

Stepsize Hedging: An Alternative Mechanism for Accelerating Gradient Descent¹

Jason M. Altschuler Pablo A. Parrilo
UPenn LIDS - MIT

1 Introduction

Consider the following question that arises in every introductory course on convex optimization:

What is the optimal choice of stepsizes for gradient descent?

Gradient descent (GD) is arguably the most foundational algorithm for continuous optimization and has a commensurately extensive literature. GD was first proposed by Cauchy in the 1800s and remains the workhorse in large-scale optimization applications across engineering, data science, and artificial intelligence. Yet, the fundamental question stated above—how to choose the sole parameter in GD—remained open even in seemingly simple convex settings, highlighting the potential for untapped algorithmic opportunities in those foundational settings and beyond.

Short or long stepsizes? Let us begin by recalling the definition of GD:

$$x_{t+1} = x_t - \alpha_t \nabla f(x_t). \quad (1)$$

The basic idea is that the negative gradient is a descent direction, meaning that if one moves a small amount in this direction, then the value of the objective f decreases. This follows immediately by a Taylor expansion. The algorithmic question at hand is: *how far* should one move? A small stepsize α_t guarantees progress, but the progress will be correspondingly small. A large stepsize α_t is a natural idea to get more mileage out of the descent direction, but this may overshoot the optimum.

Mainstream prescription for smooth convex optimization: constant stepsize schedules. A natural way to quantify this tradeoff is to optimize the progress made in a single iteration. In the standard setup of smooth convex optimization, this one-step question has a well-known answer:

For one iteration of GD, there is a stepsize $\bar{\alpha}$ that achieves the fastest worst-case convergence rate. (2)

This stepsize is explicit.² This one-step guarantee underlies the textbook prescription: use the constant schedule $\alpha_t = \bar{\alpha}$ at every iteration t . See for example the textbooks [8, 11, 12, 25, 27, 31, 35].

However, even after optimizing $\bar{\alpha}$, this constant stepsize schedule may lead to slow convergence. This has motivated an extensive literature on *accelerated* first-order methods which famously modify GD by adding momentum or other internal dynamics, see the survey [19]. Many alternative stepsize schedules have also been proposed—for example exact line search, Armijo-Goldstein rules, Polyak-type schedules, and Barzilai-Borwein-type schedules—but none had led to an analysis that improves over the textbook constant-stepsize rate. The resulting conventional wisdom was that GD cannot be accelerated merely by changing its stepsizes.

Faster convergence via hedged stepsizes? Is this conventional wisdom correct? Surprisingly, the answer is no. The key observation is that optimality of the stepsize $\bar{\alpha}$ for one iteration does *not* imply optimality of repeating the stepsize $\bar{\alpha}$ for multiple iterations. The stepsize $\bar{\alpha}$ is chosen to protect against the worst case for a single iteration. But when GD is run for multiple iterations, the bad instances for different stepsizes need not be compatible with one another. A short step may be too conservative on one kind of instance (flat objectives f), while a long step may overshoot on another (steep objectives f); over several iterations, these opposing weaknesses can fail to align (since convex objectives f are rigid and cannot change curvature arbitrarily). This suggests a multi-step opportunity:

*Can one combine stepsizes that are individually suboptimal
to obtain faster convergence over many iterations?* (3)

We refer to this idea as *stepsize hedging*: rather than optimizing each step in isolation, choose a time-varying schedule that hedges between different worst-case behaviors.

¹This expository article will appear as an invited Research Highlight in the 2026 INFORMS Computing Society Newsletter.

²For example, $\bar{\alpha} = 1/M$ for M -smooth convex objectives, and $\bar{\alpha} = 2/(M + m)$ for objectives that are also m -strongly convex.

Motivation: the special case of quadratics. Time-varying stepsizes were initially explored (only) in the special case of convex quadratic optimization. A classic result due to Young in 1953 [50] shows that in this setting, the optimal stepsizes are non-constant and related to the roots of Chebyshev polynomials. This elegant result is recalled in detail in §2. However, many core phenomena in the quadratic setting do not extend beyond. For example, these Chebyshev stepsizes can make GD diverge beyond the special case of quadratics. Over the past 70 years, the continuous optimization community has devoted significant effort to developing alternative stepsize schedules beyond quadratic optimization. These are often empirically helpful. However, despite significant effort and many candidate stepsize schedules (even adaptive), there was no theoretical evidence that *any* stepsize schedule could lead to *any* speedup over the textbook GD convergence rate for (non-quadratic) convex optimization.

1.1 Main result

In the past decade, a line of work has shown that such an algorithmic opportunity persists: time-varying stepsizes can accelerate GD for (non-quadratic) convex optimization. The first such result was shown in the thesis [2] of the first author (advised by the second author), and an exciting flurry of ensuing work has sought to push this algorithmic opportunity to its limit. See the discussion of related work in §1.2.

We summarize here the results of the two papers [3, 4]. In these papers, we propose an unconventional stepsize schedule and show that it accelerates GD for smooth convex optimization. We term this the *silver stepsize schedule* due to the occurrence of the *silver ratio* $\rho = 1 + \sqrt{2}$. It is explicit and non-adaptive, see §4 for the definition and a full discussion. Here we highlight the main results, namely the accelerated convergence rates that this enables for the convex and strongly-convex settings, respectively.

Below, for shorthand, we say that an iterative algorithm initialized at x_0 minimizes an M -smooth convex function f to ε error if its final iterate x_n satisfies $\frac{f(x_n) - f(x^*)}{M\|x_0 - x^*\|^2} \leq \varepsilon$, where x^* denotes a minimizer of f . The factor of $M\|x_0 - x^*\|^2$ normalizes the performance to make it scale-invariant and meaningful.

Theorem 1 (Silver stepsizes for convex optimization [4]). *Fix any accuracy ε and smoothness parameter M . There exists a stepsize schedule of length*

$$n \asymp \varepsilon^{-\log_\rho 2} \approx \varepsilon^{-0.7864}$$

such that GD ε -minimizes any M -smooth convex function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ in any dimension d .

Theorem 2 (Silver stepsizes for strongly convex optimization [3]). *Fix any accuracy ε , strong convexity parameter m , and smoothness parameter M . There exists a stepsize schedule of length*

$$n \asymp \kappa^{\log_\rho 2} \log \frac{1}{\varepsilon} \approx \kappa^{0.7864} \log \frac{1}{\varepsilon}$$

such that GD ε -minimizes any m -strongly convex, M -smooth function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ in any dimension d . Here $\kappa = M/m \geq 1$ denotes the condition number.

For comparison, Theorem 1 improves the textbook rate of constant-step GD from $O(\varepsilon^{-1})$ to roughly $O(\varepsilon^{-0.7864})$. In the strongly convex setting, Theorem 2 analogously improves the textbook rate of GD from $O(\kappa \log \frac{1}{\varepsilon})$ to roughly $O(\kappa^{0.7864} \log \frac{1}{\varepsilon})$. These results establish stepsize hedging as an alternative mechanism for accelerating GD. Interestingly this does not match the fully accelerated rate obtained by modifying GD beyond stepsizes [32]. Our asymptotic rates are conjectured optimal among all (non-adaptive³) stepsize schedules.

1.2 Related work

There is an extensive literature on accelerated first-order algorithms, e.g., [19, 30, 31]. As opposed to the mainstream approach which modifies GD, we focus here on an alternative mechanism for accelerating GD: time-varying stepsize schedules.

As mentioned above, the benefit of time-varying stepsizes was classically known for the special case of quadratic optimization since Young in 1953 [50], but remained open beyond quadratics. The first such results were by Altschuler [2], who showed that time-varying hedging can lead to improved rates by analyzing

³A non-adaptive schedule is a predetermined sequence $\alpha_0, \alpha_1, \dots$ that may depend on problem parameters such as M, m, n , but not on the observed iterates or gradients. In contrast, an adaptive schedule may choose α_t using information from earlier iterations.

	Quadratic	Convex
Mainstream stepsizes	$\Theta(\kappa)$ by constant stepsizes (folklore)	$\Theta(\kappa)$ by constant stepsizes (folklore)
Optimized stepsizes	$\Theta(\sqrt{\kappa})$ by Chebyshev Stepsizes [50]	$O(\kappa^{\log_\rho 2})$ by Silver Stepsizes [3] (Theorem 2)
Additional dynamics	$\Theta(\sqrt{\kappa})$ by Heavy Ball [34]	$\Theta(\sqrt{\kappa})$ by Nesterov Acceleration [32]

Table 1: Iteration complexity of various approaches for minimizing a κ -conditioned convex function. The dependence on the accuracy ε is omitted as it is always $\log 1/\varepsilon$. The story is analogous without strong convexity: rates of the form $O(\kappa^a \log 1/\varepsilon)$ here correspond to rates of the form $O(\varepsilon^{-a})$ there. This note focuses on a foundational question: how much mileage can one obtain from optimizing the stepsizes of GD?

$n = 2, 3$ in the strongly convex setting. Daccache [16] and Eloi [20] exhaustively extended these $n = 2, 3$ results to related settings and performance metrics. An exciting line of subsequent work numerically optimized longer schedules. Hedging over larger horizons n further improves the convergence rate, but searching for optimal stepsizes is a non-convex problem that is computationally difficult as n increases. Das Gupta et al. [17] developed a branch-and-bound framework to compute good schedules up to $n = 50$ for smooth convex optimization. Grimmer [21] developed a technique to round these branch-and-bound solutions to exact rational certificates; this allowed him to extend these approximate stepsize schedules up to $n = 127$ in order to get a larger constant-factor improvement.

How far can this be pushed? As n increases (beyond constants), can this time-varying hedging lead to *asymptotic* acceleration? What are the optimal stepsizes? Intriguing fractal-like patterns were observed for small n ; do these qualitative behaviors persist for large n ?

The two papers [3, 4] highlighted in this note answered this asymptotic-acceleration question affirmatively by introducing the *silver stepsize schedule*. This schedule is built recursively from the same short-step/long-step splitting pattern that appears in the optimal two-step schedule of [2]. In the smooth convex setting (Theorem 1), the silver stepsizes improve the iteration complexity of GD from the textbook rate $O(\varepsilon^{-1})$ to $O(\varepsilon^{-\log_\rho 2}) \approx O(\varepsilon^{-0.78})$. In the m -strongly convex and M -smooth setting (Theorem 2), they similarly improve the dependence on the condition number $\kappa = \frac{M}{m}$ from $O(\kappa \log \frac{1}{\varepsilon})$ for constant stepsizes to $O(\kappa^{\log_\rho 2} \log \frac{1}{\varepsilon}) \approx O(\kappa^{0.78} \log \frac{1}{\varepsilon})$. These asymptotic rates were conjectured to be asymptotically optimal among all non-adaptive stepsize schedules [3, 4]. (Concurrent work by Grimmer et al. [22] proved asymptotic acceleration using a different stepsize schedule, but at a suboptimal convergence rate, with exponent ≈ 0.95 rather than $\log_\rho 2 \approx 0.78$.)

Excitingly, in only the few years since, many papers have followed up; see §5 for a discussion of this burgeoning area.

2 Optimal stepsizes for quadratic optimization (Young 1953)

As mentioned above, in the special case of *quadratic* optimization, it is classically known that time-varying stepsizes can accelerate GD. This result is due to Young [50] and is nowadays taught in introductory optimization courses. We briefly recall this elegant argument below as it provides perhaps the simplest example of the broader algorithmic opportunity.

Young’s stepsizes. Consider running GD on the class \mathcal{F} of quadratic functions f that are m -strongly convex and M -smooth. What stepsize schedules make GD converge

$$\|x_n - x^*\| \leq R_n \|x_0 - x^*\| \tag{4}$$

at the fastest possible rate R_n ? Without loss of generality after translating, $f(x) = \frac{1}{2}x^T Hx$ where $mI \preceq H \preceq MI$. Since f is quadratic, its gradient is linear $\nabla f(x) = Hx$, hence GD is a linear map $x_{t+1} = x_t - \alpha_t \nabla f(x_t) = (I - \alpha_t H)x_t$, and therefore the n -th iterate is

$$x_n = p_n(H)x_0, \quad \text{where} \quad p_n(H) = \prod_{t < n} (I - \alpha_t H). \tag{5}$$

Observe that as one ranges over all possible choices of the stepsize schedule $\{\alpha_t\}_{t < n}$, the polynomial p_n ranges over the set \mathcal{P}_n of all degree- n polynomials satisfying the normalizing condition $p_n(0) = 1$. Therefore finding an optimal stepsize schedule is equivalent to finding an optimal polynomial $p_n \in \mathcal{P}_n$.

What is the optimal polynomial? By the above display and basic properties of the spectral norm,

$$R_n = \sup_{f \in \mathcal{F}, x_0 \neq x^*} \frac{\|x_n - x^*\|}{\|x_0 - x^*\|} = \sup_{mI \preceq H \preceq MI, x_0 \neq 0} \frac{\|p_n(H)x_0\|}{\|x_0\|} = \sup_{mI \preceq H \preceq MI} \|p_n(H)\| = \sup_{m \leq \lambda \leq M} |p_n(\lambda)|.$$

Thus the optimal polynomial $p_n \in \mathcal{P}_n$ is the one with minimal L_∞ norm over the interval $[m, M]$. It is classically known that this is the (translated and scaled) Chebyshev polynomial of the first kind, see e.g., [36]. By the definition of p_n in (5), it follows that the optimal stepsizes $\{\alpha_t\}_{t < n}$ are the inverses of the roots of this Chebyshev polynomial, in any order. See Figure 1 for an illustration and an interpretation of this phenomenon through the lens of *hedging*.

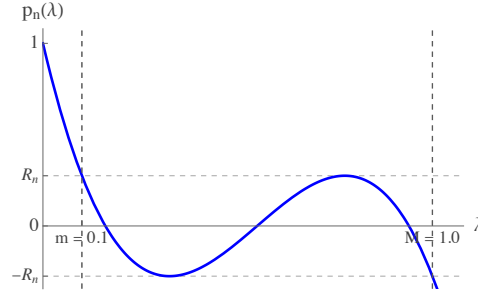


Figure 1: The translated and scaled Chebyshev polynomial $p_n \in \mathcal{P}_n$ has minimal L_∞ norm on $[m, M]$. Here visualized for $n = 4$, $m = 0.1$, $M = 1$. In quadratic optimization, the error after GD with stepsizes $\{\alpha_t\}_{t < n}$ is governed by the polynomial $p_n(\lambda) = \prod_{t < n} (1 - \alpha_t \lambda)$. Chebyshev stepsizes hedge across curvatures $\lambda \in [m, M]$ by making the worst-case errors $p_n(\lambda)$ equioscillate between $\pm R_n$.

Beyond quadratics? This argument establishes that time-varying stepsizes can accelerate GD for quadratic f , but does this algorithmic opportunity extend beyond? From an analysis perspective, the argument fails from the very beginning: if f is not quadratic, then GD is not a linear map, and the equivalence to polynomials fails. In fact, this issue is not merely an artifact of analysis techniques: fundamental phenomena for the quadratic setting are false beyond. For example, in (non-quadratic) convex optimization, these Chebyshev stepsizes lead to divergence, the stepsize order matters, stepsizes and momentum are not equivalent, etc. As a result, it was unknown if *any* stepsize schedule could lead to *any* improvement over the constant stepsize schedule in the general setting of (non-quadratic) convex optimization. Is this a missed opportunity?

3 Optimal stepsizes for convex optimization, $n = 2$ (Altschuler 2018)

The answer is yes. This was first shown in [2]. In fact, that thesis showed several such results; we state here the simplest one.

Consider $n = 2$ steps, the minimal setting where time-varying stepsizes could possibly be advantageous. What two stepsizes α_0, α_1 make GD converge

$$\|x_2 - x^*\| \leq R_2 \|x_0 - x^*\| \tag{6}$$

at the fastest possible rate R_2 ? In the worst case over objectives f that are m -strongly convex and M -smooth? Obviously $R_2 \leq R_1^2$ is possible by using $\alpha_0 = \alpha_1 = \bar{\alpha}$. Can strictly faster convergence $R_2 < R_1^2$ be achieved? With time-varying stepsizes $\alpha_0 \neq \alpha_1$? Remarkably, the answer is yes:

Theorem 3 (Theorem 8.10 of [2]). *The stepsizes α_0, α_1 that make R_2 as small as possible in (6) are unique, time-varying ($\alpha_0 \neq \alpha_1$), and strictly improve over the constant stepsize schedule ($R_2 < R_1^2$).*

As a trivial corollary, cyclically repeating $(\alpha_0, \alpha_1, \alpha_0, \alpha_1, \alpha_0, \alpha_1, \dots)$ strictly improves over the textbook rate for constant stepsizes. Indeed, the two-step convergence (6) implies $\|x_{2n} - x^*\| \leq R_2^n \|x_0 - x^*\|$, which strictly improves over the standard rate $\|x_{2n} - x^*\| \leq R_1^{2n} \|x_0 - x^*\|$ for constant stepsizes. See Figure 2.

The precise values for α_0, α_1, R_2 are complicated and not essential for this brief exposition⁴. We only highlight two features of these explicit values that will be built upon later:

⁴For the interested reader, the optimal stepsizes are $\alpha_0 = \frac{2}{m+S}$ and $\alpha_1 = \frac{2}{2M+m-S}$, with corresponding rate $R_2 = \frac{S-M}{2m+S-M}$, where $S = \sqrt{M^2 + (M-m)^2}$.

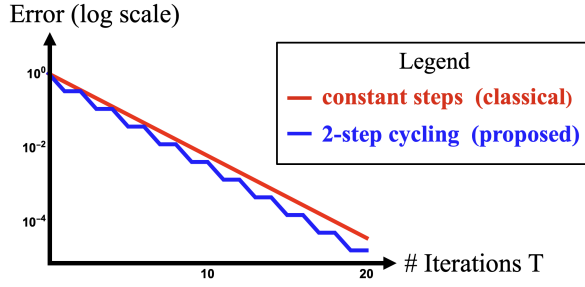


Figure 2: Illustration of Theorem 3: time-varying stepsizes that cycle between 2 stepsizes provably speed up GD for (non-quadratic) convex optimization. Here consider minimizing functions that are 1-strongly convex and 4-smooth. **Red:** the standard prescription of constant stepsizes $\alpha_t = 2/5$ in each iteration t leads to a convergence rate of 0.6^T after T iterations. **Blue:** better is cycling between $\alpha_t = 1/3$ and $1/2$ in even and odd steps; this leads to a faster convergence rate of $(1/\sqrt{3})^T \approx 0.57^T$.

- **Stepsize splitting.** The optimal stepsizes $\alpha_0 < \bar{\alpha} < \alpha_1$ are obtained by “splitting” the constant stepsize $\bar{\alpha}$ into a shorter step α_0 and a longer step α_1 as the two roots of a certain explicit quadratic equation.
- **Silver ratio.** The improvement $R_2 < R_1^2$ can be quantified as $R_2 \approx 1 - \frac{2(1+\sqrt{2})}{\kappa}$ whereas $R_1^2 \approx 1 - \frac{4}{\kappa}$, in the relevant asymptotic regime as $\kappa \rightarrow \infty$. This quantity $\rho = 1 + \sqrt{2}$ is called the silver ratio.

Summarizing, Theorem 3 shows a missed algorithmic opportunity: time-varying stepsizes provably make GD converge faster. However, this result only shows a constant factor improvement in the final iteration complexity (from $R_2 < R_1^2$) since it only considers $n = 2$ steps. The improvement increases in n . But by how much? As $n \rightarrow \infty$, what are the optimal stepsizes and rate? How much faster can one accelerate GD?

4 Silver stepsizes for convex optimization, $n \rightarrow \infty$ (Altschuler-Parrilo 2023)

The papers [3, 4] developed the *silver stepsizes* in order to prove acceleration for arbitrarily large horizons $n \rightarrow \infty$. The silver stepsizes accelerate the iteration complexity of GD from the textbook rate $O(\varepsilon^{-1})$ to $O(\varepsilon^{-\log_\rho 2})$ in the convex setting (Theorem 1), and analogously from $O(\kappa \log \frac{1}{\varepsilon})$ to $O(\kappa^{\log_\rho 2} \log \frac{1}{\varepsilon})$ in the strongly convex setting (Theorem 2). Here $\rho = 1 + \sqrt{2}$ denotes the silver ratio. In these papers we conjectured that our asymptotic rate is optimal among (non-adaptive) stepsize schedules.⁵

The silver stepsizes are constructed in a recursive manner from the 2-step solution in Theorem 3, see Figure 3. This leads to highly unconventional features: the silver stepsize schedule is non-monotonically time-varying, fractal-like, and has peaks that are exponentially infrequent but exponentially large. In the strongly-convex setting, the silver stepsizes are also approximately periodic with period of size $\kappa^{\log_\rho 2}$. See [3] for details.

The silver stepsize schedule simplifies in the (non-strongly) convex setting. See Figure 4. This is the formal limit as the strong convexity parameter $m \rightarrow 0$, or equivalently $\kappa \rightarrow \infty$. The resulting schedule has simple direct definitions, both recursively and explicitly. For simplicity, below we normalize the smoothness $M = 1$ (without loss of generality since running GD on f amounts to rescaling the stepsizes by $1/M$).

- **Recursive definition.** For any integer $n = 2^k - 1$, we recursively construct the schedule h_{2n+1} of length $2n + 1$ from the schedule h_n of length n via

$$h_{2n+1} = [h_n, 1 + \rho^{k-1}, h_n] \quad (7)$$

with base case $h_1 = [\sqrt{2}]$. This results in the pattern $[\sqrt{2}, 2, \sqrt{2}, 1 + \sqrt{2}, \dots]$ depicted in Figure 4.

- **Explicit definition.** The t -th stepsize is

$$\alpha_t = 1 + \rho^{\nu(t+1)-1} \quad (8)$$

⁵If one believes the optimality of our asymptotic rates, then one can ask even more fine-grained questions about the hidden constant in the big-O notation. We further conjecture that the silver stepsize schedule is exactly optimal in the strongly convex setting for the performance metric $\|x_n - x^*\|/\|x_0 - x^*\|$ that it was originally designed for. In the convex setting, an interesting line of follow-up work has improved the hidden-constant by almost a factor of 3 and presented conjecturally optimal stepsizes for a variety of performance metrics, see the discussion in §5.

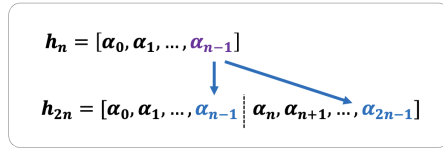


Figure 3: Schematic of how the silver stepsize schedule is recursively constructed in the strongly convex setting [3]. Let h_n denote the schedule of length n . Then h_{2n} is built from two copies of h_n , but with a key modification of the final step α_{n-1} in h_n (purple). It is “split” into a shorter step α_{n-1} and longer step α_{2n-1} (blue), given by the two roots of a quadratic equation, exactly analogous to the 2-step construction in §3.

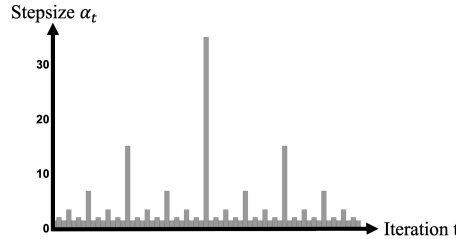


Figure 4: The silver stepsize schedule we proposed for convex optimization [3, 4]. The first 63 stepsizes are shown. They are highly unconventional: time-varying, non-monotonic, fractal-like, and sometimes use extremely aggressive stepsize “spikes” of size $\alpha_{2^k-1} = 1 + \rho^{k-1}$ where $\rho = 1 + \sqrt{2}$ denotes the silver ratio.

where $\nu(t)$ denotes the 2-adic valuation of t , i.e., the largest non-negative integer i such that 2^i divides t . For example, $\nu(1) = 0$, $\nu(2) = 1$, $\nu(3) = 0$, $\nu(4) = 2$, etc.

Algorithm analysis: multi-step descent. The core technical challenge is that hedging requires deviating from standard analyses, which bound the progress of each step individually (e.g., using the descent lemma) and then sum the progress. Such *one-step* analyses are too shortsighted to prove acceleration. Instead, one must directly analyze the *multi-step progress* in order to show that different iterations of GD synergize with each other. These holistic analyses are necessary for proving any benefit of time-varying stepsizes.

Of course, directly analyzing the long-term progress is much more difficult. This involves answering two interrelated questions, both of which are non-trivial:

- *Analysis question:* how to analyze a given stepsize schedule?
- *Design question:* how to design the optimal stepsize schedule?

Our starting point is the performance estimation problem (PEP) framework, pioneered by [18] and refined by [44], which enables numerically solving the analysis question using semidefinite programming (SDP). PEP addresses the analysis question by linearly combining valid inequalities for the class of functions under consideration; in the convex case, these are the cocoercivities relating function values and gradients at different points. Although PEP is helpful, it is important to emphasize that it is *not* a complete solution to the problem at hand. One challenge is that PEP does not fully solve the analysis question: it only produces a numerical estimate of the convergence rate for a fixed number of iterations n , whereas establishing acceleration requires rigorous symbolic proofs for arbitrarily large n . The biggest limitation, however, is that PEP does not directly tackle the design question: finding the stepsize schedule with fastest convergence is non-convex in all existing PEP formulations⁶. This poses a key difficulty for discovering hedging strategies.

Our work [3, 4] proposes a technique called *recursive gluing* which helps make both the design and analysis questions tractable for arbitrarily large n . In terms of the design question, as described above, we recursively construct the silver stepsize schedule of size $2n$ by gluing together two copies of the schedule of length n , modulo modification to a constant number of stepsizes. A key insight is that we can leverage this recursive structure of the stepsize schedule in order to also recursively approach the analysis question: we show that one can “recursively glue” two copies of the convergence proof for the smaller schedules, modulo modification to a constant number of inequalities. The key point is that the proof complexity does not increase in n . Indeed, in this way, proving the $2n$ -step convergence rate from the n -step rate is no harder than proving the 2-step convergence rate from the textbook 1-step rate. See [3, 4] for full details.

⁶It is an interesting question if there are alternative formulations or parameterizations of the stepsize design question that are convex.

5 Outlook

This general principle of multi-step descent opens up many directions for the design and analysis of algorithms, as it suggests a potential missed opportunity for any optimization algorithm analyzed with traditional one-step analyses. In just the past three years since [3, 4], many papers have already followed up. For example, recent work has showcased the generality of this stepsize-hedging phenomenon by extending the results to anytime convergence [26, 53], constrained and proximal settings [9, 10, 47], Riemannian settings [33], robustness and inexact gradient settings [7, 45, 46], operator-splitting settings [1], min-max settings [40, 41], random stepsizes [5], and other performance metrics [23, 24, 52]. The line of work [23, 24, 47, 52] has also improved the hidden constant in the big-O for the non-strongly convex setting (although still with the same asymptotic rate $\log_{\rho} 2$ as the silver stepsize schedule), and in particular [24, 52] developed composition/concatenation analysis frameworks which simplify and refine the recursive gluing technique of [3, 4] and suggest stepsize schedules that are conjecturally minimax-optimal for several performance metrics. Recent work has also used large stepsizes to obtain faster rates or sharper dynamical understanding for logistic/classification losses, including separable logistic regression, regularized logistic regression, non-separable settings, and related two-layer-network models [6, 13, 14, 15, 28, 29, 48, 49, 51]. The machine learning community has also been excited by intriguing similarities with similar stepsize schedules previously explored in empirical ML, e.g., the cyclical stepsize schedules of [42, 43] as well as learned schedules in parametric optimization [37, 38, 39]. Previously, time-varying stepsizes had no provable benefit even in (non-quadratic) convex optimization; this line of work provides a first step towards explaining the unreasonable effectiveness of time-varying stepsizes in empirical deep learning. But much more is needed to realize the potential of time-varying hedging as a powerful algorithmic primitive in both theory and practice.

References

- [1] Abbaszadehpeivasti H, Zamani M (2025) On the convergence rate of the Douglas-Rachford splitting algorithm. *Preprint at arXiv:2509.06676*.
- [2] Altschuler JM (2018) *Greed, Hedging, and Acceleration in Convex Optimization*. Master’s thesis, Massachusetts Institute of Technology.
- [3] Altschuler JM, Parrilo PA (2025) Acceleration by StepSize Hedging: Multi-Step Descent and the Silver StepSize Schedule. *Journal of the ACM* 72(2):1–38.
- [4] Altschuler JM, Parrilo PA (2025) Acceleration by stepsize hedging: Silver StepSize Schedule for smooth convex optimization. *Mathematical Programming* 213(1–2):1105–1118.
- [5] Altschuler JM, Parrilo PA (2026) Acceleration by random stepsizes: Hedging, equalization, and the arcsine stepsize schedule. *Foundations of Computational Mathematics*, to appear.
- [6] Axiotis K, Sviridenko M (2023) Gradient descent converges linearly for logistic regression on separable data. *International Conference on Machine Learning*, 1302–1319.
- [7] Bai L, Zeng Y, Zhou B (2025) Generalization of silver stepsize schedule to stochastic optimization. *Preprint at arXiv:2511.21917*.
- [8] Bertsekas DP (1999) *Nonlinear programming* (Athena Scientific).
- [9] Bok J, Altschuler JM (2025) Accelerating proximal gradient descent via silver stepsizes. *Conference on Learning Theory*, 421–453.
- [10] Bok J, Altschuler JM (2026) Optimized methods for composite optimization: a reduction perspective. *Mathematical Programming*, to appear.
- [11] Boyd S, Vandenberghe L (2004) *Convex optimization* (Cambridge University Press).
- [12] Bubeck S (2015) Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning* 8(3-4):231–357.
- [13] Cai Y, Wu J, Mei S, Lindsey M, Bartlett PL (2024) Large stepsize gradient descent for non-homogeneous two-layer networks: Margin improvement and fast optimization. *Advances in Neural Information Processing Systems*.

- [14] Crawshaw M, Liu M (2026) Tight bounds for logistic regression with large stepsize gradient descent in low dimension. *Preprint at arXiv:2602.12471*.
- [15] Crawshaw M, Woodworth B, Liu M (2025) Constant stepsize local GD for logistic regression: Acceleration by instability. *Preprint at arXiv:2506.13974*.
- [16] Daccache A (2019) *Performance estimation of the gradient method with fixed arbitrary step sizes*. Master's thesis, Université Catholique de Louvain.
- [17] Das Gupta S, Van Parys BP, Ryu EK (2024) Branch-and-bound performance estimation programming: a unified methodology for constructing optimal optimization methods. *Mathematical Programming* 567–639.
- [18] Drori Y, Teboulle M (2014) Performance of first-order methods for smooth convex minimization: a novel approach. *Mathematical Programming* 145(1–2):451–482.
- [19] d'Aspremont A, Scieur D, Taylor A (2021) Acceleration methods. *Foundations and Trends® in Optimization* 5(1-2):1–245.
- [20] Eloi D (2022) *Worst-case functions for the gradient method with fixed variable step sizes*. Master's thesis, Université Catholique de Louvain.
- [21] Grimmer B (2024) Provably faster gradient descent via long steps. *SIAM Journal on Optimization* 34(3):2588–2608.
- [22] Grimmer B, Shu K, Wang AL (2023) Accelerated gradient descent via long steps. *Preprint at arXiv:2309.09961*.
- [23] Grimmer B, Shu K, Wang AL (2025) Accelerated objective gap and gradient norm convergence for gradient descent via long steps. *INFORMS Journal on Optimization* 7(2):156–169.
- [24] Grimmer B, Shu K, Wang AL (2026) Composing optimized stepsize schedules for gradient descent. *Mathematics of Operations Research*, to appear.
- [25] Hazan E (2016) Introduction to online convex optimization. *Foundations and Trends in Optimization* 2(3-4):157–325.
- [26] Kornowski G, Shamir O (2024) Open problem: Anytime convergence rate of gradient descent. *Conference on Learning Theory*, volume 247.
- [27] Luenberger DG, Ye Y (1984) *Linear and nonlinear programming* (Springer).
- [28] Meng SY, Goujaud B, Orvieto A, De Sa C (2025) Gradient descent on logistic regression: Do large step-sizes work with data on the sphere? *Preprint at arXiv:2507.11228*.
- [29] Meng SY, Orvieto A, Cao DY, De Sa C (2024) Gradient descent on logistic regression with non-separable data and large step sizes. *Preprint at arXiv:2406.05033*.
- [30] Nemirovskii A, Yudin DB (1983) *Problem complexity and method efficiency in optimization* (Wiley).
- [31] Nesterov Y (2013) *Introductory lectures on convex optimization: A basic course*, volume 87 (Springer Science & Business Media).
- [32] Nesterov YE (1983) A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Math. Dokl.* 27(2):372–376.
- [33] Park J, Roy A, Siegel JW, Bhattacharya A (2025) Acceleration via silver step-size on Riemannian manifolds with applications to Wasserstein space. *Advances in Neural Information Processing Systems* 38.
- [34] Polyak BT (1964) Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics* 4(5):1–17.
- [35] Polyak BT (1987) *Introduction to optimization* (Optimization Software, Inc.).
- [36] Rivlin TJ (1981) *An introduction to the approximation of functions* (Courier Corporation).
- [37] Sambharya R, Bok J, Matni N, Pappas G (2025) Learning acceleration algorithms for fast parametric convex optimization with certified robustness. *Preprint at arXiv:2507.16264*.

- [38] Sambharya R, Stellato B (2025) Data-driven performance guarantees for classical and learned optimizers. *Journal of Machine Learning Research* 26(171):1–49.
- [39] Sambharya R, Stellato B (2026) Learning algorithm hyperparameters for fast parametric convex optimization. *SIAM Journal on Mathematics of Data Science*, to appear.
- [40] Shugart H, Altschuler JM (2025) Min-max optimization is strictly easier than variational inequalities. *Preprint at arXiv:2511.03052*.
- [41] Shugart H, Altschuler JM (2025) Negative stepsizes make gradient-descent-ascent converge. *Preprint at arXiv:2505.01423*.
- [42] Smith LN (2017) Cyclical learning rates for training neural networks. *IEEE Winter Conference on Applications of Computer Vision*, 464–472 (IEEE).
- [43] Smith LN, Topin N (2019) Super-convergence: Very fast training of neural networks using large learning rates. *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, volume 11006, 369–386 (SPIE).
- [44] Taylor AB, Hendrickx JM, Glineur F (2017) Smooth strongly convex interpolation and exact worst-case performance of first-order methods. *Mathematical Programming* 161(1-2):307–345.
- [45] Vernimmen P (2024) *Tight convergence analysis of exact and inexact gradient methods with constant and silver schedules*. Master’s thesis, Université Catholique de Louvain.
- [46] Vernimmen P, Glineur F (2025) Empirical and computer-aided robustness analysis of long-step and accelerated methods in smooth convex optimization. *International Conference on Modelling, Computation and Optimization in Information Systems and Management Sciences*, 354–366 (Springer).
- [47] Wang B, Ma S, Yang J, Zhou D (2026) Relaxed proximal point algorithm: Tight complexity bounds and acceleration without momentum. *INFORMS Journal on Optimization* 8(2):141–162.
- [48] Wu J, Bartlett PL, Telgarsky M, Yu B (2024) Large stepsize gradient descent for logistic loss: Non-monotonicity of the loss improves optimization efficiency. *Proceedings of the Thirty Seventh Conference on Learning Theory*, Proceedings of Machine Learning Research (PMLR).
- [49] Wu J, Marion P, Bartlett P (2025) Large stepsizes accelerate gradient descent for regularized logistic regression. *Advances in Neural Information Processing Systems* 38:104485–104525.
- [50] Young D (1953) On Richardson’s method for solving linear systems with positive definite matrices. *Journal of Mathematics and Physics* 32(1-4):243–255.
- [51] Zhang R, Wu J, Bartlett PL (2025) Gradient descent converges arbitrarily fast for logistic regression via large and adaptive stepsizes. *International Conference on Machine Learning*, volume 267, 76361–76384.
- [52] Zhang Z, Jiang R (2026) Accelerated gradient descent by concatenation of stepsize schedules. *SIAM Journal on Optimization*, to appear.
- [53] Zhang Z, Lee J, Du S, Chen Y (2025) Anytime acceleration of gradient descent. *Conference on Learning Theory*, volume 291, 5991–6013.