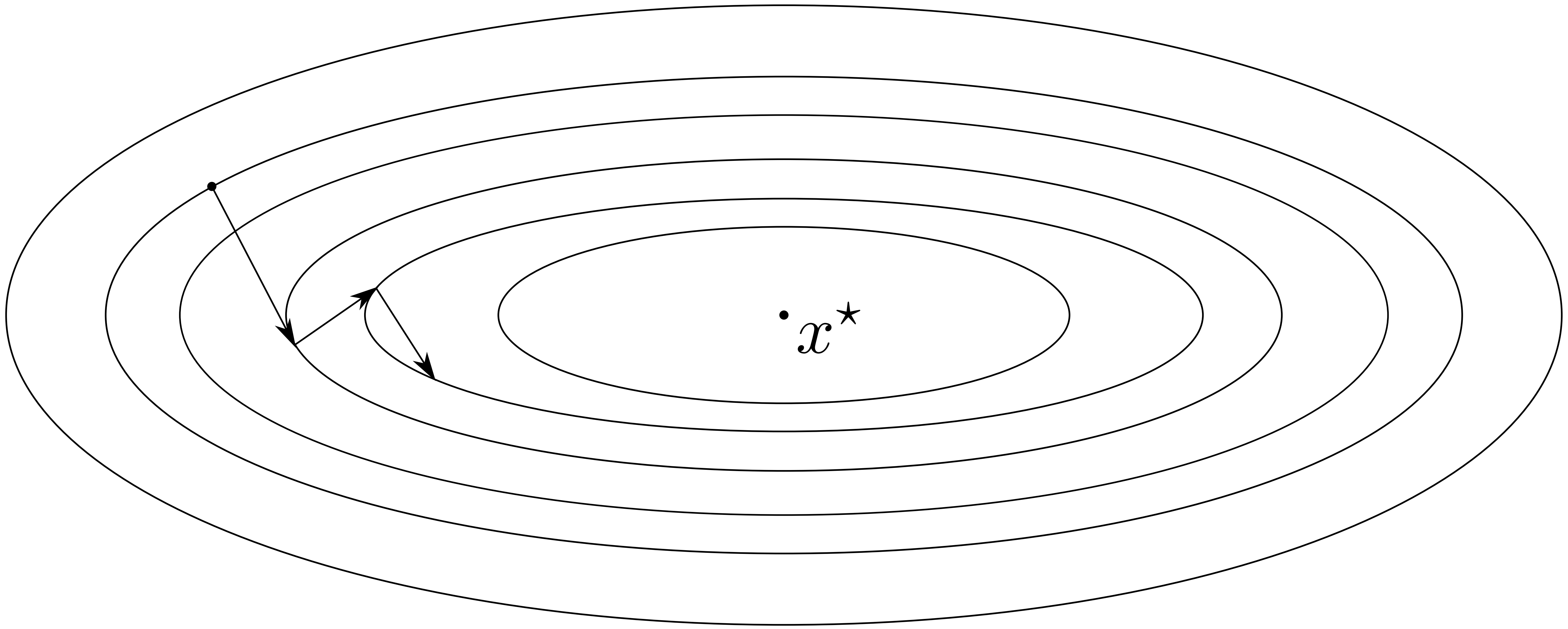
Can we achieve the lower bound?

Acceleration

Momentum

Gradient descent

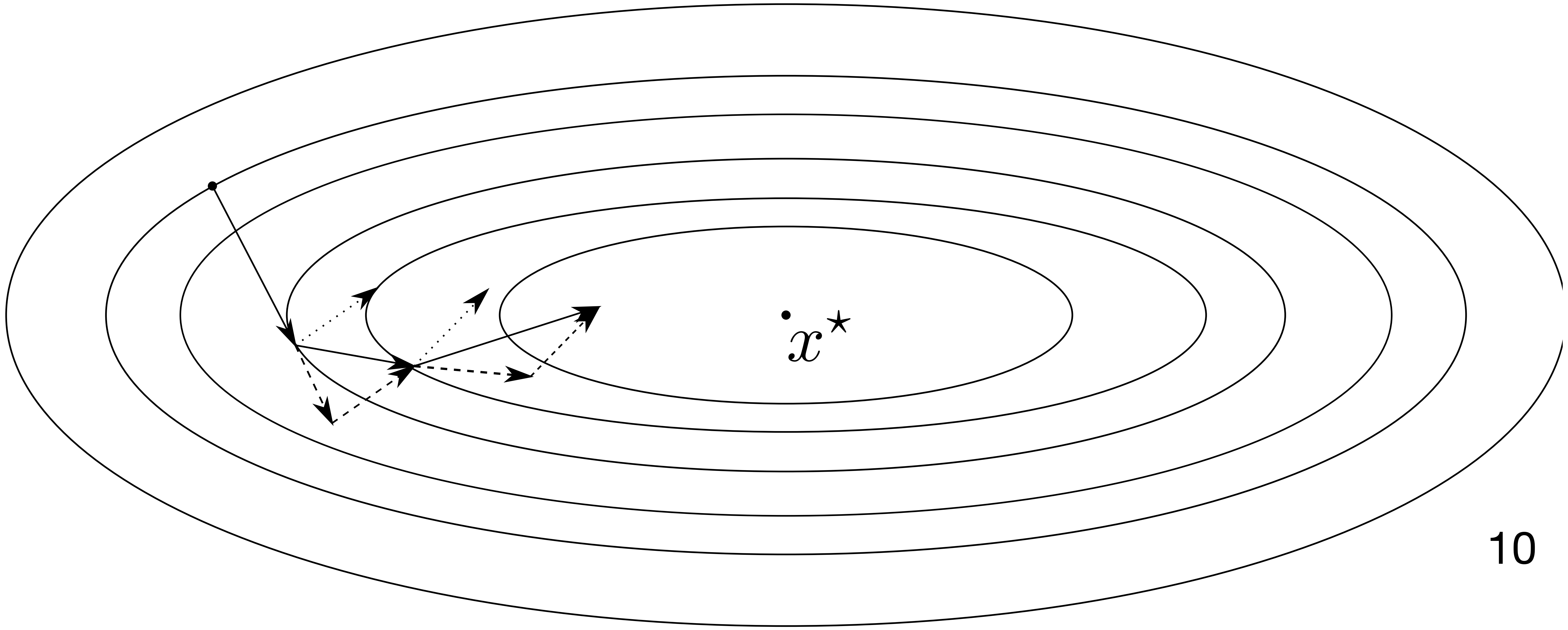
$$x^{k+1} = x^k - t\nabla f(x^k)$$



Adding momentum

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

$$y^{k+1} = x^{k+1} + \underbrace{\beta_k}_{\textbf{momentum}}(x^{k+1} - x^k)$$



Nesterov momentum

$$x^{k+1} = y^k - t\nabla f(y^k)$$
$$y^{k+1} = x^{k+1} + \frac{k}{k+3}(x^{k+1} - x^k)$$

Properties

- Original Momentum proposed by Nesterov ('83)
- No longer a descent method (i.e., we can have $f(x^{k+1}) > f(x^k)$)
- Same complexity per iteration as gradient descent

Accelerated proximal gradient method

minimize $\quad f(x) + g(x)$

$f(x)$ convex and smooth
$g(x)$ convex (may be not differentiable)

Iterations

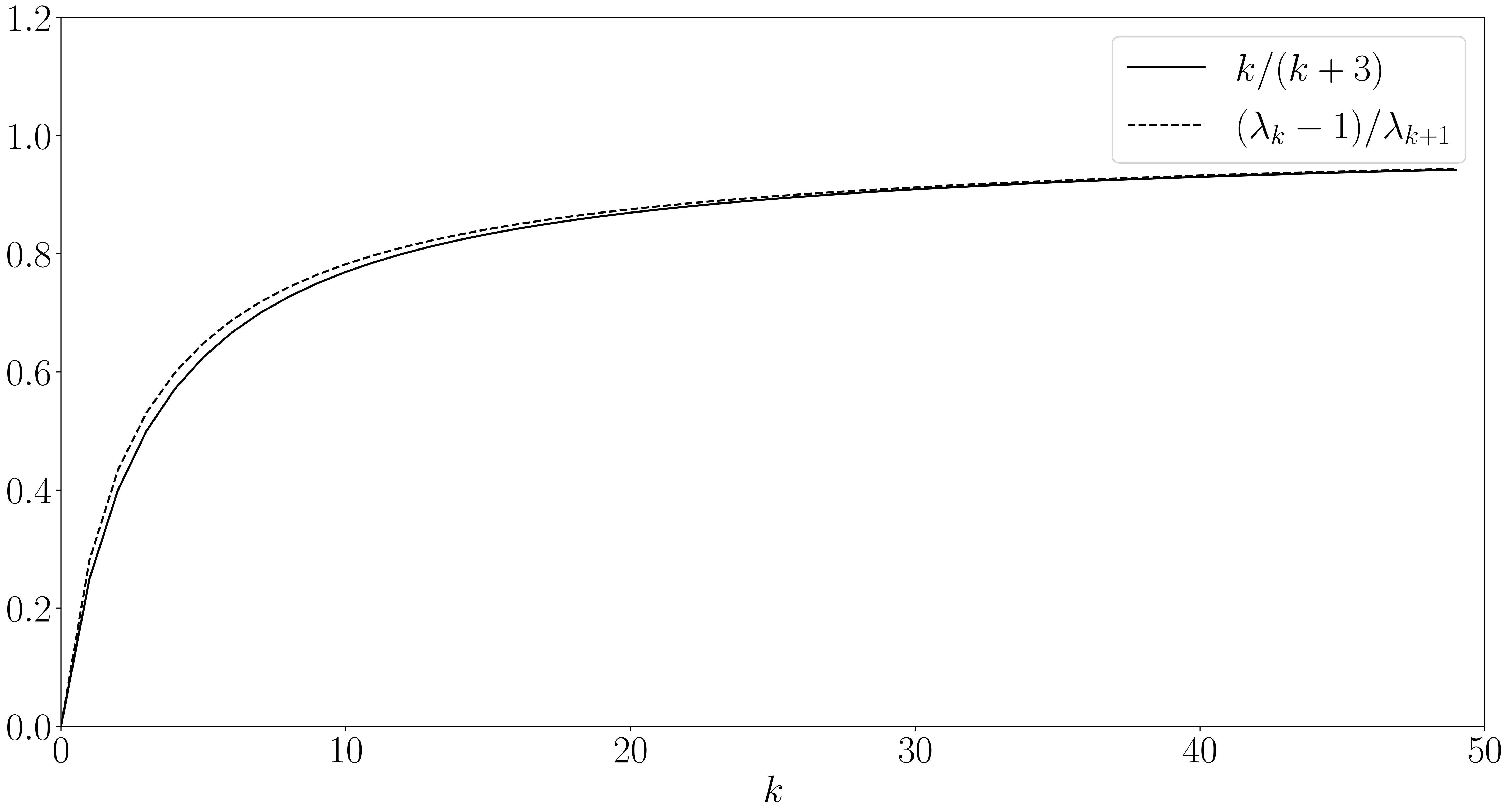
$$x^{k+1} = \mathbf{prox}_{tg}\left(y^k - t\nabla f(y^k)\right)$$
$$y^{k+1} = x^{k+1} + \frac{\lambda_k - 1}{\lambda_{k+1}}(x^{k+1} - x^k)$$

where $y_0 = x_0$ and $\lambda_{k+1} = \dfrac{1 + \sqrt{1 + 4\lambda_k^2}}{2}$

Note: $g(x) = 0$ gives accelerated gradient descent

Proximal gradient and Nesterov weights

$$\lambda_0 = 1 \qquad \lambda_{k+1} = \frac{1+\sqrt{1+4\lambda_k^2}}{2} \longrightarrow \frac{\lambda_k - 1}{\lambda_{k+1}} \approx \frac{k}{k+3} \text{ as } k \to \infty$$



Convergence rate for accelerated proximal gradient method

minimize $\quad f(x) + g(x)$

$f(x)$ convex and L-smooth
$g(x)$ convex (may be not differentiable)

Theorem

The accelerated proximal gradient method with step-size $t \leq (1/L)$ satisfies

$$f(x^k) - f(x^\star) \leq \frac{2\|x^0 - x^\star\|^2}{t(k+1)^2}$$

Proof

[Thm 4.4, A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems, Beck, Teboulle]

Note

It works for any momentum weights $(\lambda_k - 1)/\lambda_{k+1}$ such that

$$\lambda_k \geq \frac{k+2}{2} \quad \text{and} \quad \lambda_{k+1}^2 \leq \lambda_{k+1} - \lambda_k^2$$

Convergence rate for accelerated proximal gradient method

minimize $\quad f(x) + g(x)$

$f(x)$ convex and L-smooth
$g(x)$ convex (may be not differentiable)

$$f(x^k) - f(x^\star) \leq \frac{2\|x^0 - x^\star\|^2}{t(k+1)^2}$$

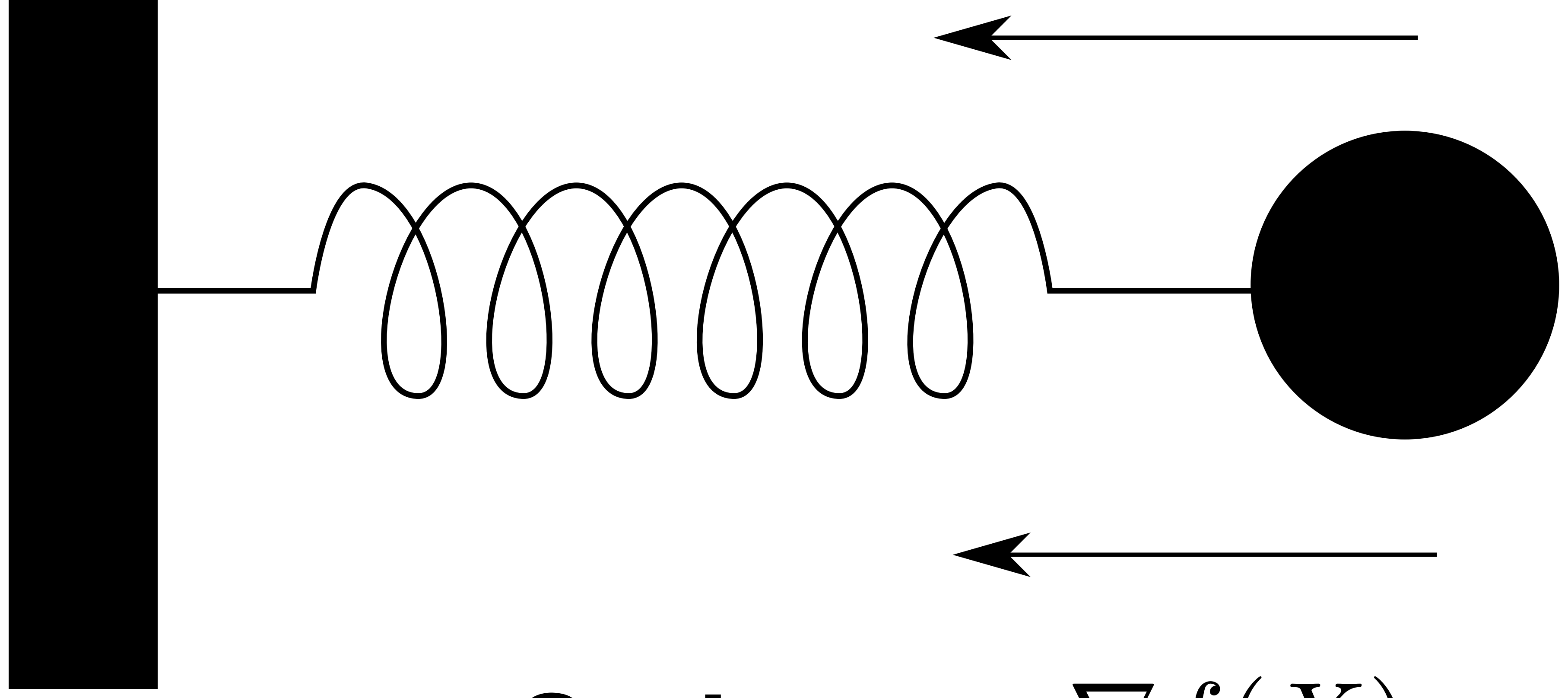
- Better iteration complexity $O(1/k^2)$ (i.e. $O(1/\sqrt{\epsilon})$
- Fast if $\mathbf{prox}$ evaluations are cheap
- Can’t do better! (from **lower bound**)

Examples and interpretations

ODE interpretation

Time-varying damping

$-\frac{3}{\tau}\dot{X}$



Spring $-\nabla f(X)$

Nesterov acceleration

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

$$y^{k+1} = x^{k+1} + \frac{k}{k+3}(x^{k+1} - x^k)$$

$$t \to 0 \qquad x^k \approx X(k\sqrt{t}) = X(\tau)$$

$$\ddot{X}(\tau) + \frac{3}{\tau}\dot{X}(\tau) + \nabla f(X(\tau)) = 0$$

damping coefficient

Note: 3 is the smallest constant that guarantees $O(1/\tau^2)$ convergence

[A Differential Equation for Modeling Nesterov's Accelerated Gradient Method: Theory and Insights, Su, Boyd, Candes]

Example: Lasso without linear convergence

minimize $\underbrace{(1/2)\|Ax - b\|_2^2}_{f(x)} + \underbrace{\gamma\|x\|_1}_{g(x)}$

Proximal gradient descent
(Iterative Shrinkage Thresholding Algorithm)

$$x^{k+1} = S_{\gamma t}\left(x^k - tA^T(Ax^k - b)\right)$$

ISTA

Accelerated proximal gradient descent
(Fast Iterative Shrinkage Thresholding Algorithm)

$$x^{k+1} = S_{\gamma t}\left(y^k - tA^T(Ay^k - b)\right)$$

$$y^{k+1} = x^{k+1} + \frac{\lambda_k - 1}{\lambda_{k+1}}(x^{k+1} - x^k)$$

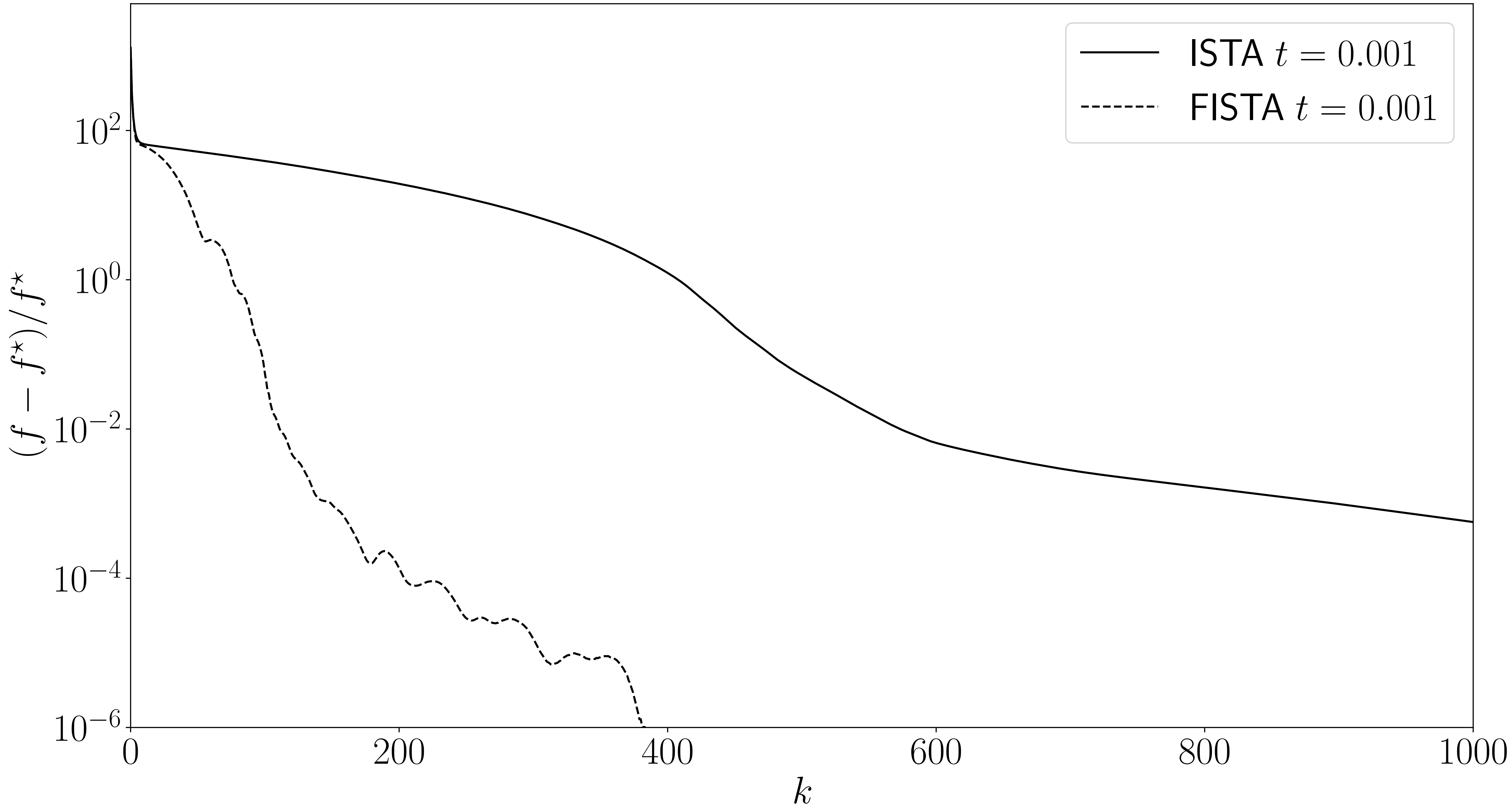
FISTA

[A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems, Beck, Teboulle]

Example: Lasso without linear convergence

Fast Iterative Soft Thresholding Algorithm (FISTA)

minimize $\quad (1/2)\|Ax - b\|_2^2 + \gamma\|x\|_1$



Example

randomly generated $\quad A \in \mathbf{R}^{300\times 500}$

$\Rightarrow \quad \nabla^2 f = A^T A \succeq 0$

$\Rightarrow \quad f$ not strongly convex

FISTA is much faster

Typical rippling behavior
(not a descent method)

Image deblurring

minimize $\quad (1/2)\|Ax - b\|_2^2 + \gamma \|x\|_1$

x: reconstructed image
in wavelet basis (sparse)

original

blurred

$k = 100$ $\quad$ $k = 200$

ISTA

FISTA

[A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems, Beck, Teboulle]

More sophisticated accelerations

Other algorithms

Acceleration can also be applied also to ADMM

[Fast Alternating Direction Optimization Methods, Goldstein, O'Donoghue, Setzer, Baraniuk]

Momentum with **restarts**
(reset momentum when it makes
small progress)

$\longrightarrow$

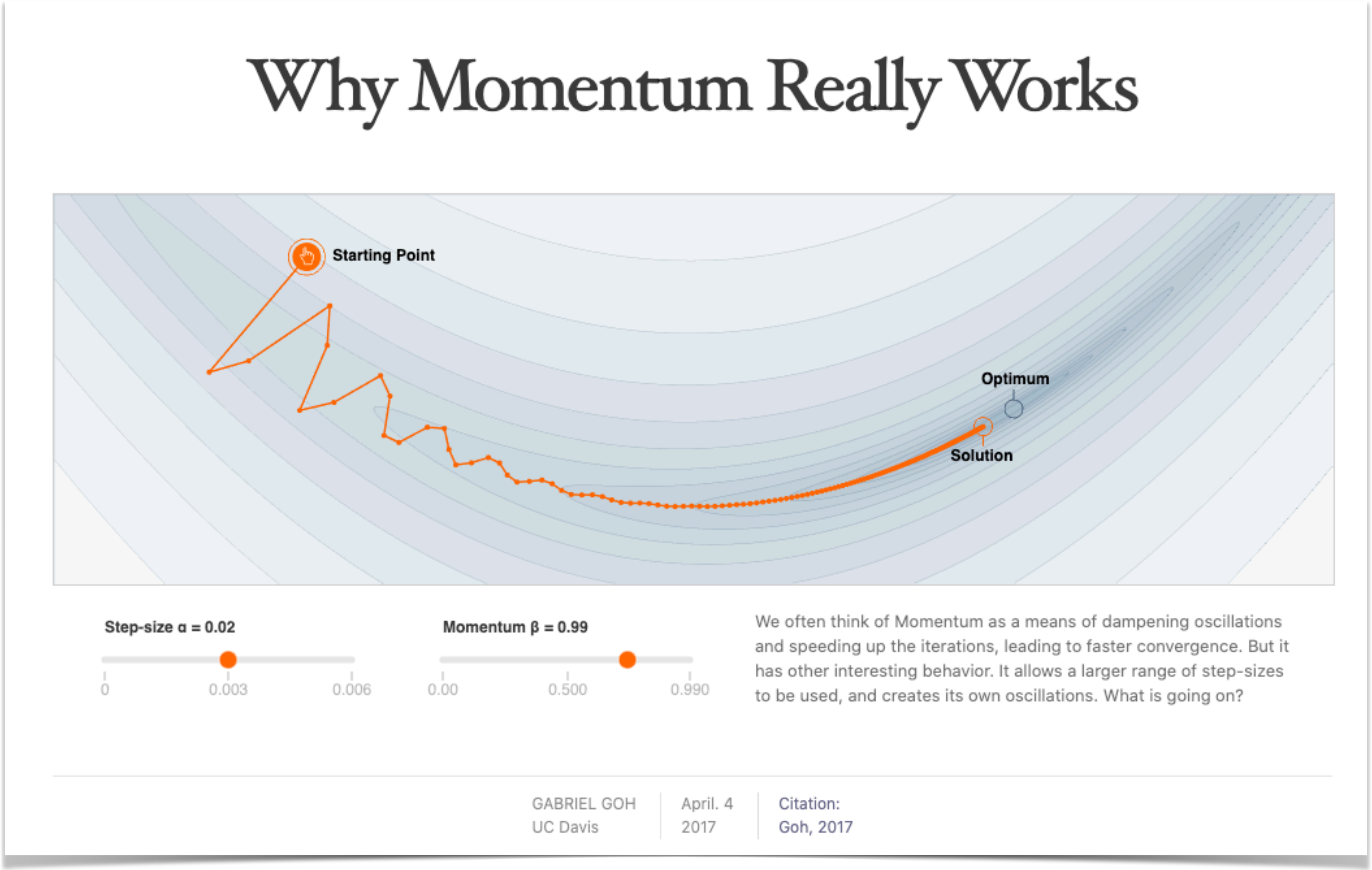
**Improved
convergence rate**
$O(1/k^2)$

Nonlinear acceleration
(e.g., Anderson Acceleration)

Adaptively pick weights by solving
a small optimization problem
(usually least-squares)

[Acceleration Methods, d'Aspremont, Scieur, Taylor]

Momentum intuition and much more



https://distill.pub/2017/momentum/

All deep learning optimization algorithms
are based on
Momentum/Acceleration:

RMSprop, AdaGrad, Adam, etc.

Summary of nonlinear optimization

Nonlinear optimization

Optimality conditions

- KKT optimality conditions
- Subgradient optimality conditions $0 \in \partial f(x^\star)$

General
Necessary

Convex
Necessary and sufficient

First order methods: Moderate accuracy on Large-scale data

- Gradient descent
- Subgradient methods
- Proximal algorithms (e.g., ISTA)
- Operator splitting algorithms (e.g., ADMM)

Convergence rates

Typical rates
(gradient descent, proximal gradient, ADMM, etc.)

- L-smoothness: $O(1/k)$, accelerated $O(1/k^2)$
- μ-strong convexity: $O(\log(1/k))$
- We can always combine **line search**
- Convergence bounds usually in terms of cost function distance

Operator theory

- Helps developing and analyzing serial and distributed algorithms
- Algorithms **always** converge for convex problems (independently from step size)
- Convergence bounds usually in terms of iterates distance

First-order methods

Per-iteration cost

Number of iterations

- Gradient/subgradient method
- Forward-backward splitting (proximal algorithms)
- Accelerated forward-backward splitting
- Douglas-Rachford splitting (ADMM)
- Interior-point methods (not covered)

Large-scale systems

- start with feasible method with cheapest per-iteration cost
- if too many iterations, transverse down the list

Acceleration in nonlinear optimization

Today, we learned to:

- **Derive** lower bounds on cost optimality for first-order methods
- **Accelerate** first-order algorithms by adding momentum term
- **Apply** acceleration schemes to get the best possible convergence
- **Select** the appropriate algorithms to apply in large-scale optimization

Next lecture

- Extensions and nonconvex optimization