Adaptive Accelerated Gradient Descent Methods for Convex Optimization

Anonymous authors
Paper under double-blind review

Abstract

This work proposes A^2GD, a novel adaptive accelerated gradient descent method for convex and composite optimization. Smoothness and convexity constants are updated via Lyapunov analysis. Inspired by stability analysis in ODE solvers, the method triggers line search only when accumulated perturbations become positive, thereby reducing gradient evaluations while preserving strong convergence guarantees. By integrating adaptive step size and momentum acceleration, A^2GD outperforms existing first-order methods across a range of problem settings.

1 Introduction

In this paper, we study the convex optimization problem

$$\min_{x \in \mathbb{R}^d} f(x), \tag{1}$$

where f is μ-strongly convex and L-smooth. When $\mu = 0$, we additionally assume f is coercive so that a global minimizer exists. We also consider the composite convex problem

$$\min_{x \in \mathbb{R}^d} f(x) := h(x) + g(x), \tag{2}$$

where h is L-smooth and g is convex, possibly non-smooth, with a proximal operator.

First-order methods, which rely only on gradient information, are widely used in machine learning for their efficiency and scalability (Bottou et al., 2018). Among them, *gradient descent* (GD), defined by

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \tag{3}$$

is fundamental. Despite its simplicity, GD faces two main challenges:

- **Step size selection.** Convergence depends heavily on the step size α_k. Small α_k slows progress; large α_k risks divergence. For L-smooth functions, $\alpha_k = 1/L$ is standard, but this global constant often mismatches local curvature.

- **Slow convergence.** Even with an optimal step size, GD is slow on ill-conditioned problems, i.e., when $L/\mu \gg 1$.

We briefly review strategies addressing these issues:

Backtracking line search Backtracking line search begins with a large step size α_k and reduces it until conditions such as the Armijo–Goldstein criterion (Armijo, 1966; Goldstein, 1962/63) or Wolfe condition (Wolfe, 1969) are satisfied. Extensions (Ito and Fukuda, 2021; Liu and Yang, 2017) adapt line search to composite settings. Guminov et al. (2019) update parameters in Nesterov's method with backtracking, while Lan et al. (2023) develop a parameter-free method that attains optimal complexity bounds for both convex and strongly convex problems, achieving the best known results for non-convex problems. An adaptive variant (Cavalcanti et al., 2025) reduces backtracking steps, improving efficiency. Despite robustness and simplicity, line search usually requires 3–4 extra function or gradient evaluations per iteration, increasing cost.

Adaptive step sizes   Adaptive schemes such as the Barzilai–Borwein (BB) method (Barzilai and Borwein, 1988) estimate step sizes from past iterates:

$$\alpha_k = \frac{\langle x_k - x_{k-1}, \nabla f(x_k) - \nabla f(x_{k-1}) \rangle}{\|\nabla f(x_k) - \nabla f(x_{k-1})\|^2}, \tag{4}$$

with low computational overhead. However, BB-type methods are heuristic, and may diverge even for simple convex problems (Burdakov et al., 2019); guarantees are largely limited to quadratic cases (Dai and Liao, 2002). Extensions (Zhou et al., 2006; Dai et al., 2015) improve robustness but still lack general theory.

Polyak's method (Polyak, 1969), foundational to adaptive approaches such as AdaGrad and AMSGrad (Vaswani et al., 2020), ensures convergence but requires the optimal value f^*, which is rarely available.

Line-search free methods.   Recent years have seen growing interest in line-search free adaptive methods. These algorithms keep the per-iteration cost of gradient descent while often achieving faster convergence and lower sensitivity to hyperparameters. Malitsky and Mishchenko (2020; 2024) introduced adaptive proximal gradient methods with theoretical guarantees, though lack of acceleration can hinder performance on ill-conditioned problems. Li and Lan (2024) and Cavalcanti et al. (2025) proposed adaptive NAG variants with backtracking-free updates, though the rule in Li and Lan (2024) is relatively complex.

Acceleration   Momentum-based methods accelerate convergence by leveraging past updates. The heavy-ball method (Polyak, 1964) and Nesterov's accelerated gradient (NAG) (Nesterov, 2003) achieve the optimal rate $1 - \sqrt{\mu/L}$ under strong convexity, assuming known L and μ. In the convex case ($\mu = 0$), NAG with step size $1/(k+3)$ (Nesterov, 1983) achieves the optimal $O(1/k^2)$ rate. Nesterov later extended this framework to composite problems by incorporating line search into accelerated proximal methods (Nesterov, 2012), also attaining $O(1/k^2)$.

In training deep neural networks, Adam (Adaptive Moment Estimation) (Kingma and Ba, 2015) is a widely used optimization algorithm that combines momentum and adaptive step sizes for fast and stable convergence. However, the original Adam algorithm lacks convergence guarantees, even for convex problems.

CONTRIBUTION

- We develop A^{2}GD, an adaptive accelerated gradient method with provable accelerated linear convergence for smooth and composite convex optimization.
- We adapt stability analysis from ODE solvers to reduce line search overhead, activating it only when accumulated perturbations are positive. The method is thus line-search reduced rather than line-search free (Fig. 2), and it outperforms existing line-search free methods in both theory and practice.
- We show numerically that A^{2}GD also consistently outperforms AGD variants (where a single A denotes either adaptivity or acceleration) and other methods combining adaptivity and acceleration.

Limitations and Extensions   While A^{2}GD achieves adaptive acceleration with strong theoretical guarantees, these results rely on convexity to ensure positivity of the Bregman divergence. Extending the framework to nonconvex settings remains an open problem. We provide preliminary evidence on a composite $\ell_{1\text{-}2}$ problem, where the nonconvex regularizer admits a closed-form proximal operator.

Although line search adds little overhead in practice—typically only a few extra evaluations—a sharper theoretical bound on the number of triggered line search steps is an important future direction.

Another extension is the stochastic setting. Developing a stochastic variant of A^{2}GD that preserves both adaptivity and acceleration under variance conditions would broaden applicability to large-scale machine learning, providing a step toward a theoretical justification of the empirical success of Adam.

Background on convex functions Let $f : \mathbb{R}^d \to \mathbb{R}$ be differentiable. The Bregman divergence between $x, y \in \mathbb{R}^d$ is defined as

$$D_f(y, x) := f(y) - f(x) - \langle \nabla f(x), y - x \rangle.$$

The function f is μ-strongly convex if for some $\mu > 0$,

$$D_f(y, x) \geq \frac{\mu}{2} \|y - x\|^2, \quad \forall x, y \in \mathbb{R}^d.$$

It is L-smooth, for some $L > 0$, if its gradient is L-Lipschitz:

$$\|\nabla f(y) - \nabla f(x)\| \leq L \|y - x\|, \quad \forall x, y \in \mathbb{R}^d.$$

The condition number is defined by $\kappa = L/\mu$. Let $\mathcal{S}_{L,\mu}$ denote the class of all differentiable functions that are both μ-strongly convex and L-smooth.

For $f \in \mathcal{S}_{L,\mu}$, the Bregman divergence satisfies (Nesterov, 2003):

$$\frac{1}{2L} \|\nabla f(x) - \nabla f(y)\|^2 \leq D_f(x, y) \leq \frac{1}{2\mu} \|\nabla f(x) - \nabla f(y)\|^2, \quad \forall x, y \in \mathbb{R}^d. \tag{5}$$

Taking $y = x^*$, where x^* minimizes f and $\nabla f(x^*) = 0$, yields:

$$\|\nabla f(x)\|^2 \geq 2\mu(f(x) - f(x^*)), \quad \forall x \in \mathbb{R}^d. \tag{6}$$

2 Adaptive Gradient Descent Method

We illustrate our main idea using gradient descent and later extend it to accelerated gradient descent. The steepest descent step chooses

$$\alpha_k^* = \arg\min_{\alpha > 0} f(x_k - \alpha \nabla f(x_k)), \tag{7}$$

which entails solving a one-dimensional convex problem. While conceptually simple, this can be costly unless a closed form is available.

For L-smooth functions, the fixed step size $\alpha_k = 1/L$ guarantees convergence, but is often overly conservative when local curvature is much smaller than L. To improve efficiency, we design step sizes that adapt to local geometry using $f(x_k)$ and $\nabla f(x_k)$.

We estimate the local Lipschitz constant L_k through Lyapunov analysis of the gradient descent method (3). Consider the Lyapunov function

$$E_k = f(x_k) - f(x^*), \tag{8}$$

where $x^* \in \arg\min f(x)$ and $f(x^*) = \min f$. Expanding f at x_{k+1} gives

$$\begin{aligned}
E_{k+1} - E_k &= f(x_{k+1}) - f(x_k) \\
&= \langle \nabla f(x_{k+1}), x_{k+1} - x_k \rangle - D_f(x_k, x_{k+1}) \\
&= -\alpha_k \langle \nabla f(x_{k+1}), \nabla f(x_k) \rangle - D_f(x_k, x_{k+1}) \\
&= -\frac{\alpha_k}{2} \|\nabla f(x_{k+1})\|^2 - \frac{\alpha_k}{2} \|\nabla f(x_k)\|^2 \\
&\quad + \frac{\alpha_k}{2} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1}).
\end{aligned}$$

Applying (6) to $\|\nabla f(x_{k+1})\|^2$ and rearranging yields

$$(1 + \mu \alpha_k) E_{k+1} \leq E_k - \frac{\alpha_k}{2} \|\nabla f(x_k)\|^2 + \frac{\alpha_k}{2} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1}). \tag{9}$$

If we use a line search to choose a small enough α_k such that

$$\alpha_k = \frac{1}{L_k} \leq \frac{2D_f(x_k, x_{k+1})}{\|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2}, \tag{10}$$

then dropping the negative terms in (9) gives the linear convergence

$$E_{k+1} \leq (1 + \mu/L_k)^{-1} E_k.$$

3

Since $\alpha_k = 1/L_k$, choosing a smaller α_k is equivalent to using a larger L_k. By (5), the criterion (10) holds once $L_k \geq L$. Standard backtracking starts with an initial estimate of L_k and increases it iteratively by a factor $r > 1$ until (10) is satisfied. This procedure requires at most $O(|\log L/\log r|)$ iterations. A more adaptive and efficient backtracking scheme was recently proposed in Cavalcanti et al. (2025), which we adapt for our purposes and briefly recall below.

Rewriting the stopping criterion (10) gives

$$v = \frac{2L_k D_f(x_k, x_{k+1})}{\|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2} \geq 1.$$

If $v < 1$, the criterion is not satisfied. Instead of increasing L_k by a fixed ratio, we update it as $L_k \leftarrow rL_k/v$, where $r > 1$ is a base ratio (e.g., $r = 3$). This adaptive scaling adjusts to the gap between the current condition and the stopping criterion, improving both efficiency and accuracy.

Line search adds computational overhead, since each update of L_k requires reevaluating $\nabla f(x_{k+1})$ and/or $f(x_{k+1})$, which is often the dominant cost in gradient-based methods. To reduce this cost and avoid line search, line-search free approaches have been proposed; see the literature review in the introduction.

However, enforcing line-search free updates is often too rigid and restrictive. In contrast, we reduce the number of line-search steps to a small fraction, achieving comparable cost to line-search free methods. Our approach is inspired by stability analysis in ODE solvers. The following result can be established by induction.

Lemma 2.1 (A variant of Lemma 5.7.1. in Gautschi (2011)). *Let $\{E_k\}$ be a positive sequence satisfying*

$$E_{k+1} \leq \delta_k(E_k + b_k), \quad k = 0, 1, \ldots,$$

where $\delta_k > 0$ and $b_k \in \mathbb{R}$. Then

$$E_{k+1} \leq \left(\prod_{i=0}^{k} \delta_i\right) E_0 + p_k, \quad k = 0, 1, \ldots,$$

where the accumlated perturbation

$$p_k = \sum_{i=0}^{k} \left(\prod_{j=i}^{k} \delta_j\right) b_i, \quad satisfying \quad p_k = \delta_k(p_{k-1} + b_k).$$

We use an adaptive gradient descent method (ad-GD) to illustrate our main idea and refer to Appendix A for the detailed algorithmic formulation. Applying Lemma 2.1 to GD under the Lyapunov analysis equation 9 gives

$$\delta_k = (1 + \mu/L_k)^{-1}, \quad b_k = b_k^{(1)} + b_k^{(2)},$$

$$b_k^{(1)} = \frac{1}{2L_k}\|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1}),$$

$$b_k^{(2)} = -\frac{1}{2L_k}\|\nabla f(x_k)\|^2.$$

The strong convexity constant μ may be unknown in practice. In implementation, we replace it by the upper bound $\mu_k = \min_{1 \leq i \leq k} L_k$.

In the line-search criterion (10), L_k is chosen so that $b_k^{(1)} < 0$, ensuring $b_k < 0$ at each step. Enforcing negativity step by step is sufficient but not necessary. Instead, we perform line search only when $p_k > 0$, and update L_k until the weighted sum $p_k \leq 0$. Once this condition holds, exponential decay follows:

$$E_{k+1} \leq \prod_{i=0}^{k} \left(1 + \frac{\mu}{L_i}\right)^{-1} E_0, \quad \text{if } p_k \leq 0.$$

4

In the early iterations, when iterates are far from optimal and $\|\nabla f(x_k)\|$ is large, the negative terms $b_k^{(2)}$ accumulate and can offset positive terms later. Line search is triggered only if the accumulated perturbation p_k becomes positive. In other words, classical line search enforces $b_k < 0$ in the ℓ_∞ sense for $\{b_k\}$, while our relaxation allows a weighted ℓ_2 sense. Figures 1 and 2 illustrate this idea.
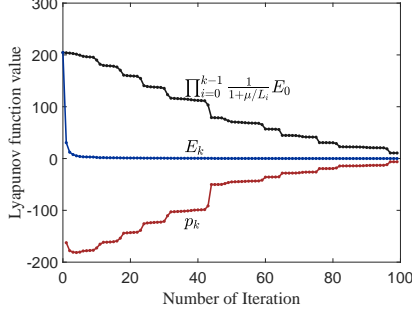

Figure 1: The accumulated perturbation p_k (red curve) remains negative and approaches zero. The actual values of E_k (blue curve) decay faster than the theoretical exponential rate $\left(\prod_{i=0}^{k} \delta_i\right) E_0$ (black curve) due to the negative perturbation p_k. Additionally, E_k decays more rapidly in the early iterations because of the large magnitude of the negative term $b_k^{(2)} = -\frac{1}{2L_k}\|\nabla f(x_k)\|^2$.
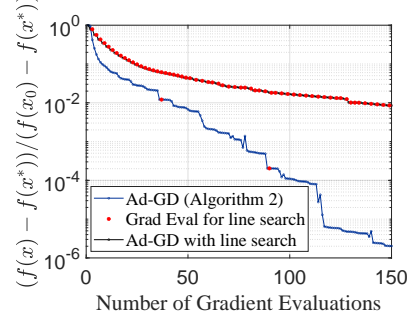
Figure 2: For a logistic regression problem with ℓ_2 regularizer (14), gradient descent with line search enforcing $b_k^{(1)} \leq 0$ (top curve) performs backtracking every 3–4 iterations on average. In contrast, ad-GD (Algorithm 2 in Appendix A) with line search targeting $p_k \leq 0$ (bottom curve) incurs only few additional gradient evaluations. Red dots indicate iterations where extra gradient evaluations occur during line search.

Theorem 2.2. *Assume $f \in \mathcal{S}_{L,\mu}$. Let $\{x_k\}$ be the sequence generated by gradient descent method (3) with line search ensuring $p_k \leq 0$. Then we have*

$$E_k \leq \prod_{i=0}^{k-1} \frac{1}{1 + \mu/L_i} E_0 \leq \left(\frac{1}{1 + \mu/(c_r L)}\right)^k E_0.$$

Proof. As $p_k \leq 0$ for all k, linear convergence follows from (9). By (5), the stopping criterion (10) is satisfied once $L_k \leq c_r L$ with at most $O(|\log L/\log r|)$ search steps, where $c_r \geq 1$ depends on the line-search scaling factor. Since $\mu_k \geq \mu$, the desired linear convergence rate follows. $\square$

Remark 2.1. To improve efficiency, we set the next step size as $\alpha_{k+1} = \frac{2D_f(x_k, x_{k+1})}{\|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2}$. The gradient $\nabla f(x_{k+1})$ can be reused in the following gradient descent step. However, computing $D_f(x_k, x_{k+1})$ requires function evaluations $f(x_k)$ and $f(x_{k+1})$, which may be costly. To avoid these evaluations, we approximate $2D_f(x_k, x_{k+1})$ by its symmetrized form:

$$2D_f(x_k, x_{k+1}) \approx D_f(x_k, x_{k+1}) + D_f(x_{k+1}, x_k) = \langle \nabla f(x_{k+1}) - \nabla f(x_k), \, x_{k+1} - x_k \rangle.$$

This reduces the ratio to the form used in the BB gradient method (4). In contrast to BB, convergence of ad-GD is guaranteed by enforcing $p_k \leq 0$.

Remark 2.2. There are several variants depending on how we define δ_k and split $b_k^{(1)}$ and $b_k^{(2)}$. For example, we can use $\delta_k = 1 - \mu/L_k$, $b_k^{(2)} = 0$, and the rest is $b_k^{(1)}$. Then $b_k^{(1)} \leq 0$ is equivalent to the criteria proposed by (Nesterov, 2012) (Appendix A).

3 Adaptive Accelerated Gradient Descent Method

In this section, we apply our adaptive strategy to accelerated gradient methods. We derive an identity for the difference of the Lyapunov function and adaptively adjust L_k and μ_k to ensure the accumulated perturbation satisfies $p_k \leq 0$. The perturbation term $b_k^{(1)}$ remains

unchanged, while $b_k^{(2)}$ includes additional terms and is used to adjust μ_k. The contraction rate δ_k improves from $(1 + \mu_k/L_k)^{-1}$ to $(1 + \sqrt{\mu_k/L_k})^{-1}$.

We will use the Hessian-based Nesterov accelerated gradient (HNAG) flow proposed in Chen and Luo (2019)

$$\begin{cases} x' = y - x - \beta \nabla f(x), \\ y' = x - y - \dfrac{1}{\mu} \nabla f(x), \end{cases} \tag{11}$$

where β is a parameter. An implicit and explicit (IMEX) discretization of (11) is

$$\begin{cases} x_{k+1} - x_k = \alpha_k \left(y_k - x_{k+1} \right) - \dfrac{1}{L_k} \nabla f(x_k), \\ y_{k+1} - y_k = -\dfrac{\alpha_k}{\mu_k} \nabla f(x_{k+1}) + \alpha_k \left(x_{k+1} - y_{k+1} \right), \end{cases} \tag{12}$$

where $\alpha_k > 0$ is the time step size and $L_k := (\alpha_k \beta_k)^{-1}$. Denote by $\boldsymbol{z} = (x, y)^{\mathsf{T}}$. Introduce the Lyapunov function

$$\mathcal{E}(\boldsymbol{z}; \mu) := f(x) - f(x^*) + \frac{\mu}{2} \|y - x^*\|^2.$$

The proof of the following identity can be found in Appendix B.

Lemma 3.1. *We have the identity*

$$(1 + \alpha_k)\mathcal{E}(\boldsymbol{z}_{k+1}; \mu_k) - \mathcal{E}(\boldsymbol{z}_k; \mu_k)$$

$$= \frac{1}{2} \left(\frac{\alpha_k^2}{\mu_k} - \frac{1}{L_k} \right) \|\nabla f(x_{k+1})\|_*^2 \quad \text{(I)}$$

$$+ \frac{1}{2L_k} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1}) \quad \text{(II)}$$

$$- \frac{1}{2L_k} \|\nabla f(x_k)\|_*^2 + \frac{\alpha_k \mu_k}{2} \left(\|x_{k+1} - x^*\|^2 - \frac{1}{\mu_k} D_f(x^*, x_{k+1}) - (1 + \alpha_k) \|x_{k+1} - y_{k+1}\|^2 \right) \quad \text{(III)}.$$

We can simply set $\alpha_k = \sqrt{\frac{\mu_k}{L_k}}$ so that (I) $= 0$. To control (II) and (III), define perturbations

$$b_k^{(1)} = \frac{1}{2L_k} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1}),$$

$$b_k^{(2)} = -\frac{1}{2L_k} \|\nabla f(x_k)\|^2 + \frac{\alpha_k \mu_k}{2} \left(R_k^2 - (1 + \alpha_k) \|x_{k+1} - y_{k+1}\|^2 \right), \tag{13}$$

$$p_k = \frac{1}{1 + \alpha_k} \left(p_{k-1} + b_k^{(1)} + b_k^{(2)} \right), \quad \forall k \geq 1 \text{ and } p_0 = 0.$$

The term $b_k^{(1)}$ measures deviation from the Lipschitz condition and is used to adjust L_k, while $b_k^{(2)}$ measures deviation from the strong convexity assumption and is used to adjust μ_k. To enforce the lower bound $\mu_k \geq \mu$ when $\mu > 0$, we introduce

$$R_k^2 := (1 - \mu/\mu_k) R^2,$$

using the inequality $D_f(x^*, x_{k+1}) \geq \frac{\mu}{2} \|x_{k+1} - x^*\|^2$ and an upper bound R such that $\|x_{k+1} - x^*\|^2 \leq R^2$. If $\mu_k < \mu$, then $b_k^{(2)} < 0$ and no further reduction of μ_k is allowed. The parameter μ can be a conservative estimate of the true convexity constant and serves as a lower bound for μ_k.

Line search is triggered only when $p_k > 0$. If $b_k^{(1)} > 0$, L_k is updated using adaptive backtracking Cavalcanti et al. (2025). If $b_k^{(2)} > 0$, the convexity is not strong enough to support a large step, so we reduce μ_k. In the limiting case $\mu_k = 0$, we have $b_k^{(2)} \leq 0$.

To update μ_k more precisely, we solve $b_k^{(2)} = 0$, treating L_k as known and using the fixed rule $\alpha_k = \sqrt{\mu_k/L_k}$ for the step size. The leading term in the second part of $b_k^{(2)}$ is $\alpha_k \mu_k R_k^2 = \mu_k^{3/2} R_k^2 / L_k^{1/2}$, and the equation essentially reduces to a non-trivial scaling

$$\frac{\mu_k^{3/2} R_k^2}{L_k^{1/2}} \approx \frac{\|\nabla f(x_k)\|^2}{L_k} \quad \Rightarrow \quad \mu_k \propto \frac{\|\nabla f(x_k)\|^{4/3}}{L_k^{1/3} R_k^{4/3}}.$$

To preserve decay of the Lyapunov function, we enforce

$$\mu_{k+1} \leq \mu_k \quad \Rightarrow \quad \mathcal{E}(z_{k+1}; \mu_{k+1}) \leq \mathcal{E}(z_{k+1}; \mu_k).$$

To establish convergence guarantee, the parameter μ_k cannot decay too fast. To control the decay rate of μ_k, we adopt the perturbation idea proposed in Chen et al. (2025). We introduce a parameter ε and impose the lower bound $\mu_k \geq \varepsilon$ in the update formulas (line 12 and line 16 in Algorithm 1).

To ensure monotonic descent, updates with $f(x_{k+1}) > f(x_k)$ are rejected by setting $x_{k+1} = x_k$. When $\|y_k - x^\star\| \gg \|x_k - x^\star\|$, the Lyapunov function $\mathcal{E}$ may decrease primarily due to $\|y_k - x^\star\|$, while $f(x_k)$ stagnates. To mitigate this, a restart is triggered by setting $y_k = x_k$ if $f(x_k)$ fails to decrease over five consecutive iterations. These monitoring steps are omitted from Algorithm 1, but are used in practice to enhance stability.

Algorithm 1: Adaptive Accelerated Gradient Method (A^{2}GD)

Input: $x_0, y_0 \in \mathbb{R}^n$, $L_0 > 0$, $\mu_0 > 0$, $R > 0$, $0 < \text{tol} \ll 1$, $\varepsilon > 0$, $m \geq 1$

1 **while** $\|\nabla f(x_k)\| > \text{tol}\|\nabla f(x_0)\|$ **do**

2    $\alpha_k \leftarrow \sqrt{\mu_k/L_k}$;

3    $x_{k+1} \leftarrow \frac{1}{\alpha_k+1} x_k + \frac{\alpha_k}{\alpha_k+1} y_k - \frac{1}{L_k(\alpha_k+1)} \nabla f(x_k)$;

4    $y_{k+1} \leftarrow \frac{\alpha_k}{\alpha_k+1} x_{k+1} + \frac{1}{\alpha_k+1} y_k - \frac{\alpha_k}{\mu_k(\alpha_k+1)} \nabla f(x_{k+1})$;

5    $b_k^{(1)} \leftarrow \frac{1}{2L_k} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1})$;

6    $b_k^{(2)} \leftarrow -\frac{1}{2L_k} \|\nabla f(x_k)\|_*^2 + \frac{\alpha_k \mu_k}{2} \left(R_k^2 - (1+\alpha_k)\|x_{k+1} - y_{k+1}\|^2 \right)$;

7    $p_k \leftarrow \frac{1}{1+\alpha_k}(p_{k-1} + b_k^{(1)} + b_k^{(2)})$;

8   **if** $p_k > 0$ **then**

9   **if** $b_k^{(1)} > 0$ **then**

10        $v \leftarrow \frac{2L_k D_f(x_k, x_{k+1})}{\|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2}$, $L_k \leftarrow 3L_k/v$;

11   **if** $b_k^{(2)} > 0$ **then**

12        $\mu_k \leftarrow \max\left\{\varepsilon, \min\left\{\mu_k, \frac{\|\nabla f(x_k)\|^{4/3}}{L_k^{1/3}\left(R_k^2 - (1+\alpha_k)\|x_{k+1}-y_{k+1}\|^2\right)^{2/3}}\right\}\right\}$;

13   Go to line 2;

14   **else**

15      $L_{k+1} \leftarrow \frac{\|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2}{2D_f(x_k, x_{k+1})}$;

16      $\mu_{k+1} \leftarrow \max\left\{\varepsilon, \min\left\{\mu_k, \frac{\|\nabla f(x_k)\|^{4/3}}{L_k^{1/3}\left(R_k^2 - (1+\alpha_k)\|x_{k+1}-y_{k+1}\|^2\right)^{2/3}}\right\}\right\}$;

17   **if** *decay condition* **then**

18      $\varepsilon \leftarrow \varepsilon/2$;

19      $m \leftarrow \lfloor \sqrt{2} \cdot m \rfloor + 1$;

20    $k \leftarrow k + 1$;

To reduce sensitivity to initialization, we include a warm-up phase using the adaptive proximal gradient descent (AdProxGD) method from Malitsky and Mishchenko (2024). Starting from x_0, we perform 10 iterations of AdProxGD and initialize A^{2}GD with $x_0 = y_0 := x_{10}$ and $\mu_0 := \min_{1 \leq k \leq 10}\{L_k\}$, and set $R = 100\|\nabla f(x_0)\|/\mu_0$. Although dynamical update of R may improve convergence, the method is typically robust with a fixed R.

The lower bound ε of μ_k is halved if certain decay conditions are met: if either $\mathcal{E}_k/\mathcal{E}_0 \leq (R^2 + 1)\varepsilon/2$, or the iteration steps for one fixed ε are larger than m. Since $\mathcal{E}_k$ is not directly computable, we can replace $\mathcal{E}_k/\mathcal{E}_0$ with $\|\nabla f(x_k)\|^2/\|\nabla f(x_0)\|^2$. If $\mu > 0$, then $\mathcal{E}_k/\mathcal{E}_0 \leq (R^2 + 1)\varepsilon/2$ will be satisfied in $\mathcal{O}(|\log \varepsilon|)$ steps. If $\mu = 0$, the maximal iteration steps for one fixed ε is bounded by m. We thus obtain the following convergence results.

Theorem 3.2. *Let (x_k, y_k) be the iterates generated by the above algorithm. Assume function f is μ-strongly convex with $\mu \geq 0$. Let k_s be the total number of steps after halving ε exactly s times, i.e. $\varepsilon = 2^{-s}\varepsilon_0$.*

1. *When $\mu = 0$, ther exists a constant $C > 0$ so that*

$$\frac{\mathcal{E}_{k_s}}{\mathcal{E}_0} \leq \frac{R^2 + 1}{\left(Ck_s + \varepsilon_0^{-1/2}\right)^2} = \mathcal{O}\left(\frac{1}{k_s^2}\right)$$

   So $\mathcal{O}(\sqrt{1/\mathrm{tol}})$ iteration steps can acheive $\mathcal{E}_{k_s}/\mathcal{E}_0 \leq \mathrm{tol}$.

2. *When $\mu > 0$, the iteration number to achieve $\mathcal{E}_{k_s}/\mathcal{E}_0 \leq (R^2 + 1)2^{-s}\varepsilon_0 \leq \mathrm{tol}$ is bounded by $\mathcal{O}(\sqrt{L/\mu} \ \ln \mathrm{tol})$.*

4 Numerical Experiments

We test A^{2}GD on smooth convex minimization tasks and compare it with several leading first-order methods, grouped into two categories:

- **Accelerated but non-adaptive methods:** Nesterov's accelerated gradient (NAG) with step size $1/(k+3)$ (Nesterov, 1983), accelerated over-relaxation heavy ball (AOR-HB) (Wei and Chen, 2025), and the triple momentum method (TM) (Van Scoy et al., 2018).
- **Adaptive methods:** adaptive proximal gradient descent (AdProxGD) (Malitsky and Mishchenko, 2024), the auto-conditioned fast gradient method (AC-FGM) (Li and Lan, 2024), and NAGfree (Cavalcanti et al., 2025).

For all examples, we set the tolerance to $\mathrm{tol} = 10^{-6}$ and use the stopping criterion $\|\nabla f(x_k)\| \leq \mathrm{tol} \cdot \|\nabla f(x_0)\|$. All experiments were run in MATLAB R2023a on a desktop with an Intel Core i5-7200U CPU (2.50 GHz) and 8 GB RAM.

Regularized Logistic Regression We report numerical simulations on a logistic regression problem with an ℓ_2 regularizer:

$$\min_{x \in \mathbb{R}^n} \left\{ \sum_{i=1}^m \log\left(1 + \exp(-b_i a_i^\top x)\right) + \frac{\lambda}{2}\|x\|^2 \right\}, \tag{14}$$

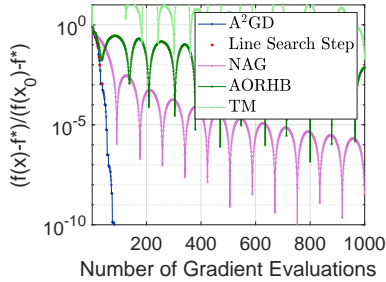where $(a_i, b_i) \in \mathbb{R}^n \times \{-1, 1\}$ for $i = 1, 2, \ldots, m$.

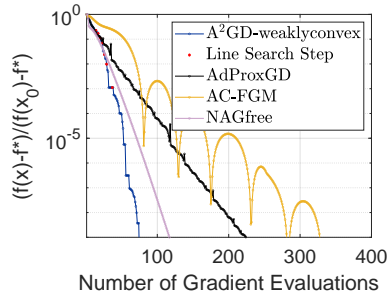Figure 3: A2GD compared to non-adaptive accelerated gradient methods.

Figure 4: A2GD compared to adaptive methods.

For this problem, $\mu = \lambda$ and $L = \lambda_{\max}\left(\sum_{i=1}^m a_i a_i^\top\right) + \lambda$. We use (a_i, b_i) from the Adult Census Income dataset. After removing entries with missing values, the dataset contains

30,162 samples. The Lipschitz constant is 6.30×10^4. With regularization parameter $\lambda = 0.1$, the condition number is $\kappa = 6.30 \times 10^5$.

We can extend the algorithm A^2GD and convergence analysis to the composite case and leave the details in Appendix C. We compare the performance of A^2GD with other first-order proximal methods: (1) the Adaptive Proximal Gradient descent (AdProxGD) method (Malitsky and Mishchenko, 2024); (2) Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) (Beck and Teboulle, 2009); (3) AOR-HB with perturbation Chen et al. (2025). We still use tol $= 10^{-6}$ but the stopping criterion is changed to $\|\nabla h(x_k) + q_k\| \le \text{tol}\|\nabla h(x_0)\|$ for all examples.

Maximum Likelihood Estimate of the Information Matrix We consider the maximum likelihood estimation problem from (Boyd and Vandenberghe, 2004, (7.5)):

$$\begin{aligned} &\underset{X \in \mathbb{R}^{n \times n}}{\text{minimize}} \quad f(X) := -\log \det X + \text{tr}(XY), \\ &\text{subject to} \quad \lambda_{\min} \le \lambda(X) \le \lambda_{\max}, \end{aligned} \tag{15}$$

where X is symmetric positive definite and $\lambda_{\min}, \lambda_{\max} > 0$ are given bounds. The condition number of f is $\kappa = \lambda_{\max}^2 / \lambda_{\min}^2$.

Problem (15) has a composite form, combining the smooth part $f(X)$ with a nonsmooth indicator $g(X)$ enforcing spectral constraints. The proximal operator of g requires eigendecomposition, projection of eigenvalues onto $[\lambda_{\min}, \lambda_{\max}]$, and matrix reconstruction—steps that dominate the cost. Thus, reducing gradient and proximal evaluations, especially during backtracking, is critical for efficiency.
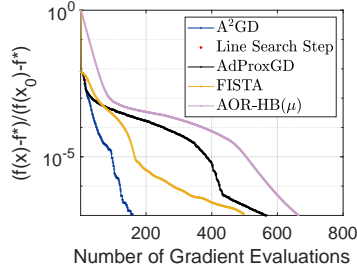

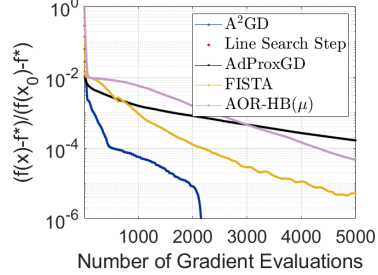Figure 5: Error curves under setting (1).

Figure 6: Error curves under setting (2).

Following Malitsky and Mishchenko (2024), we construct the data matrix Y as follows: sample a random vector $y \in \mathbb{R}^n$, and define $y_i = y + \delta_i$ for $i = 1, \ldots, M$, with $\delta_i \sim \mathcal{N}(0, I_n)$. Then set $Y = \frac{1}{M} \sum_{i=1}^{M} y_i y_i^\top$. We test our algorithm under two settings: (1) $n = 100$, $M = 50$, $\lambda_{\min} = 0.1$, $\lambda_{\max} = 10$; (2) $n = 50$, $M = 100$, $\lambda_{\min} = 0.1$, $\lambda_{\max} = 10^3$.
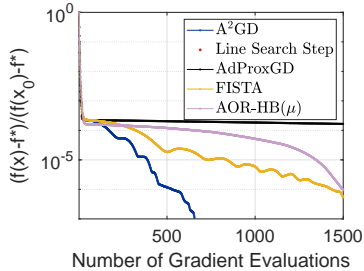

Figure 7: L_2-error curves for $\ell_{1\text{-}2}$ problem with $n = 500, p = 1000$.

$\ell_{1\text{-}2}$ **nonconvex minimization problem** We consider the $\ell_{1\text{-}2}$ minimization problem

$$\min_{x \in \mathbb{R}^n} \quad \frac{1}{2}\|Ax - b\|^2 + \lambda(\|x\|_1 - \|x\|_2), \tag{16}$$

introduced by Yin et al. (2015), promotes sparser solutions than standard convex penalties.

The matrix $A \in \mathbb{R}^{n \times p}$ is generated from a standard Gaussian distribution, and the ground truth $x^* \in \mathbb{R}^p$ has sparsity 50. The observation vector is constructed as $b = Ax^*$. We set the regularization parameter $\lambda = 1$ and consider two problem sizes: $n = 500$, $p = 1000$. The initial point is sampled as $x_0 = y_0 \sim 10\mathcal{N}(0, I_p)$.

Across all tests, our A^2GD method consistently outperforms baseline algorithms. A representative error trajectory is shown in Fig. 4, where A^2GD curve (in blue) decays much faster than others.

References

Larry Armijo. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific J. Math.*, 19(3):1–3, 1966. URL http://dml.mathdoc.fr/item/1102995080.

Jonathan Barzilai and Jonathan Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 01 1988. ISSN 0272-4979. doi: 10.1093/imanum/8.1.141. URL https://doi.org/10.1093/imanum/8.1.141.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

Léon Bottou, Frank Curtis, and Jorge Nocedal. Optimization Methods for Large-Scale Machine Learning. *SIAM Review*, 60(2):223–311, January 2018. ISSN 0036-1445, 1095-7200. doi: 10.1137/16M1080173.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Oleg Burdakov, Yuhong Dai, and Na Huang. Stabilized Barzilai-Borwein method. *Journal of Computational Mathematics*, 37(6):916–936, 2019. ISSN 1991-7139. doi: https://doi.org/10.4208/jcm.1911-m2019-0171. URL https://global-sci.com/article/84452/stabilized-barzilai-borwein-method.

Joao Cavalcanti, Laurent Lessard, and Ashia Wilson. Adaptive backtracking for faster optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=SrGP0RQbYH.

Long Chen and Hao Luo. First order optimization methods based on Hessian-driven Nesterov accelerated gradient flow, 2019. URL https://arxiv.org/abs/1912.09276.

Long Chen, Luo Hao, and Jingrong Wei. Accelerated gradient methods through variable and operator splitting, 2025. URL https://arxiv.org/abs/2505.04065.

Yu Hong Dai, Mehiddin Al-Baali, and Xiaoqi Yang. A positive Barzilai-Borwein-like stepsize and an extension for symmetric linear systems. In *Numerical Analysis and Optimization, NAO-III 2014*, volume 134, pages 59–75, United States, January 2015. Springer New York LLC. ISBN 9783319176888. doi: 10.1007/978-3-319-17689-5_3.

Yu-Hong Dai and Li-Zhi Liao. R-linear convergence of the Barzilai and Borwein gradient method. *IMA Journal of Numerical Analysis*, 22(1):1–10, 01 2002. ISSN 0272-4979. doi: 10.1093/imanum/22.1.1. URL https://doi.org/10.1093/imanum/22.1.1.

Walter Gautschi. *Numerical analysis*. Springer Science & Business Media, 2011.

A.A. Goldstein. Cauchy's method of minimization. *Numerische Mathematik*, 4:146–150, 1962/63. URL http://eudml.org/doc/131525.

Sergey Guminov, Yu Nesterov, Pavel Dvurechensky, and Alexander Gasnikov. Accelerated primal-dual gradient descent with linesearch for convex, nonconvex, and nonsmooth optimization problems. *Doklady Mathematics*, 99:125–128, 03 2019. doi: 10.1134/S1064562419020042.

Masaru Ito and Mituhiro Fukuda. Nearly Optimal First-Order Methods for Convex Optimization under Gradient Norm Measure: an Adaptive Regularization Approach. *Journal of Optimization Theory and Applications*, 188(3):770–804, March 2021. doi: 10.1007/s10957-020-01806-7. URL https://ideas.repec.org/a/spr/joptap/v188y2021i3d10.1007_s10957-020-01806-7.html.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015. URL https://arxiv.org/abs/1412.6980.

Guanghui Lan, Yuyuan Ouyang, and Zhe Zhang. Optimal and parameter-free gradient minimization methods for convex and nonconvex optimization. *arXiv preprint arXiv:2310.12139*, 2023.

Tianjiao Li and Guanghui Lan. A simple uniformly optimal method without line search for convex optimization, 2024. URL `https://arxiv.org/abs/2310.10082`.

Mingrui Liu and Tianbao Yang. Adaptive accelerated gradient converging method under Hölderian error bound condition. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/2612aa892d962d6f8056b195ca6e550d-Paper.pdf`.

Yura Malitsky and Konstantin Mishchenko. Adaptive gradient descent without descent. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6702–6712. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/malitsky20a.html`.

Yura Malitsky and Konstantin Mishchenko. Adaptive proximal gradient method for convex optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=qlH21Ig1IC`.

Yurii Nesterov. A method of solving a convex programming problem with convergence rate $O\left(\frac{1}{k^2}\right)$. *Doklady Akademii Nauk*, 269(3):543–547, 1983.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

Yurii Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140:125 – 161, 2012. URL `https://api.semanticscholar.org/CorpusID:18206201`.

B. Polyak. Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, 9(3):14–29, 1969. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(69)90061-5. URL `https://www.sciencedirect.com/science/article/pii/0041555369900615`.

Boris Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964.

Bryan Van Scoy, Randy A. Freeman, and Kevin M. Lynch. The Fastest Known Globally Convergent First-Order Method for Minimizing Strongly Convex Functions. *IEEE Control Systems Letters*, 2(1):49–54, January 2018. ISSN 2475-1456. doi: 10.1109/LCSYS.2017.2722406.

Sharan Vaswani, Frederik Kunstner, Issam Hadj Laradji, Si Yi Meng, Mark W. Schmidt, and Simon Lacoste-Julien. Adaptive gradient methods converge faster with overparameterization (and you can do a line-search). *ArXiv*, abs/2006.06835, 2020. URL `https://api.semanticscholar.org/CorpusID:219636069`.

Jingrong Wei and Long Chen. Accelerated over-relaxation heavy-ball method: Achieving global accelerated convergence with broad generalization. *ICLR*, 2025.

Philip Wolfe. Convergence conditions for ascent methods. *SIAM Rev.*, 11(2):226–235, April 1969. ISSN 0036-1445. doi: 10.1137/1011036. URL `https://doi.org/10.1137/1011036`.

Penghang Yin, Yifei Lou, Qi He, and Jack Xin. Minimization of ℓ_{1-2} for compressed sensing. *SIAM Journal on Scientific Computing*, 37(1):A536–A563, 2015. doi: 10.1137/140952363. URL `https://doi.org/10.1137/140952363`.

Bin Zhou, Li Gao, and Yu-Hong Dai. Gradient methods with adaptive step-sizes. *Computational Optimization and Applications*, 35:69–86, 09 2006. doi: 10.1007/s10589-006-6446-0.

APPENDIX A: ADAPTIVE GRADIENT DESCENT METHOD

We present an algorithm for adaptive gradient descent method (Ad-GD) which is a simplified version of A^2GD without momentum.

Algorithm 2: Adaptive Gradient Descent Method (Ad-GD)

Input: Initial point $x_0 \in \mathbb{R}^n$, initial step size $L_0 > 0$, initial strong convexity estimate
$\mu_0 > 0$
Output: Sequence $\{x_k\}$

1 **for** $k = 0, 1, 2, \ldots$ **do**

2     $x_{k+1} \leftarrow x_k - \frac{1}{L_k} \nabla f(x_k)$;

3     $b_k^{(1)} \leftarrow \frac{1}{2L_k} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1})$;

4     $b_k^{(2)} \leftarrow -\frac{1}{2L_k} \|\nabla f(x_k)\|^2$;

5     $p_k \leftarrow \left(1 + \frac{\mu_k}{L_k}\right)^{-1} \left(p_{k-1} + b_k^{(1)} + b_k^{(2)}\right)$;

6   **if** $p_k > 0$ **then**

7   Use adaptive backtracking to update L_k so that $b_k^{(1)} \leq 0$;

8     $L_k \leftarrow \frac{\|\nabla f(x_k) - \nabla f(x_{k+1})\|^2}{2D_f(x_k, x_{k+1})}$;

9     $\mu_k \leftarrow \min\{\mu_k, L_k\}$;

There are several variants of Ad-GD depending on how we define δ_k and split $b_k^{(1)}$ and $b_k^{(2)}$. For example, we can use $\delta_k = 1 - \mu/L_k$, $b_k^{(2)} = 0$, and

$$b_k^{(1)} := \frac{1}{2L_k} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1}) - \frac{1}{2L_k} \|\nabla f(x_{k+1})\|^2. \qquad (17)$$

The inequality $b_k^{(1)} \leq 0$ is equivalent to the criteria proposed by Nesterov in Nesterov (2012).

Proposition 4.1. *The inequality $b_k^{(1)} \leq 0$ is equivalent to*

$$m_{L_k}(x_{k+1}; x_k) \geq f(x_{k+1}), \qquad (18)$$

where $m_{L_k}(y; x) = f(x) + \langle \nabla f(x), y - x \rangle + \frac{L_k}{2} \|y - x\|^2$, $x_{k+1} = \arg\min_y m_{L_k}(y; x_k)$.

Proof. First, we have the identity

$$f(x_k) - f(x_{k+1}) - \frac{\alpha_k}{2} \|\nabla f(x_k)\|^2$$

$$= \frac{\alpha_k}{2} \|\nabla f(x_{k+1})\|^2 - \frac{\alpha_k}{2} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 + D_f(x_k, x_{k+1}).$$

Notice that $x_{k+1} = \arg\min_y m_{L_k}(y; x_k) \Leftrightarrow x_{k+1} = x_k - \frac{1}{L_k} \nabla f(x_k)$, so

$$m_{L_k}(x_{k+1}; x_k) \geq f(x_{k+1})$$

$$\Leftrightarrow f(x_k) + \left\langle \nabla f(x_k), -\frac{1}{L_k} \nabla f(x_k) \right\rangle + \frac{L_k}{2} \left\| \frac{1}{L} \nabla f(x_k) \right\|^2 \geq f(x_{k+1})$$

$$\Leftrightarrow f(x_k) - f(x_{k+1}) - \frac{1}{2L_k} \|\nabla f(x_k)\|^2 \geq 0$$

$$\Leftrightarrow \frac{1}{2L_k} \|\nabla f(x_{k+1})\|^2 + \frac{1}{2L_k} \|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2 - D_f(x_k, x_{k+1}) \geq 0.$$

Thus, equivalence is proved. $\qquad \square$

APPENDIX B: IDENTITIES OF ACCELERATED GRADIENT METHODS

We will use the Hessian-based Nesterov accelerated gradient (HNAG) flow proposed in Chen and Luo (2019)

$$\begin{cases} x' = y - x - \beta\nabla f(x), \\ y' = x - y - \dfrac{1}{\mu}\nabla f(x). \end{cases} \tag{19}$$

Denote by $\boldsymbol{z} = (x, y)^\intercal$ and $\mathcal{G}(\boldsymbol{z})$ the right hand side of (11), which now becomes $\boldsymbol{z}' = \mathcal{G}(\boldsymbol{z})$. In the notation $\nabla\mathcal{E}$, we consider μ as a fixed parameter and take derivative with respect to $\boldsymbol{z}$.

Lemma 4.2. *We have the identity*

$$-\nabla\mathcal{E}(\boldsymbol{z}) \cdot \mathcal{G}(\boldsymbol{z}) = \mathcal{E}(\boldsymbol{z}) + \beta\left\|\nabla f(x)\right\|_*^2 + \frac{\mu}{2}\left\|y - x\right\|^2 + D_f(x^*, x) - \frac{\mu}{2}\left\|x - x^\star\right\|^2. \tag{20}$$

Proof. A direct computation gives

$$\begin{aligned} -\nabla\mathcal{E}(\boldsymbol{z}) \cdot \mathcal{G}(\boldsymbol{z}) &= \begin{pmatrix} \nabla f(x) \\ \mu(y - x^\star) \end{pmatrix}\begin{pmatrix} (x - x^\star) - (y - x^\star) + \beta\nabla f(x) \\ (y - x^\star) - (x - x^\star) + \frac{1}{\mu}\nabla f(x) \end{pmatrix} \\ &= \langle\nabla f(x), x - x^\star\rangle + \beta\left\|\nabla f(x)\right\|_*^2 + \mu\left\|y - x^\star\right\|^2 - \mu(y - x^\star, x - x^\star) \\ &= \mathcal{E}(\boldsymbol{z}) + \beta\left\|\nabla f(x)\right\|_*^2 + D_f(x^*, x) + \frac{\mu}{2}\left\|y - x\right\|^2 - \frac{\mu}{2}\left\|x - x^\star\right\|^2. \end{aligned} \tag{21}$$

$\square$

Lemma 4.3. *We have the identity*

$$\begin{aligned} &(1 + \alpha_k)\mathcal{E}(\boldsymbol{z}_{k+1}; \mu_k) - \mathcal{E}(\boldsymbol{z}_k; \mu_k) \\ = (\text{I}) \quad &\frac{1}{2}\left(\frac{\alpha_k^2}{\mu_k} - \frac{1}{L_k}\right)\left\|\nabla f(x_{k+1})\right\|_*^2 \\ (\text{II}) + &\frac{1}{2L_k}\left\|\nabla f(x_{k+1}) - \nabla f(x_k)\right\|^2 - D_f(x_k, x_{k+1}) \\ (\text{III}) - &\frac{1}{2L_k}\left\|\nabla f(x_k)\right\|_*^2 + \frac{\alpha_k\mu_k}{2}\left(\left\|x_{k+1} - x^\star\right\|^2 - \frac{2}{\mu_k}D_f(x^\star, x_{k+1}) - (1 + \alpha_k)\left\|x_{k+1} - y_{k+1}\right\|^2\right). \end{aligned}$$

Proof. Treat μ_k as a fixed parameter. We expand the difference

$$\mathcal{E}(\boldsymbol{z}_{k+1}; \mu_k) - \mathcal{E}(\boldsymbol{z}_k; \mu_k) = \langle\nabla\mathcal{E}(\boldsymbol{z}_{k+1}; \mu_k), \boldsymbol{z}_{k+1} - \boldsymbol{z}_k\rangle - D_\mathcal{E}(\boldsymbol{z}_k, \boldsymbol{z}_{k+1}; \mu_k), \tag{22}$$

where the negative term $-D_\mathcal{E}(\boldsymbol{z}_k, \boldsymbol{z}_{k+1}; \mu_k)$ is expanded as $-D_f(x_k, x_{k+1}) - \frac{\mu_k}{2}\left\|y_k - y_{k+1}\right\|^2$. Using the identity (21) in the continuous level, we have

$$\begin{aligned} &\langle\nabla\mathcal{E}(\boldsymbol{z}_{k+1}; \mu_k), \alpha_k\mathcal{G}(\boldsymbol{z}_{k+1}, \mu_k)\rangle = -\alpha_k\mathcal{E}(\boldsymbol{z}_{k+1}, \mu_k) \\ &- \frac{1}{L_k}\left\|\nabla f(x_{k+1})\right\|_*^2 - \alpha_k D_f(x^*, x_{k+1}) + \frac{\alpha_k\mu_k}{2}\left(\left\|x_{k+1} - x^\star\right\|^2 - \left\|x_{k+1} - y_{k+1}\right\|^2\right). \end{aligned}$$

The difference between the scheme and the implicit Euler method is

$$\boldsymbol{z}_{k+1} - \boldsymbol{z}_k - \alpha_k\mathcal{G}(\boldsymbol{z}_{k+1}, \mu_k) = \alpha_k\begin{pmatrix} y_k - y_{k+1} + \beta_k(\nabla f(x_{k+1}) - \nabla f(x_k)) \\ 0 \end{pmatrix}.$$

which will bring more terms

$$\begin{aligned} &\langle\nabla_x\mathcal{E}(\boldsymbol{z}_{k+1}, \mu_k), \boldsymbol{z}_{k+1} - \boldsymbol{z}_k - \alpha_k\mathcal{G}(\boldsymbol{z}_{k+1}, \mu_k)\rangle \\ &= \frac{1}{L_k}(\nabla f(x_{k+1}), \nabla f(x_{k+1}) - \nabla f(x_k)) + \alpha_k\langle\nabla f(x_{k+1}), y_k - y_{k+1}\rangle. \end{aligned}$$

We then use the identity of squares for the cross term of gradients

$$\frac{1}{L_k}\langle \nabla f(x_{k+1}), \nabla f(x_{k+1}) - \nabla f(x_k)\rangle$$

$$= -\frac{1}{2L_k}\|\nabla f(x_k)\|_*^2 + \frac{1}{2L_k}\|\nabla f(x_{k+1})\|_*^2 + \frac{1}{2L_k}\|\nabla f(x_{k+1}) - \nabla f(x_k)\|_*^2.$$

As expected, this cross term brings more positive squares but also contribute a negative one.

On the second term, we write as

$$\alpha_k \langle \nabla f(x_{k+1}), y_k - y_{k+1}\rangle = \left\langle \frac{\alpha_k}{\sqrt{\mu_k}}\nabla f(x_{k+1}), \sqrt{\mu_k}(y_k - y_{k+1})\right\rangle$$

$$= \frac{\alpha_k^2}{2\mu_k}\|\nabla f(x_{k+1})\|_*^2 + \frac{\mu_k}{2}\|y_k - y_{k+1}\|^2 - \frac{1}{2}\left\|\frac{\alpha_k}{\sqrt{\mu_k}}\nabla f(x_{k+1}) - \sqrt{\mu_k}(y_k - y_{k+1})\right\|^2$$

$$= \frac{\alpha_k^2}{2\mu_k}\|\nabla f(x_{k+1})\|_*^2 + \frac{\mu_k}{2}\|y_k - y_{k+1}\|^2 - \frac{1}{2}\alpha_k^2\mu_k\|x_{k+1} - y_{k+1}\|^2.$$

Combining altogether, we get the desired identity. $\qquad\square$

PROOF OF THEOREM 3.2

First, we prove convergence of Algorithm 1 within a single inner iteration, i.e. ε is fixed, in the following lemma. It bears similarity to (Chen et al., 2025, Theorem 8.3), and is a direct result of Lemma 4.3.

Lemma 4.4. *Suppose f is convex and L-smooth. Let $z_k = (x_k, y_k)$ be the iterates generated by Algorithm 1 within an inner iteration where $\mu = \varepsilon$. Assume that there exists $R > 0$ such that*

$$\|x_k - x^*\| \le R, \quad \forall k \ge 0,$$

and that there exists $l \in (\varepsilon, L)$ such that $L_k \ge l$ for all $k \ge 0$. Then the Lyapunov function exhibits linear convergence up to a perturbation:

$$\mathcal{E}(z_k; \varepsilon) \le \left(\frac{1}{1 + \sqrt{\varepsilon/(rL)}}\right)^k \mathcal{E}(z_0; \varepsilon) + \frac{\varepsilon}{2}R^2,$$

where r is the backtracking ratio (in Algorithm 1, $r = 3$).

Proof. By Lemma 4.3, we have

$$\mathcal{E}(z_{k+1}; \mu_{k+1}) \le \frac{1}{1 + \alpha_k}\mathcal{E}(z_k; \mu_k) + \frac{1}{1 + \alpha_k}\left(b_k^{(1)} + b_k^{(2)}\right) + \frac{\alpha_k\mu_k}{2(1 + \alpha_k)}R^2.$$

Since $l \le L_k \le rL$, it follows that

$$\sqrt{\frac{\varepsilon}{rL}} \le \alpha_k \le \sqrt{\frac{\varepsilon}{l}}.$$

Therefore,

$$\mathcal{E}(z_{k+1}; \mu_{k+1}) \le \frac{1}{1 + \sqrt{\varepsilon/(rL)}}\mathcal{E}(z_k; \mu_k) + \frac{1}{1 + \alpha_k}\left(b_k^{(1)} + b_k^{(2)}\right) + \frac{\varepsilon\sqrt{\varepsilon/l}}{2(1 + \sqrt{\varepsilon/l})}R^2.$$

Iterating the inequality yields

$$\mathcal{E}(z_{k+1}) \le \left(\frac{1}{1 + \sqrt{\varepsilon/(rL)}}\right)^{k+1}\mathcal{E}(z_0) + p_{k+1} + \frac{\varepsilon\sqrt{\varepsilon/l}}{2(1 + \sqrt{\varepsilon/l})}\sum_{i=0}^{k}\left(\frac{1}{1 + \sqrt{\varepsilon/l}}\right)^i R^2,$$

where p_{k+1} is the accumulated perturbation. By Algorithm 1, we have $p_{k+1} \le 0$.

Finally, the geometric sum is bounded as

$$\sum_{i=0}^{k} \left(\frac{1}{1 + \sqrt{\varepsilon/l}} \right)^{i} \leq \frac{1 + \sqrt{\varepsilon/l}}{\sqrt{\varepsilon/l}}.$$

Substituting this estimate gives the claimed bound

$$\mathcal{E}(z_{k+1}; \varepsilon) \leq \left(\frac{1}{1 + \sqrt{\varepsilon/(rL)}} \right)^{k+1} \mathcal{E}(z_0; \varepsilon) + \frac{\varepsilon}{2} R^2.$$

$\square$

Proof of Theorem 3.2. We distinguish between the convex case ($\mu = 0$) and the strongly convex case ($\mu > 0$).

If $\mu = 0$, in this case, the proof of (Chen et al., 2025, Theorem 8.4) applies directly, once the single-inner-iteration convergence relation (Lemma 4.4) is established. Therefore, no further argument is needed.

If instead, $\mu > 0$, recall that in the algorithm the effective radius is updated as

$$R_k^2 = \left(1 - \frac{\mu}{\mu_k} \right) R^2.$$

Thus, whenever $\mu_k \geq \mu$, we obtain $R_k^2 \leq 0$, which implies that further reduction of μ_k is no longer admissible. In particular, μ_k will stop decreasing once the tolerance parameter ε satisfies $\varepsilon \leq \mu$.

Since ε is halved at each outer stage, the final value of μ_k is therefore bounded below by $\mu/2$. At the same time, the smoothness parameter satisfies $L_k \leq rL$ by construction. Hence, in the terminal stage we obtain an effective condition number bounded by

$$\kappa_{\text{eff}} = \frac{L_k}{\mu_k} \leq \frac{rL}{\mu/2} = \frac{2rL}{\mu}.$$

Applying the convergence estimate from Lemma 4.4 in this regime, the Lyapunov function contracts linearly:

$$\mathcal{E}_{k_s} \leq \left(\frac{1}{1 + \sqrt{\mu_k/L_k}} \right)^{k_s} \mathcal{E}_0 \leq \left(\frac{1}{1 + \sqrt{\mu/2rL}} \right)^{k_s} \mathcal{E}_0.$$

Therefore, to ensure that $\mathcal{E}_{k_s} \leq \text{tol} \cdot \mathcal{E}_0$, it suffices to take

$$k_s \geq \frac{\ln(1/\text{tol})}{\ln\left(1 + \sqrt{\mu/2rL} \right)} = \mathcal{O}\left(\sqrt{2rL/\mu} \ln(1/\text{tol}) \right).$$

This establishes the desired complexity bound in both cases. $\square$

APPENDIX C: COMPOSITE CONVEX OPTIMIZATION

We derive the continuous time analogy to Lemma 3.1. First, define the composite right hand side update

$$\mathcal{G}(z) = \left(y - x - \beta(\nabla h(x) + q), x - y - \frac{1}{\mu}(\nabla h(x) + q) \right)^{\text{T}},$$

where $q \in \partial g(x)$. Let $\mathcal{E}_h(z; \mu) = h(x) - h(x^*) + \frac{\mu}{2}\|y - x^*\|^2$, then $\mathcal{E}(z; \mu) = \mathcal{E}_h(z; \mu) + (g(x) - g(x^*))$ is splitted into a smooth part and a non-smooth part.

Lemma 4.5. *We have the following inequality*

$$-\left\langle \nabla \mathcal{E}_h(x) + \begin{pmatrix} q \\ 0 \end{pmatrix}, \mathcal{G}(z) \right\rangle \geq \mathcal{E}(z) + \beta\|\nabla h(x) + q\|_*^2 + \frac{\mu}{2}\|y - x\|^2 + D_h(x^*, x) - \frac{\mu}{2}\|x - x^*\|^2.$$

Proof. A direct computation gives

$$-\left\langle \nabla\mathcal{E}_h(x) + \begin{pmatrix} q \\ 0 \end{pmatrix}, \mathcal{G}(z) \right\rangle = \begin{pmatrix} \nabla h(x) + q \\ \mu(y - x^\star) \end{pmatrix} \begin{pmatrix} (x - x^\star) - (y - x^\star) + \beta(\nabla h(x) + q) \\ (y - x^\star) - (x - x^\star) + \frac{1}{\mu}(\nabla h(x) + q) \end{pmatrix}$$

$$= \langle \nabla h(x) + q, x - x^\star \rangle + \beta \|\nabla h(x) + q\|_*^2 + \mu \|y - x^\star\|^2 - \mu(y - x^\star, x - x^\star)$$

$$\geq \mathcal{E}(z) + \beta \|\nabla h(x) + q\|_*^2 + D_h(x^*, x) + \frac{\mu}{2} \|y - x\|^2 - \frac{\mu}{2} \|x - x^\star\|^2,$$

$$(23)$$

the last inequality following from $q \in \partial g(x)$. $\qquad\square$

Lemma 4.6. *We have the following inequality*

$$(1 + \alpha_k)\mathcal{E}(z_{k+1}; \mu_k) - \mathcal{E}(z_k; \mu_k)$$

$$\leq (\mathrm{I}) \quad \frac{1}{2}\left(\frac{\alpha_k^2}{\mu_k} - \frac{1}{L_k}\right) \|\nabla h(x_{k+1}) + q_{k+1}\|_*^2$$

$$(\mathrm{II}) + \frac{1}{2L_k}\|\nabla h(x_{k+1}) - \nabla h(x_k)\|^2 - D_h(x_k, x_{k+1})$$

$$(\mathrm{III}) - \frac{1}{2L_k}\|\nabla h(x_k) + q_{k+1}\|_*^2 + \frac{\alpha_k \mu_k}{2}\left(\|x_{k+1} - x^\star\|^2 - \frac{2}{\mu_k}D_h(x^\star, x_{k+1}) - (1 + \alpha_k)\|x_{k+1} - y_{k+1}\|^2\right).$$

Proof. The proof is similar to the smooth convex case. Expand the difference of $\mathcal{E}$ at z_{k+1},

$$\mathcal{E}(z_{k+1}; \mu_k) - \mathcal{E}(z_k; \mu_k) \leq \left\langle \nabla\mathcal{E}_h(z_{k+1}; \mu_k) + \begin{pmatrix} q_{k+1} \\ 0 \end{pmatrix}, z_{k+1} - z_k \right\rangle - D_{\mathcal{E}_h}(z_k, z_{k+1}; \mu_k), \quad (24)$$

where the negative term $-D_{\mathcal{E}_h}(z_k, z_{k+1}; \mu_k)$ is expanded as $-D_h(x_k, x_{k+1}) - \frac{\mu_k}{2}\|y_k - y_{k+1}\|^2$. The inequality is due to the definition of the subgradient.

From Lemma 4.5, we have

$$\left\langle \nabla\mathcal{E}_h(z_{k+1}; \mu_k) + \begin{pmatrix} q_{k+1} \\ 0 \end{pmatrix}, \alpha_k\mathcal{G}(z_{k+1}, \mu_k) \right\rangle \leq -\alpha_k\mathcal{E}(z_{k+1}, \mu_k)$$

$$- \frac{1}{L_k}\|\nabla h(x_{k+1}) + q_{k+1}\|_*^2 - \alpha_k D_h(x^*, x_{k+1}) + \frac{\alpha_k \mu_k}{2}\left(\|x_{k+1} - x^\star\|^2 - \|x_{k+1} - y_{k+1}\|^2\right).$$

The difference between the scheme and the implicit Euler method is

$$z_{k+1} - z_k - \alpha_k\mathcal{G}(z_{k+1}, \mu_k) = \alpha_k\begin{pmatrix} y_k - y_{k+1} + \beta_k(\nabla h(x_{k+1}) - \nabla h(x_k)) \\ 0 \end{pmatrix}.$$

which will bring more terms

$$\langle \nabla_x\mathcal{E}_h(z_{k+1}, \mu_k) + q_{k+1}, z_{k+1} - z_k - \alpha_k\mathcal{G}(z_{k+1}, \mu_k) \rangle$$

$$= \frac{1}{L_k}\left(\nabla h(x_{k+1}) + q_{k+1}, \nabla h(x_{k+1}) - \nabla h(x_k)\right) + \alpha_k\langle \nabla h(x_{k+1}) + q_{k+1}, y_k - y_{k+1} \rangle.$$

For the first term, we use the identity of squares

$$\frac{1}{L_k}\left(\nabla h(x_{k+1}) + q_{k+1}, \nabla h(x_{k+1}) - \nabla h(x_k)\right)$$

$$= -\frac{1}{2L_k}\|\nabla h(x_k) + q_{k+1}\|_*^2 + \frac{1}{2L_k}\|\nabla h(x_{k+1}) + q_{k+1}\|_*^2 + \frac{1}{2L_k}\|\nabla h(x_{k+1}) - \nabla h(x_k)\|_*^2.$$

As expected, this cross term brings more positive squares but also contribute a negative one.

For the second term, we rewrite as

$$\alpha_k\langle \nabla h(x_{k+1}) + q_{k+1}, y_k - y_{k+1} \rangle = \left\langle \frac{\alpha_k}{\sqrt{\mu_k}}\nabla h(x_{k+1}) + q_{k+1}, \sqrt{\mu_k}(y_k - y_{k+1}) \right\rangle$$

$$= \frac{\alpha_k^2}{2\mu_k}\|\nabla h(x_{k+1}) + q_{k+1}\|_*^2 + \frac{\mu_k}{2}\|y_k - y_{k+1}\|^2 - \frac{1}{2}\left\|\frac{\alpha_k}{\sqrt{\mu_k}}(\nabla h(x_{k+1}) + q_{k+1}) - \sqrt{\mu_k}(y_k - y_{k+1})\right\|^2$$

$$= \frac{\alpha_k^2}{2\mu_k}\|\nabla h(x_{k+1}) + q_{k+1}\|_*^2 + \frac{\mu_k}{2}\|y_k - y_{k+1}\|^2 - \frac{1}{2}\alpha_k^2\mu_k\|x_{k+1} - y_{k+1}\|^2.$$

Combining altogether, we get the desired inequality. $\qquad\square$

Algorithm 3: A^{2}GD method for composite optimization

Input: $x_0, y_0 \in \mathbb{R}^n$, $L_0, \mu_0, R > 0$, tol > 0, $\varepsilon > 0$, $m \geq 1$

1 while $k = 0$ *or* $\|\nabla f(x_k) + q_k\| >$ tol$\|\nabla f(x_0)\|$ **do**

2 $\alpha_k \leftarrow \sqrt{\mu_k / L_k}$;

3 $w_{k+1} \leftarrow \frac{1}{\alpha_k+1} x_k + \frac{\alpha_k}{\alpha_k+1} y_k - \frac{1}{L_k(\alpha_k+1)} \nabla h(x_k)$;

4 $x_{k+1} \leftarrow \text{prox}_{\frac{1}{L_k(\alpha_k+1)} g}(w_{k+1})$;

5 $q_{k+1} \leftarrow L_k(\alpha_k + 1)(w_{k+1} - x_{k+1})$;

6 $y_{k+1} \leftarrow \frac{\alpha_k}{\alpha_k+1} x_{k+1} + \frac{1}{\alpha_k+1} y_k - \frac{\alpha_k}{\mu_k(\alpha_k+1)}(\nabla h(x_{k+1}) + q_{k+1})$;

7 $b_k^{(1)} \leftarrow \frac{1}{2L_k}\|\nabla h(x_{k+1}) - \nabla h(x_k)\|^2 - D_h(x_k, x_{k+1})$;

8 $b_k^{(2)} \leftarrow -\frac{1}{2L_k}\|\nabla h(x_k) + q_{k+1}\|_*^2 + \frac{\alpha_k \mu_k}{2}\left(R^2 - (1+\alpha_k)\|x_{k+1} - y_{k+1}\|^2\right)$;

9 $p_k \leftarrow \frac{1}{1+\alpha_k}(p_{k-1} + b_k^{(1)} + b_k^{(2)})$;

10   **if** $p_k > 0$ **then**

11   **if** $b_k^{(1)} > 0$ **then**

12 $v \leftarrow \frac{2L_k D_f(x_k, x_{k+1})}{\|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2}$, $L_k \leftarrow 3L_k/v$;

13   **if** $b_k^{(2)} > 0$ **then**

14 $\mu_k \leftarrow \max\left\{\varepsilon, \min\left\{\mu_k, \frac{\|\nabla h(x_k)+q_{k+1}\|^{4/3}}{L_k^{1/3}(R^2-(1+\alpha_k)\|x_{k+1}-y_{k+1}\|^2)^{2/3}}\right\}\right\}$;

15   Go to line 2;

16   **else**

17 $L_k \leftarrow \frac{\|\nabla h(x_{k+1}) - \nabla h(x_k)\|^2}{2D_h(x_k, x_{k+1})}$;

18 $\mu_{k+1} \leftarrow \max\left\{\varepsilon, \min\left\{\mu_k, \frac{\|\nabla h(x_k)+q_{k+1}\|^{4/3}}{L_k^{1/3}(R^2-(1+\alpha_k)\|x_{k+1}-y_{k+1}\|^2)^{2/3}}\right\}\right\}$;

19   **if** *decay condition* **then**

20 $\varepsilon \leftarrow \varepsilon/2$;

21 $m \leftarrow \lfloor \sqrt{2} \cdot m \rfloor + 1$;

22 $k \leftarrow k + 1$;

Theorem 4.7. *Let (x_k, y_k) be the iterates generated by Algorithm 3. Assume function f is μ-convex with $\mu \geq 0$. Assume there exists $R > 0$ such that*

$$\|x_k - x^*\| \leq R, \qquad \forall \ k \geq 0.$$

Let k_s be the total number of steps after halving ε exactly s times, i.e. $\varepsilon = 2^{-s}\varepsilon_0$.

1. *When $\mu = 0$, ther exists a constant $C > 0$ so that*

$$\frac{\mathcal{E}_{k_s}}{\mathcal{E}_0} \leq \frac{R^2 + 1}{\left(Ck_s + \varepsilon_0^{-1/2}\right)^2} = \mathcal{O}\left(\frac{1}{k_s^2}\right)$$

2. *When $\mu > 0$, the iteration number to achieve $\mathcal{E}_{k_s}/\mathcal{E}_0 \leq (R^2 + 1)2^{-s}\varepsilon_0 \leq$ tol is bounded by $\mathcal{O}(\sqrt{L/\mu} \ \ln \text{tol})$,*

where $\mathcal{E}_k = \mathcal{E}(\boldsymbol{z}_k; \mu_k) = f(x_k) - f(x^\star) + \frac{\mu_k}{2}\|y_k - x^\star\|^2$.

LLM USAGE

In preparing this manuscript, large language models (LLMs) were employed exclusively to assist with language-related tasks, such as improving readability, grammar, and style. The models were not used for research ideation, development of methods, data analysis, or

interpretation of results. All scientific content, including problem formulation, theoretical analysis, and experimental validation, was conceived, executed, and verified entirely by the authors. The authors bear full responsibility for the accuracy and integrity of the manuscript.

ETHICS STATEMENT

This work is purely theoretical and algorithmic, focusing on convex optimization methods. It does not involve human subjects, sensitive data, or applications that raise ethical concerns related to privacy, security, fairness, or potential harm. All experiments are based on publicly available datasets or synthetic data generated by standard procedures. The authors believe that this work fully adheres to the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

We have taken several measures to ensure the reproducibility of our results. All theoretical assumptions are explicitly stated, and complete proofs are provided in the appendix. For the experimental evaluation, we describe the setup, parameter choices, and baselines in detail in the main text. The source code for our algorithms and experiments are available as supplementary materials. Together, these resources should allow others to reproduce and verify our theoretical and empirical findings.