A lower bound for stepsize-based acceleration of gradient descent

Jianhao Ma
Tsinghua University

Joint work with Yuxin Chen (University of Pennsylvania)

August 29, 2026

GD for smooth convex functions

Assume $f : \mathbb{R}^d \to \mathbb{R}$ is convex and L-smooth, with a minimizer $x_\star$.

Gradient descent (GD) uses the update

$$\boxed{x_{t+1} = x_t - \eta_t \nabla f(x_t)}$$

GD for smooth convex functions

Assume $f : \mathbb{R}^d \to \mathbb{R}$ is convex and L-smooth, with a minimizer $x_\star$.

Gradient descent (GD) uses the update

$$\boxed{x_{t+1} = x_t - \eta_t \nabla f(x_t)}$$

Textbook result (Bubeck, 2015)

For the constant stepsize $\eta_t = 1/L$, the last iterate satisfies

$$f(x_T) - f(x_\star) \;\leq\; \frac{2L\,\|x_0 - x_\star\|^2}{T} \;=\; O\big(T^{-1}\big)\,.$$

Nesterov's GD

Starting from $x_0 = y_0$, Nesterov's method uses

$$\boxed{\begin{aligned} x_t &= y_{t-1} - \frac{1}{L}\nabla f(y_{t-1}), \\ y_t &= x_t + \frac{t-1}{t+2}(x_t - x_{t-1}) \end{aligned}}$$

Nesterov's GD

Starting from $x_0 = y_0$, Nesterov's method uses

$$\boxed{\begin{aligned} x_t &= y_{t-1} - \frac{1}{L}\nabla f(y_{t-1}), \\ y_t &= x_t + \frac{t-1}{t+2}(x_t - x_{t-1}) \end{aligned}}$$

Theorem (Nesterov, 1983)

After T iterations, the last iterate of Nesterov's GD satisfies

$$f(x_T) - f(x_\star) \;\leq\; \frac{2L\,\|x_0 - x_\star\|^2}{(T+1)^2} \;=\; O\big(T^{-2}\big)\,.$$

Nesterov's GD

Theorem (Nesterov, 1983)

After T iterations, the last iterate of Nesterov's GD satisfies

$$f(x_T) - f(x_\star) \leq \frac{2L\|x_0 - x_\star\|^2}{(T+1)^2} = O(T^{-2}).$$

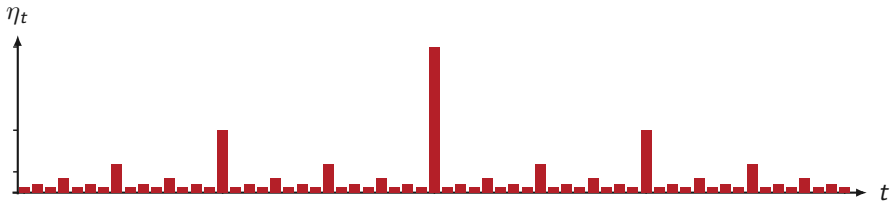
- The $O(T^{-2})$ rate is optimal in the first-order oracle model.
- Can plain GD attain this accelerated rate using stepsizes alone?

GD with silver stepsize

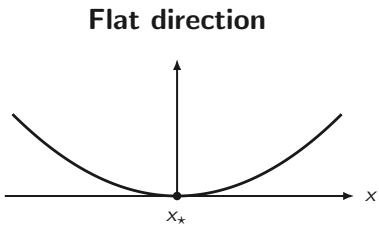
Result (Altschuler–Parrilo, 2025; Grimmer–Shu–Wang, 2025)

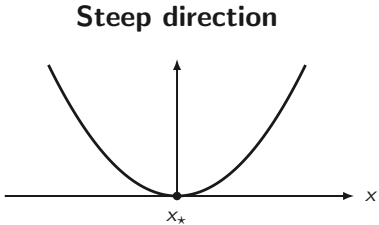
For every convex L-smooth objective, plain GD with the silver stepsize schedule achieves

$$f(x_T) - f(x_\star) = O\left(\frac{L\|x_0 - x_\star\|^2}{T^{\log_2(1+\sqrt{2})}}\right) = O\left(\frac{L\|x_0 - x_\star\|^2}{T^{1.2716\ldots}}\right).$$

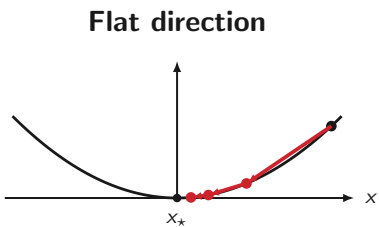


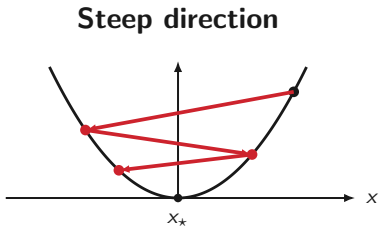
Main idea: stepsize hedging





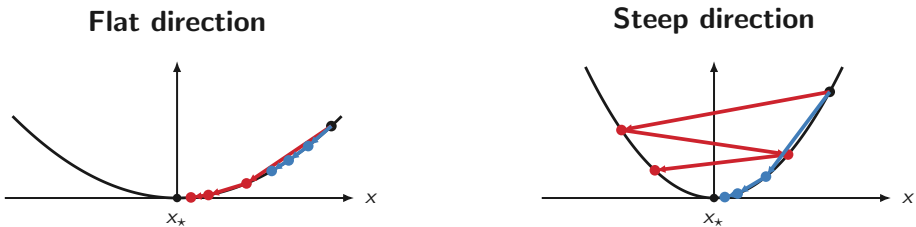
Main idea: stepsize hedging





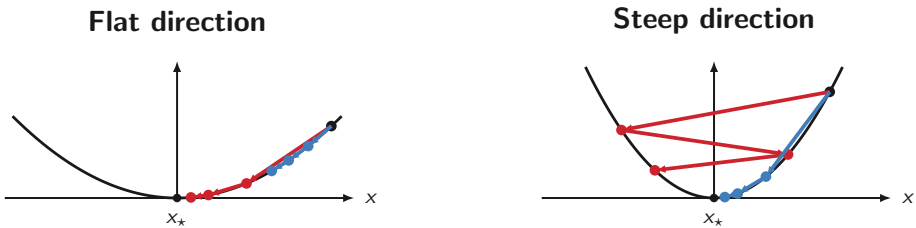
- Large stepsizes converge faster along low-curvature (flat) directions.

Main idea: stepsize hedging



- Large stepsizes converge faster along low-curvature (flat) directions.
- Small stepsizes converge faster along high-curvature (steep) directions.

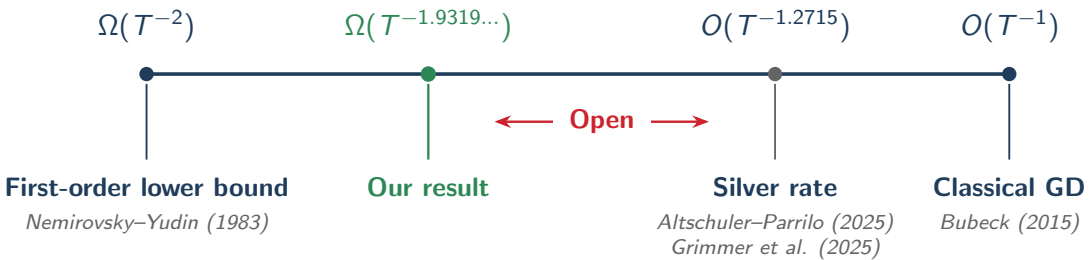
Main idea: stepsize hedging



- Large stepsizes converge faster along low-curvature (flat) directions.
- Small stepsizes converge faster along high-curvature (steep) directions.
- **Hedging both long and short stepsizes leads to acceleration.**

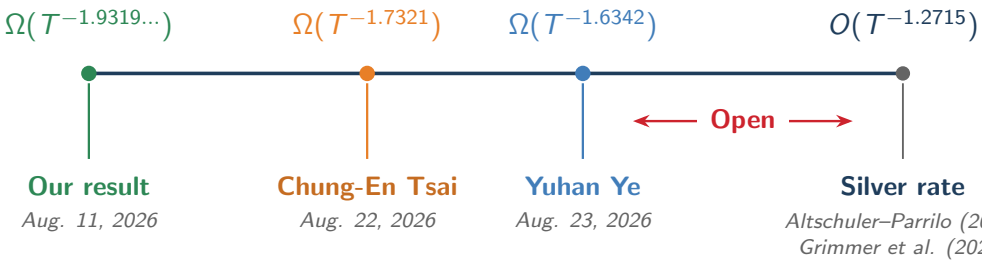
Open problem

Our result



Proof found by **GPT-5.6 Sol Pro**; verified by the authors; formalized in **Lean**.

Latest progress



Formal result

Theorem 1 (Ma–Chen, 2026)

For every $p \in (1.9319\ldots, 2)$, there is $c_p > 0$ such that, for every $T \geq 1$, $L > 0$, every predetermined nonnegative stepsize schedule $\eta \in \mathbb{R}^T_{\geq 0}$, and every prescribed x_0, there exist $d \leq T+1$, a function $f : \mathbb{R}^d \to \mathbb{R}$ that is convex and L-smooth, and $x_\star \in \arg\min f$ for which

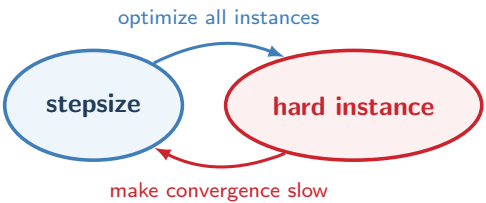
$$f(x_T) - f(x_\star) \geq \frac{c_p L \|x_0 - x_\star\|^2}{(T+1)^p}.$$

- the guarantee concerns the last iterate x_T;
- the stepsize schedule is predetermined and nonnegative;
- the theorem covers $p > 1.9319\ldots$, not the endpoint.

Talk roadmap

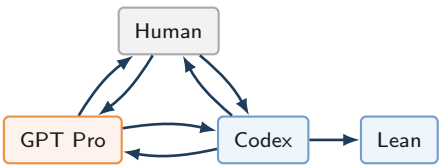
PART I

The hard instance



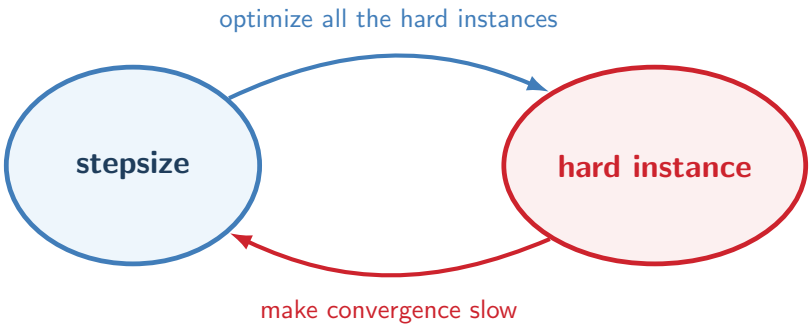
PART II

Human–AI workflow



Part I

The hard instance



First candidate: quadratics

$$q_{\text{chain}}(x) := \frac{1}{2}\left[x_1^2 + \sum_{i=1}^{m-1}(x_i - x_{i+1})^2 + x_m^2\right] - x_1$$

smooth convex	$f(x) = c\, q_{\text{chain}}(x)$	quadratic zero-chain
smooth strongly convex	$f(x) = c\, q_{\text{chain}}(x) + \frac{\mu}{2}\|x\|^2$	quadratic + curvature floor

Canonical constructions: Nesterov (2004, §§2.1.2 and 2.1.4).

Quadratics are simple

Theorem (Chebyshev acceleration) Nemirovsky (1992)

Let $f(x) = \frac{1}{2}x^\top Ax - b^\top x$ with $0 \preceq A \preceq LI$ and minimizer $x_\star$. For any prescribed horizon T, plain GD with stepsize $\eta_t = \frac{1}{L\sin^2\left(\frac{t\pi}{2T+1}\right)}$ satisfies

$$f(x_T) - f^\star \leq \frac{L\|x_0 - x_\star\|^2}{2(2T+1)^2} = O(T^{-2}).$$

Larger stepsizes accelerate convergence

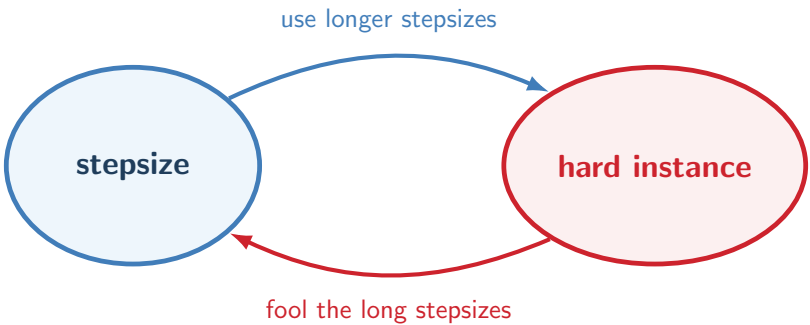
Theorem (bounded-step quadratic obstruction) Tsai et al. (2026, Lemmas 2.1, 2.3)

For fixed $C > 0$, $\alpha \geq 0$, and any stepsize schedule satisfying $\max_{1 \leq t \leq T} \eta_t \leq CT^\alpha / L$, the worst-case last-iterate error over convex L-smooth quadratics is

$$f(x_T) - f^\star = \Omega\left(\frac{L \|x_0 - x_\star\|^2}{T^{\min\{1+\alpha/2, 2\}}} \right).$$

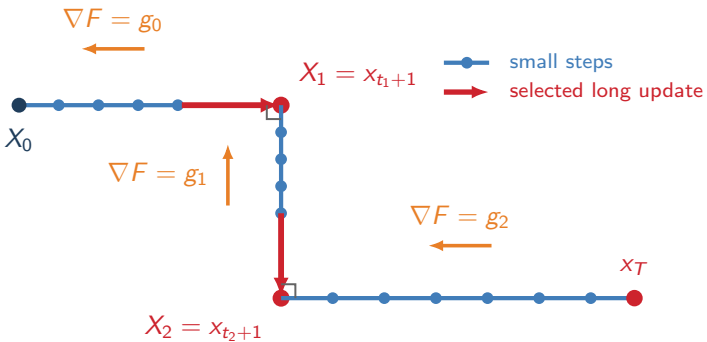
- We need to find a non-quadratic hard instance.
- We only need to consider long stepsizes.

Resisting oracle approach



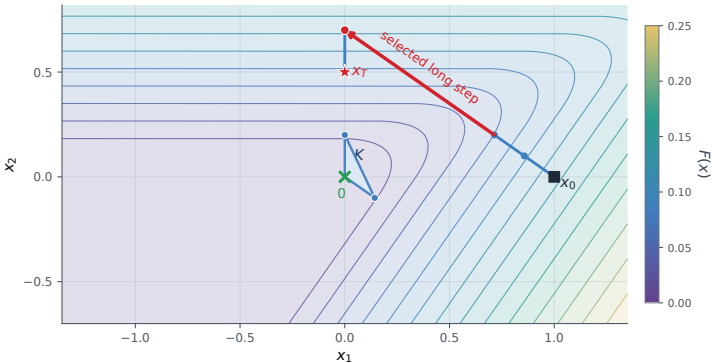
The hard trajectory

We try to hide as much information as possible from the (selected) long stepsizes.



The hard function

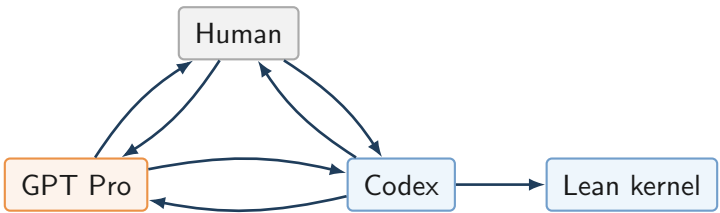
$$F(x) = \frac{1}{2}\|x\|^2 - \frac{1}{2}\operatorname{dist}(x, K)^2, \qquad K = \operatorname{conv}\{0, g_0, \ldots, g_m\}, \qquad \nabla F(x) = \Pi_K(x).$$



Two-dimensional example for $H = (1, 1, 5, 1)$; geometry redrawn from Tsai (2026, Fig. 4).

Part II

Human–AI workflow



Fun facts

- Initially, we tried to prove the lower bound for strongly convex functions, but failed.

Fun facts

- Initially, we tried to prove the lower bound for strongly convex functions, but failed.
- We used GPT-5.5 Pro but did not succeed. Soon after GPT-5.6 was released, we obtained an initial proof.

Fun facts

- Initially, we tried to prove the lower bound for strongly convex functions, but failed.
- We used GPT-5.5 Pro but did not succeed. Soon after GPT-5.6 was released, we obtained an initial proof.
- The proof was not derived in a single attempt; it emerged through repeated attempts and revisions.

Fun facts

- Initially, we tried to prove the lower bound for strongly convex functions, but failed.
- We used GPT-5.5 Pro but did not succeed. Soon after GPT-5.6 was released, we obtained an initial proof.
- The proof was not derived in a single attempt; it emerged through repeated attempts and revisions.
- Along the way, we proposed several proof strategies, but ultimately none were adopted.

Fun facts

- Initially, we tried to prove the lower bound for strongly convex functions, but failed.
- We used GPT-5.5 Pro but did not succeed. Soon after GPT-5.6 was released, we obtained an initial proof.
- The proof was not derived in a single attempt; it emerged through repeated attempts and revisions.
- Along the way, we proposed several proof strategies, but ultimately none were adopted.
- Understanding the proof was difficult: it took longer than finding it.

Human effort

1. **Prompting:** We supplied the problem statement and the general proof idea.

2. **Pruning:** During discussions with ChatGPT, we acted as a pruner and terminated unpromising approaches early.

3. **Proofreading:** We proofread extensively and invested substantial effort in making the proof readable.

What is Lean?

Lean is a proof assistant in which mathematical statements and proofs are written as code and checked by a small trusted kernel.

Example: every square is nonnegative

```
import Mathlib

theorem square_nonnegative (x : Real) :
    0 <= x^2 := by
  positivity
```

Can I use Lean if I don't know it?

When can we trust Lean?

1. Audit the statement

Did we formalize the theorem we intended?

2. Audit the dependencies

Check the assumptions, axioms, and imported results.

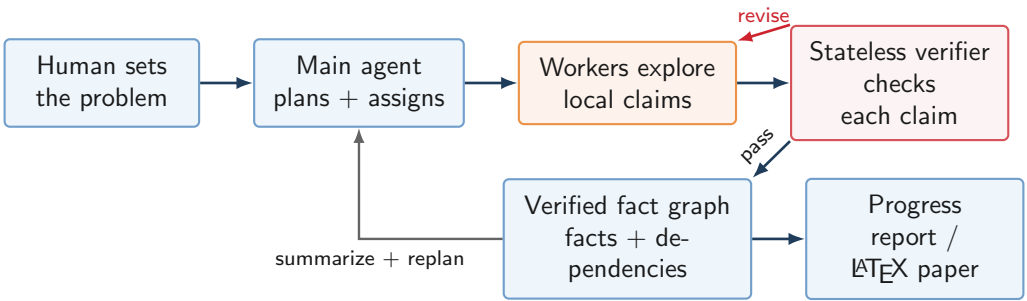
What the kernel certifies

If Lean accepts the complete proof code, the derivation is correct relative to the audited statement and dependencies.

In a human–AI proof loop, Lean can also return goals and diagnostics.

A principled workflow: Danus as an example

Following Liu et al. (2026), parallel proof search is organized around a verifier-gated fact graph.



Takeaways

1. **An interesting result:** Stepsizes alone cannot accelerate plain GD all the way to the optimal $O(1/T^2)$ rate.

2. **A powerful construction:** It extends beyond our setting—even to anytime GD; see the blogs by Tsai (2026) and Ye (2026).

3. **A powerful proof tool:** LLMs can accelerate proof search, exposition, and formal verification.

Paper: arXiv:2608.10418 Lean: github.com/jianhaoma/gd-lower-bound-lean