Adaptive Proximal Gradient Method for Convex Optimization

Yura Malitsky
Faculty of Mathematics
University of Vienna, Austria
yurii.malitskyi@univie.ac.at

Konstantin Mishchenko
Samsung AI Center, UK
konsta.mish@gmail.com

Abstract

In this paper, we explore two fundamental first-order algorithms in convex optimization, namely, gradient descent (GD) and proximal gradient method (ProxGD). Our focus is on making these algorithms entirely adaptive by leveraging local curvature information of smooth functions. We propose adaptive versions of GD and ProxGD that are based on observed gradient differences and, thus, have no added computational costs. Moreover, we prove convergence of our methods assuming only *local* Lipschitzness of the gradient. In addition, the proposed versions allow for even larger stepsizes than those initially suggested in [MM20].

1 Intro

In this paper, we address a convex minimization problem

$$\min_{x\in\mathbb{R}^d} F(x).$$

We are interested in the cases when either F is differentiable and then we will use notation $F = f$, or it has a composite additive structure as $F = f + g$. Here, f represents a convex and differentiable function, while g is convex, lower semi-continuous (lsc), and prox-friendly. Throughout the paper, we will interchangeably refer to the smoothness of f and the Lipschitzness of ∇f, occasionally with the adjective "locally," indicating that it is restricted to a bounded set. We will refer to this property as smoothness, without mentioning the Lipschitzness of f, so we hope there will be no confusion in this regard.

For simplicity, in most of the introduction, we consider only the simpler problem $\min_x f(x)$. We study one of the most classical optimization algorithms — *gradient descent* —

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

Its simplicity and the sole prerequisite of knowing the gradient of f make it appealing for diverse applications. This method is central in modern continuous optimization, forming the bedrock for numerous extensions.

Given the initial point x^0, the only thing we need to implement (1) is to choose a stepsize α_k (also known as a learning rate in machine learning literature). This seemingly tiny detail is crucial for the method convergence and performance. When a user invokes GD as a solver, the standard approach would be to pick an arbitrary value for α_k, run the algorithm, and observe its behavior. If it diverges at some point, the user would try a smaller stepsize and repeat the same procedure. If, on the other hand, the method takes too much time to converge, the user might try to increase the stepsize. In practice, this approach is not very efficient, as we have no theoretical guarantees for a randomly guessed stepsize, and the divergence may occur after a long time. Both underestimating and overestimating the stepsize can, thus, lead to a large overhead.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-3193

Below we briefly list possible approaches to choosing or estimating the stepsize and we provide a more detailed literature overview in Section 5.

Fixed stepsize. When f is L-smooth, GD can utilize a fixed stepsize $\alpha_k = \alpha < \frac{2}{L}$ and values larger than $\frac{2}{L}$ will provably lead to divergence. Consequently, in such scenarios, the rate of convergence is given by $f(x^k) - f_* = \mathcal{O}\left(\frac{1}{\alpha k}\right)$, clearly indicating a direct dependence on the stepsize. Nevertheless, several drawbacks emerge from this approach:

(a) L is not available in many practical scenarios;

(b) if the curvature of f changes a lot, GD with the global value of L may be too conservative;

(c) f may be not globally L-smooth.

For illustration, consider the following functions. Firstly, when dealing with $f(x) = \frac{1}{2}\|Ax - b\|^2$, where $A \in \mathbb{R}^{n\times d}$ and $b \in \mathbb{R}^n$, estimating L involves evaluating the largest eigenvalue of $A^\top A$. Second, the logistic loss $f(x) = \log(1 + \exp(-ba^\top x))$, with $a \in \mathbb{R}^d, b \in \{-1, 1\}$, is almost flat for large x, yet for values of x closer to 0, it has $L = \frac{1}{4}\|a\|^2$. Thus, if the solution is far from 0, gradient descent with a constant stepsize would be too conservative. Finally, consider $f(x) = x^4$. While this simple objective is not globally L-smooth for any value of L, on any bounded set it *is* smooth, and we would hope we can still minimize objectives like that.

Linesearch. Also known as backtracking in the literature. In the k-th iteration we compute x^{k+1} with a certain stepsize α_k and check a specific condition. If the condition holds, we accept x^{k+1} and proceed to the next iteration; otherwise we halve α_k and recompute x^{k+1} using this reduced stepsize. This approach, while the most robust and theoretically sound, incurs substantially higher computational costs compared to regular GD due to the linesearch procedure.

Adagrad-type algorithms. These are the methods of the type[1]

$$\begin{aligned} v_k &= v_{k-1} + \|\nabla f(x^k)\|^2 \\ x^{k+1} &= x^k - \frac{d_k}{\sqrt{v_k}}\nabla f(x^k), \end{aligned} \tag{2}$$

where $v_{-1} \geq 0$ is some constants, and d_k is an estimate of $\|x^0 - x^*\|$ for some solution x^*. While such methods indeed have certain nice properties, d_k is usually either constant or quickly converges to a constant value, so a quick glance at (2) will reveal that its stepsizes are decreasing. Therefore, despite the name, we cannot expect true adaptivity of this method to the local curvature of f.

Heuristics. Numerous heuristics exist for selecting α_k based on local properties of f and ∇f, with the Barzilai-Borwein method [BB88] being among the most widely popular. However, it is crucial to note that we are not particularly interested in such approaches, as they lack consistency and may even lead to divergence, even for simple convex problems.

We have already mentioned *adaptivity* a few times, without properly introducing it. Now let us try to properly understand its meaning in the context of gradient descent. Besides the initial point x^0, GD has only one degree of freedom — its stepsize. From the analysis we know that it has to be approximately an inverse of the local smoothness. We call a method *adaptive*, if it automatically adapts a stepsize to this local smoothness without additional expensive computation and the method does not deteriorate the rate of the original method in the worst case. In our case, the original method is GD with a fixed stepsize.

By this definition, GD with linesearch is not adaptive, because it finds the right stepsize with some extra evaluations of f or ∇f. GD with diminishing steps (as in subgradient or Adagrad methods) is also not adaptive, because decreasing steps cannot in general represent well the function's curvature; also the rate of the subgradient method is definitely worse. It goes without saying, that for a *good*

[1]We provide only the simplest instance of such algorithms.

method its rate must experience improvement when we confine the class of smooth convex functions to the strongly convex ones.

Contribution

In a previous work [MM20], which serves as the cornerstone for the current paper, the authors proposed an adaptive gradient method named *"Adaptive Gradient Descent without Descent"* (AdGD). In the current paper, we

- deepen our understanding of AdGD and identify its limitations;
- refine its theory to accommodate even larger steps;
- extend the revised algorithm from unconstrained to the proximal case.

The analysis in the last two cases is not a trivial extension, and we were rather pleasantly surprised that this was possible at all. After all, the theory of GD is well-established and we thought it to be too well-explored for us to discover something new.

Continuous point of view. It is instructive for some time to switch from the discrete setting to the continuous and to compare gradient descent (GD) with its parent — gradient flow (GF)

$$x'(t) = -\nabla f(x(t)), \qquad x(0) = x_0, \tag{3}$$

where t is the time variable and $x'(t)$ denotes the derivative of $x(t)$ with respect to t. To guarantee the existence and uniqueness of a trajectory $x(t)$ of GF, it is sufficient to assume that ∇f is *locally* Lipschitz-continuous. Then one can prove convergence of $x(t)$ to a minimizer of f in just a few lines. For GD, on the other hand, the central assumption is *global* Lipschitzness of ∇f. Our analysis of gradient descent makes it level: local Lipschitzness suffices for both. Or to put it differently, we provide an adaptive discretization of GF that converges under the same assumptions as the original continuous problem (3).

Proximal case. We emphasize that there is already an excellent extension by Latafat et al. [Lat+23] of the work [MM20] to the additive composite case. Our proposed result, however, is based on an improved unconstrained analysis and uses a different (and simpler) proof. We believe that both these facts will be of interest. We don't have a good understanding why, but for us finding the proof for the proximal case was quite challenging. It does not follow the standard lines of arguments and uses a novel Lyapunov energy in the analysis.

Nonconvex problems. We believe that our algorithm will be no less important in the nonconvex case, where gradients are rarely globally Lipschitz continuous and where the curvature may change more drastically. It is true that our analysis applies only to the convex case, but, as far as we know, limited theory has never yet prevented practitioners from using methods in a broader setting. And based on our (speculative) experience, we found it challenging to identify nonconvex functions where the method did not converge to a local solution.

Outline. In Section 2, we begin by revisiting AdGD from [MM20], examining its limitations, and demonstrating a simple way to enhance it. This section maintains an informal tone, making it easily accessible for quick reading and classroom presentation. In Section 3, we further improve the method and provide all formal proofs, most of which we move to the Appendix. Section 4 extends the improved method to the proximal case. In Section 5 we put our finding in the perspective and compare it to some existing works. Lastly, in Section 6 (see also Appendix D), we conduct experiments to evaluate the proposed method against different linesearch variants.

1.1 Preliminaries

We say that a mapping is *locally Lipschitz* if it is Lipschitz over any compact set of its domain. A function $f\colon \mathbb{R}^d \to \mathbb{R}$ is *(locally) smooth* if its gradient ∇f is (locally) Lipschitz.

A convex L-smooth function f is characterized by the following inequality

$$f(y) - f(x) - \langle \nabla f(x), y - x\rangle \geqslant \frac{1}{2L}\|\nabla f(y) - \nabla f(x)\|^2 \quad \forall x, y. \tag{4}$$

https://doi.org/10.52202/079017-3193

This is equivalently of saying that ∇f is a $\frac{1}{L}$-*cocoercive* operator, that is

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \geqslant \frac{1}{L} \|\nabla f(y) - \nabla f(x)\|^2 \quad \forall x, y. \tag{5}$$

For a convex differentiable f that is not L-smooth one can only say that ∇f is *monotone*, that is

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \geqslant 0 \quad \forall x, y. \tag{6}$$

We use notation $[t]_+ = \max\{t, 0\}$ and for any $a > 0$ we suppose that $\frac{a}{0} = +\infty$. With a slight abuse of notation, we write $[n]$ to denote the set $\{1, \ldots, n\}$. A solution and the value of the optimization problem $\min f(x)$ are denoted by x^* and f_*, respectively.

2 Adaptive gradient descent: better analysis

Let us start with the simpler problem of $\min_x f(x)$ with a convex, locally smooth $f\colon \mathbb{R}^d \to \mathbb{R}$. To solve it, in [MM20], the authors proposed a method called *adaptive gradient descent without descent* (AdGD), whose update is given below:

$$\begin{aligned} \alpha_k &= \min\left\{\sqrt{1+\theta_{k-1}}\alpha_{k-1}, \frac{\|x^k - x^{k-1}\|}{2\|\nabla f(x^k) - \nabla f(x^{k-1})\|}\right\}, \qquad \text{where } \theta_k = \frac{\alpha_k}{\alpha_{k-1}} \\ x^{k+1} &= x^k - \alpha_k \nabla f(x^k). \end{aligned} \tag{7}$$

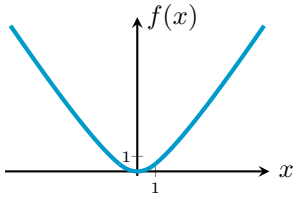
Similarly to the standard GD, this method leads to $\mathcal{O}(1/k)$ convergence rate. However, unlike the former, it doesn't require any knowledge about Lipschitz constant of ∇f and doesn't even require a global Lipschitz continuity of ∇f.

The update for α_k has two ingredients. The first bound $\alpha_k \leqslant \sqrt{1+\theta_{k-1}}\alpha_{k-1}$ sets how fast steps may increase from iteration to iteration. The second $\alpha_k \leqslant \frac{\|x^k - x^{k-1}\|}{2\|\nabla f(x^k) - \nabla f(x^{k-1})\|}$ corresponds to the estimate of local Lipschitzness of ∇f.

It is important to understand how essential these bounds are. Do we really need to control the growth rate of α_k or is it an artifact of our analysis? For the second bound, it is not clear whether 2 in the denominator is necessary. For example, given L-smooth f, our scheme (7) does not encompass a standard GD with $\alpha_k = \frac{1}{L}$ for all k.

First bound. Answering the first question is relatively easy. Consider the following function

$$f(x) = \begin{cases} \frac{1}{2}x^2, & x \in [-1, 1] \\ a(|x| - \log(1 + |x|)) + b, & x \notin [-1, 1] \end{cases} \tag{8}$$



where parameters $a, b > 0$ are chosen to ensure that $f(\pm 1)$ and $f'(\pm 1)$ are well-defined, namely $a = 2$ and $b = 2\log 2 - \frac{3}{2}$, see Lemma 3 in Appendix A.

From an optimization point of view, f is a nice function. In particular, it is convex (even locally strongly convex) and its gradient is 1-Lipschitz, see Lemma 3. This means that both GD and AdGD linearly converge on it. However, if we remove the first condition for α_k in AdGD, this new modified algorithm will fail to converge. We can prove an even stronger statement. Specifically, let $c \geqslant 1$, $\alpha_0 = 1$ and consider the following method

$$\begin{aligned} \alpha_k &= \frac{\|x^k - x^{k-1}\|}{c\|\nabla f(x^k) - \nabla f(x^{k-1})\|}, \quad \forall k \geqslant 1 \\ x^{k+1} &= x^k - \alpha_k \nabla f(x^k), \quad \forall k \geqslant 0. \end{aligned} \tag{9}$$

In other words, the update in (9) is the same as in (7) except we removed the first constraint for α_k in (7) and introduced a constant factor c to make the second one more general.

Theorem 1. *For any $c \geqslant 1$ there exists x^0 such that the method (9) applied to f defined in (8) diverges.*

The formal proof of this statement is in Appendix A, but its main idea should be intuitively clear. First, observe that for x with a large absolute value, $f(x)$ behaves mostly like a linear function. However, $f'(x)$ approaches -1 when $x \to -\infty$ and $+1$ when $x \to +\infty$. Therefore, if x^k and x^{k-1} have the same sign, the local smoothness estimate will be too optimistic and x^{k+1} will "leapfrog" the optimum. In contrast, if the signs of x^k and x^{k-1} are different, then x^{k+1} will fail to get sufficiently close to the optimum. It is interesting to remark that on this function both versions of the Barzilai-Borwein method will diverge as well.

Consequently, the answer to the first question is affirmative: we do need some extra condition for the stepsize α_k.

Second bound. The answer to the second question is the opposite: it is indeed an artifact of our previous analysis. In the next section, we propose an improvement over the previous version [MM20]. We give a concise presentation in an informal way. We keep a more formal style for section 3 where an even better version (also slightly more complicated) will be presented.

2.1 Improving AdGD

The analysis of GD usually starts from the standard identity, followed by convexity inequality

$$\begin{aligned}\|x^{k+1} - x^*\|^2 &= \|x^k - \alpha_k \nabla f(x^k) - x^*\|^2 \\ &= \|x^k - x^*\|^2 - 2\alpha_k \langle \nabla f(x^k), x^k - x^* \rangle + \alpha_k^2 \|\nabla f(x^k)\|^2 \\ &\leqslant \|x^k - x^*\|^2 - 2\alpha_k \big(f(x^k) - f(x^*)\big) + \alpha_k^2 \|\nabla f(x^k)\|^2. \end{aligned} \tag{10}$$

In [MM20] the only "nontrivial" step in the proof was upper bounding $\alpha_k^2 \|\nabla f(x^k)\|^2$, that is $\|x^{k+1} - x^k\|^2$. Now we do it in a slightly different way. First, we need the following fact.

Lemma 1. For GD iterates (x^k) with arbitrary positive stepsizes, it holds

$$\langle \nabla f(x^k), \nabla f(x^{k-1}) \rangle \leqslant \|\nabla f(x^{k-1})\|^2. \tag{11}$$

Proof. This is just monotonicity of ∇f in disguise:

$$\begin{aligned}\|\nabla f(x^{k-1})\|^2 - \langle \nabla f(x^k), \nabla f(x^{k-1}) \rangle &= \langle \nabla f(x^{k-1}) - \nabla f(x^k), \nabla f(x^{k-1}) \rangle \\ &= \frac{1}{\alpha_{k-1}} \langle \nabla f(x^{k-1}) - \nabla f(x^k), x^{k-1} - x^k \rangle \geqslant 0. \end{aligned}$$ ■

Now we are going to bound $\|x^{k+1} - x^k\|^2$. For convenience, denote the approximate local Lipschitz constant as

$$L_k = \frac{\|\nabla f(x^k) - \nabla f(x^{k-1})\|}{\|x^k - x^{k-1}\|}.$$

Let α_k satisfy $\alpha_k \|\nabla f(x^k) - \nabla f(x^{k-1})\| \leqslant \gamma \|x^k - x^{k-1}\|$ for some $\gamma > 0$, that is $\alpha_k L_k \leqslant \gamma$. Using $\|u\|^2 = \|u - v\|^2 - \|v\|^2 + 2\langle u, v \rangle$, we have

$$\begin{aligned}\|x^{k+1} - x^k\|^2 &= \alpha_k^2 \|\nabla f(x^k)\|^2 \\ &= \alpha_k^2 \|\nabla f(x^k) - \nabla f(x^{k-1})\|^2 - \alpha_k^2 \|\nabla f(x^{k-1})\|^2 + 2\alpha_k^2 \langle \nabla f(x^k), \nabla f(x^{k-1}) \rangle \\ &= \alpha_k^2 L_k^2 \|x^k - x^{k-1}\|^2 - \alpha_k^2 \|\nabla f(x^{k-1})\|^2 + 2\alpha_k^2 \langle \nabla f(x^k), \nabla f(x^{k-1}) \rangle \\ &\overset{(11)}{\leqslant} \gamma^2 \|x^k - x^{k-1}\|^2 + \alpha_k^2 \langle \nabla f(x^k), \nabla f(x^{k-1}) \rangle \\ &= \gamma^2 \|x^k - x^{k-1}\|^2 + \alpha_k \theta_k \langle \nabla f(x^k), x^{k-1} - x^k \rangle \\ &\leqslant \gamma^2 \|x^k - x^{k-1}\|^2 + \alpha_k \theta_k (f(x^{k-1}) - f(x^k)), \end{aligned} \tag{12}$$

where the last inequality follows from convexity of f. For $\gamma < 1$ we can rewrite (12) as

$$\|x^{k+1} - x^k\|^2 \leqslant \frac{\gamma^2}{1-\gamma^2} \|x^k - x^{k-1}\|^2 - \frac{\gamma^2}{1-\gamma^2} \|x^{k+1} - x^k\|^2 + \frac{\alpha_k \theta_k}{1-\gamma^2} (f(x^{k-1}) - f(x^k)).$$

https://doi.org/10.52202/079017-3193

Substituting this inequality into (10) gives us

$$\|x^{k+1}-x^*\|^2+\frac{\gamma^2}{1-\gamma^2}\|x^{k+1}-x^k\|^2+\alpha_k\left(2+\frac{\theta_k}{1-\gamma^2}\right)(f(x^k)-f_*)$$
$$\leqslant \|x^k-x^*\|^2+\frac{\gamma^2}{1-\gamma^2}\|x^k-x^{k-1}\|^2+\frac{\alpha_k\theta_k}{1-\gamma^2}(f(x^{k-1})-f_*). \tag{13}$$

As we want to telescope the above inequality, we require

$$\frac{\alpha_k\theta_k}{1-\gamma^2}\leqslant\alpha_{k-1}\left(2+\frac{\theta_{k-1}}{1-\gamma^2}\right)\iff\alpha_k^2\leqslant\left(2(1-\gamma^2)+\theta_{k-1}\right)\alpha_{k-1}^2.$$

On the other hand, we have already used that $\alpha_k L_k\leqslant\gamma$. These two conditions lead to the bound

$$\alpha_k=\min\left\{\sqrt{2(1-\gamma^2)+\theta_{k-1}}\alpha_{k-1},\frac{\gamma}{L_k}\right\}, \tag{14}$$

where $\gamma\in(0,1)$ can be arbitrary. Now by playing with different values of γ, we obtain different instances of adaptive gradient descent method. For instance, by setting $\gamma=\frac{1}{\sqrt{2}}$, we get

$$\alpha_k=\min\left\{\sqrt{1+\theta_{k-1}}\alpha_{k-1},\frac{1}{\sqrt{2}L_k}\right\},$$

which is a strict improvement upon the original version in [MM20]. A simple reason why this is possible is that, unlike in [MM20], we did not resort to the Cauchy-Schwarz inequality and instead relied on transformation (12) and Lemma 1.

Algorithm 1 Adaptive gradient descent

Input: $x^0\in\mathbb{R}^d$, $\theta_0=0$, $\alpha_0>0$
$x^1=x^0-\alpha_0\nabla f(x^0)$
for $k=1,2,\ldots$ **do**
  $L_k=\frac{\|\nabla f(x^k)-\nabla f(x^{k-1})\|}{\|x^k-x^{k-1}\|}$
  $\alpha_k=\min\{\sqrt{1+\theta_{k-1}}\alpha_{k-1},\frac{1}{\sqrt{2}L_k}\}$
  $x^{k+1}=x^k-\alpha_k\nabla f(x^k)$
  $\theta_k=\frac{\alpha_k}{\alpha_{k-1}}$

Algorithm 2 Adaptive gradient descent-2

Input: $x^0\in\mathbb{R}^d$, $\theta_0=\frac{1}{3}$, $\alpha_0>0$
$x^1=x^0-\alpha_0\nabla f(x^0)$
for $k=1,2,\ldots$ **do**
  $L_k=\frac{\|\nabla f(x^k)-\nabla f(x^{k-1})\|}{\|x^k-x^{k-1}\|}$
  $\alpha_k=\min\left\{\sqrt{\frac{2}{3}+\theta_{k-1}}\alpha_{k-1},\frac{\alpha_{k-1}}{\sqrt{[2\alpha_{k-1}^2L_k^2-1]_+}}\right\}$
  $x^{k+1}=x^k-\alpha_k\nabla f(x^k)$
  $\theta_k=\frac{\alpha_k}{\alpha_{k-1}}$

We summarize the new scheme in Algorithm 1. We do not provide a formal proof for this scheme and hope that inequality (13) should be sufficient for the curious reader to complete the proof. In any case, the next section will contain a further improvement with all the missing proofs.

Remark 1. One might notice that we have used several times monotonicity of ∇f, where we actually could use a stronger property of cocoercivity (5). That is true, but we just prefer simplicity. We recommend work [Lat+23] that exploits cocoercivity in this framework.

3 Adaptive gradient descent: larger stepsize

In this section, we modify Algorithm 1 to use even larger steps resulting in Algorithm 2. This, however, will require a slightly more complex analysis.

Recall the notation $[t]_+=\max\{t,0\}$ and note that the second bound $\alpha_k\leqslant\frac{\alpha_{k-1}}{\sqrt{[2\alpha_{k-1}^2L_k^2-1]_+}}$ in step 5 of Algorithm 2 is equivalent to

$$\alpha_k^2L_k^2-\frac{\alpha_k^2}{2\alpha_{k-1}^2}\leqslant\frac{1}{2}, \tag{15}$$

which obviously allows for a larger range of α_k than $\alpha_k^2L_k^2\leqslant\frac{1}{2}$ in Algorithm 1. On the other hand, the first bound $\alpha_k\leqslant\sqrt{\frac{2}{3}+\theta_{k-1}}\alpha_{k-1}$ is definitely worse. At the moment, it is not even clear whether it allows α_k to increase.

Remark 2. A notable distinction between Algorithm 2 and Algorithm 1 is that the former allows to use a standard fixed step $\alpha_k = \frac{1}{L}$, provided that f is L-smooth. For instance, if we start from $\alpha_0 = \frac{1}{L}$ and use $L \geqslant L_k$ in every iteration (we can always use a larger value), then it follows from (15) and $\theta_{k-1} \geqslant \frac{1}{3}$ that $\alpha_k = \frac{1}{L}$ for all $k \geqslant 1$.

Algorithm 2 requires an initial stepsize α_0. While the algorithm converges for any value $\alpha_0 > 0$, it is important to choose initial step α_0 wisely. We suggest to do the following

$$\text{choose } \alpha_0 \text{ such that } \alpha_0 L_1 \in \left[\frac{1}{\sqrt{2}}, 2\right]. \tag{16}$$

The upper bound ensures that α_0 is not too large, while the lower ensures that it is not too small either. In most scenarios, this requires to run a linesearch, but we emphasize that it is only needed for the first iteration. Further discussion on this topic is in Appendix B.1.

We first prove that the sequence (x^k) is bounded and then derive the convergence result. Both statements are proved in Appendix B.2.

Lemma 2. The sequence (x^k) is bounded. In particular, for any solution x^* we have $x^k \in B(x^*, R)$, where

$$R^2 = \|x^0 - x^*\|^2 + 2\alpha_0^2\|\nabla f(x^0)\|^2 + \alpha_0(f(x^0) - f_*). \tag{17}$$

Theorem 2. *Let f be convex with a locally Lipschitz gradient ∇f, $x^0 \in \mathbb{R}^d$, and $\alpha_0 > 0$. Then the sequence (x^k) generated by Algorithm 2 converges to a solution of $\min_x f(x)$ and*

$$\min_{i\in[k]}\big(f(x^i) - f_*\big) \leqslant \frac{R^2}{2\sum_{i=1}^k \alpha_i}, \tag{18}$$

where R is defined as in (17). In particular, if α_0 satisfies (16), then

$$\min_{i\in[k]}\big(f(x^i) - f_*\big) \leqslant \frac{LR^2}{\sqrt{2}k}, \tag{19}$$

where L is the Lipschitz constant of ∇f over $B(x^*, R)$.

Of course, the important bound here is (18). The second bound only shows that our choice of stepsizes α_k cannot be too bad. The bound in (19) is stronger than the bound $\frac{\sqrt{3}LR^2}{2k}$, which could be obtained as a direct consequence of Lemma 7 with simple analysis. The derivation of the sharper bound as in (19) is presented in Appendix B.3 with, unfortunately, much more involved analysis.

4 Adaptive proximal gradient method

In this section, we turn to a more general problem of composite optimization,

$$\min_x F(x) \coloneqq f(x) + g(x), \tag{20}$$

where $g\colon \mathbb{R}^d \to (-\infty, +\infty]$ is a proper convex lsc function and $f\colon \mathbb{R}^d \to \mathbb{R}$ is a convex differentiable function with locally Lipschitz ∇f. Additionally, we assume that g is prox-friendly, that is we can efficiently compute its proximal mapping $\operatorname{prox}_g = (\mathrm{Id} + \partial g)^{-1}$.

We present Algorithm 3 that is a verbatim adaptation of Algorithm 2 with the proximal operator applied on top of the main update (similarly, it could be applied to Algorithm 1). However, its analysis is not a straightforward generalization. We encountered two issues in the proof:

- combining previous analysis of AdGD and the prox-mapping. As shown even in (13), we operate with the vectors x^k and x^{k-1} in terms of f. However, using the prox-inequality gives us the value $g(x^{k+1})$, which is not straightforward to combine with $f(x^k)$ and $f(x^{k-1})$.
- proving convergence of (x^k). The challenge arises from having a non-linear update due to the prox-mapping and allowing α_k to go to ∞, making the proof quite different from the traditional approach.

We define R in the same way as in (17)

$$R^2 = \|x^0 - x^*\|^2 + 2\alpha_0^2\|\widetilde{\nabla} F(x^0)\|^2 + \alpha_0(F(x^0) - F_*), \tag{21}$$

where $\widetilde{\nabla} F(x^0)$ denotes a subgradient of F at x^0.

https://doi.org/10.52202/079017-3193

Algorithm 3 Adaptive proximal gradient method

Input: $x^0 \in \mathbb{R}^d$, $\theta_0 = \frac{1}{3}$, $\alpha_0 > 0$
$x^1 = \operatorname{prox}_{\alpha_0 g}(x^0 - \alpha_0 \nabla f(x^0))$
for $k = 1, 2, \dots$ **do**
  $L_k = \dfrac{\|\nabla f(x^k) - \nabla f(x^{k-1})\|}{\|x^k - x^{k-1}\|}$
  $\alpha_k = \min\left\{\sqrt{\frac{2}{3} + \theta_{k-1}}\alpha_{k-1}, \frac{\alpha_{k-1}}{\sqrt{[2\alpha_{k-1}^2 L_k^2 - 1]_+}}\right\}$ // $[t]_+ := \max(t, 0)$
  $x^{k+1} = \operatorname{prox}_{\alpha_k g}(x^k - \alpha_k \nabla f(x^k))$
  $\theta_k = \frac{\alpha_k}{\alpha_{k-1}}$

Theorem 3. *Let f be convex with a locally Lipschitz gradient ∇f, g be convex lsc, $x^0 \in \mathbb{R}^d$, and $\alpha_0 > 0$. Then the sequence (x^k) generated by Algorithm 3 converges to a solution of (20) and*

$$\min_{i \in [k]} \big(F(x^i) - F_*\big) \leqslant \frac{R^2}{2\sum_{i=1}^k \alpha_i}. \tag{22}$$

In particular, if α_0 satisfies (16), then

$$\min_{i \in [k]} \big(F(x^i) - F_*\big) \leqslant \frac{LR^2}{\sqrt{2}k}, \tag{23}$$

where L is the Lipschitz constant of ∇f over $B(x^*, R)$.

5 Literature and discussion

Linesearch. There are many variants of linesearch procedures that go back to celebrated works of Goldstein [Gol62] and Armijo [Arm66]. We discuss an efficient implementation of the latter in detail in the next section. For other variants of linesearch, we refer to [BN16; Sal17].

Adagrad-type methods. Original Adagrad algorithm was proposed simultaneously in [DHS11] and [MS10]. The method has had a stunning impact on machine learning applications. It has also spawned a stream of various extensions that retain the same idea of using eventually decreasing steps. Because of this, its adaptivity is more prominent in the non-smooth regime, where stepsizes must be diminishing to guarantee convergence. Recent works [DM23; IHC23] have proposed ways to increase d_k in the update (2) and [KMJ23] even proved convergence of some Adagrad-type methods on smooth objectives. However, the stepsize in these methods eventually stops increasing, making them less adaptive.

In addition, Adagrad-type methods are usually sensitive to the initialization, as they either degrade in performance when $d_k = D$ and D is not chosen carefully, or their convergence rate depends multiplicatively on $\log(\|x^0 - x^*\|/d_0)$. In contrast, in our methods, the cost of estimating α_0 to satisfy condition (16) is additive and its impact vanishes as the total number of iterations increases.

Refined results on GD with a fixed stepsize. Paper [TV22] summarizes quite well the difficulty of GD analysis with large steps. In it, the authors derive sharp convergence bounds separately for two cases $\alpha L \in (0, 1]$ and $\alpha L \in (1, 2)$, and the latter case is considerably harder. In our analysis it is even harder, since the steps can go far beyond the global upper bound $\frac{2}{L}$. A surprising recent result [Gri23] showcases how little is understood in this case.

Small gradient. The lack-of-descent property makes it hard to deduce the $\mathcal{O}(1/k)$ rate for the last-iterate $\|\nabla f(x^k)\|$, which is known for GD with a fixed stepsize. We leave it as an open problem to establish a rate.

Extensions. Because the analysis of the algorithm is so special, it is not easy to extend it to basic generalizations of GD. However, some works have already built upon it. In [VMC21], the authors

consider a convex smooth minimization subject to linear constraints and combined the adaptive GD [MM20] with the Chambolle-Pock algorithm [CP10]. The authors of [Lat+23] went even further and considered a more general composite minimization problem subject to linear constraints, where the same two ideas as before were combined with a novel way of handling the prox mapping.

If we consider variational inequalities settings in the monotone case, then it is not clear how such adaptivity can help, since the most natural extension, the *forward-backward* method will diverge. Furthermore, the adaptive golden ratio algorithm [Mal19], which inspired the development of AdProxGD, already includes all the features that AdProxGD has.

6 Experiments

In the experiments[2] we compare our method to the ProxGD with Armijo's linesearch. We believe it is the best and arguably the most popular alternative to our method. An efficient implementation of Armijo's linesearch requires two parameters, $s > 1$ and $r < 1$. In the k-th iteration, the first iteration of linesearch starts from $\alpha_k = s\alpha_{k-1}$, that is, we want to try a slightly larger step than in the previous iteration. If linesearch does not terminate, we start decreasing a stepsize geometrically with a ratio r. Formally, we are looking for the largest $\alpha_k = sr^i\alpha_{k-1}$, for $i = 0, 1, \ldots$, such that for $x^{k+1} = \operatorname{prox}_{\alpha_k g}(x^k - \alpha_k \nabla f(x^k))$ it holds that

$$f(x^{k+1}) \leqslant f(x^k) + \langle \nabla f(x^k), x^{k+1} - x^k \rangle + \frac{1}{2\alpha_k}\|x^{k+1} - x^k\|^2.$$

It is evident that each iteration of this linesearch requires one evaluation of f and prox_g. However, it is important to highlight that in some cases, the last evaluation of $f(x^{k+1})$ (during linesearch) may not incur any additional costs, as certain expensive operations, such as matrix-vector multiplication, can be reused to compute the next gradient $\nabla f(x^{k+1})$. Throughout our comparisons, we consistently took these factors into account and reported only essential operations that cannot be further reused.

Our legend will stay the same for all plots:

AdProxGD (1.2, 0.5) (1.5, 0.8) (1.1, 0.5) (1.2, 0.9)
(1.1, 0.9) (1.5, 0.5) (1.2, 0.8) (1.1, 0.8) (1.5, 0.9)

where each pair of numbers represents (s, r) for ProxGD with linesearch described above. As we will see, the choice of (s, r) matters a lot. More experiments are provided in the Appendix D.

Maximum likelihood estimate of the information matrix. We consider [BV04, Equation (7.5)], where our goal is to estimate the inverse of a covariance matrix Y subject to eigenvalue bounds. Formally, this problem can be formulated as follows

$$\min_{X \in \mathbb{S}^n} f(X) = \log\det X - \operatorname{tr}(XY) \quad \text{subject to} \quad lI \preccurlyeq X \preccurlyeq uI, \tag{24}$$

where $\mathbb{S}^n$ denotes the space of n-by-n symmetric matrices and $A \preccurlyeq B$ means that $B - A$ is positive semidefinite.

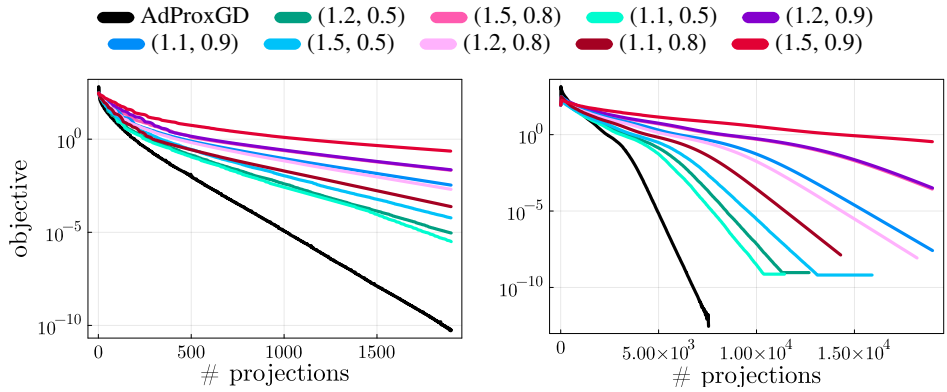
Computing projection onto the constraint set $\mathcal{C} = \{X : lI \preccurlyeq X \preccurlyeq uI\}$ requires computing matrix eigendecomposition. However, it is noteworthy that once the eigendecomposition is computed, both the objective and gradient evaluations can be carried out at a low cost. Consequently, when comparing methods, we only emphasized the number of projections conducted. We generated a random $y \in \mathbb{R}^n$ with entries from $N(0, 10)$ and $\delta_i \in \mathbb{R}^n$ with entries from $N(0, 1)$, and then set $y_i = y + \delta_i$, for $i = 1, \ldots, M$. Then we computed $Y = \frac{1}{M}\sum_{i=1}^M y_i y_i^\top$. The results are presented in Figure 1. For two scenarios we generated, the proposed method converged faster than any of the linesearch versions.

Acknowledgments and Disclosure of Funding

The authors would like to thank Puya Latafat, who found a subtle error in the convergence proof of (x^k) in Theorem 3 in the first version of this manuscript. Thanks to him, we were able to simplify our proof considerably. We also thank anonymous reviewers who found many typos and inaccuracies. Yura Malitsky's research was partially funded by the Austrian Science Fund (FWF) [10.55776/STA223].

[2] https://github.com/ymalitsky/AdProxGD

 https://doi.org/10.52202/079017-3193



(a) $n = 100$, $l = 0.1$, $u = 10$, $M = 50$ (b) $n = 50$, $l = 0.1$, $u = 1000$, $M = 100$

Figure 1: Maximum likelihood estimate, problem (24)

References

[Arm66] L. Armijo. "Minimization of functions having Lipschitz continuous first partial derivatives". In: *Pacific J. Math.* 16.1 (1966), pp. 1–3. DOI: 10.2140/pjm.1966.16.1.

[BB88] J. Barzilai and J. M. Borwein. "Two-point step size gradient methods". In: *IMA J Numer Anal* 8.1 (1988), pp. 141–148. DOI: 10.1093/imanum/8.1.141.

[BN16] J. Y. Bello Cruz and T. T. Nghia. "On the convergence of the forward–backward splitting method with linesearches". In: *Optim. Methods Softw.* 31.6 (2016), pp. 1209–1238.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. DOI: 10.1017/cbo9780511804441.

[CP10] A. Chambolle and T. Pock. "A first-order primal-dual algorithm for convex problems with applications to imaging". In: *J Math Imaging Vis* 40.1 (2010), pp. 120–145. DOI: 10.1007/s10851-010-0251-1.

[DM23] A. Defazio and K. Mishchenko. "Learning-rate-free learning by D-adaptation". In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. 2023, pp. 7449–7479.

[DHS11] J. Duchi, E. Hazan, and Y. Singer. "Adaptive subgradient methods for online learning and stochastic optimization". In: *J. Mach. Learn. Res.* 12 (2011), pp. 2121–2159.

[Gol62] A. A. Goldstein. "Cauchy's method of minimization". In: *Numer. Math.* 4.1 (1962), pp. 146–150. DOI: 10.1007/bf01386306.

[Gri23] B. Grimmer. "Provably faster gradient descent via long steps". In: (2023). arXiv: 2307.06324.

[IHC23] M. Ivgi, O. Hinder, and Y. Carmon. "DoG is SGD's best friend: A parameter-free dynamic step size schedule". In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. 2023, pp. 14465–14499.

[KMJ23] A. Khaled, K. Mishchenko, and C. Jin. "DoWG unleashed: An efficient universal parameter-free gradient descent method". In: (2023). arXiv: 2305.16284.

[Lat+23] P. Latafat, A. Themelis, L. Stella, and P. Patrinos. "Adaptive proximal algorithms for convex optimization under local Lipschitz continuity of the gradient". In: (2023). arXiv: 2301.04431.

[Mal19] Y. Malitsky. "Golden ratio algorithms for variational inequalities". In: *Math. Program.* 184.1-2 (2019), pp. 383–410. DOI: 10.1007/s10107-019-01416-w. arXiv: 1803.08832.

[MM20] Y. Malitsky and K. Mishchenko. "Adaptive gradient descent without descent". In: *Proceedings of the 37th International Conference on Machine Learning*. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 6702–6712. arXiv: 1910.09529.

[MS10] H. B. McMahan and M. Streeter. "Adaptive bound optimization for online convex optimization". In: *Proceedings of the 23rd Annual Conference on Learning Theory (COLT)*. 2010.

[Sal17] S. Salzo. "The variable metric forward-backward splitting algorithm under mild differentiability assumptions". In: *SIAM J. Optim.* 27.4 (2017), pp. 2153–2181. DOI: 10.1137/16m1073741.

[TV22] M. Teboulle and Y. Vaisbourd. "An elementary approach to tight worst case complexity analysis of gradient based methods". In: *Math. Program.* 201.1-2 (2022), pp. 63–96. DOI: 10.1007/s10107-022-01899-0.

[VMC21] M.-L. Vladarean, Y. Malitsky, and V. Cevher. "A first-order primal-dual method with adaptivity to local smoothness". In: *NeurIPS*. Vol. 34. 2021, pp. 6171–6182. arXiv: 2110.15148.

Appendix

A Counterexample

Lemma 3. The function f defined in (8) satisfies the following properties:

1. f is convex.
2. f' is L-Lipschitz with $L = 1$.
3. f is locally strongly convex, i.e., for any bounded set $\mathcal{X}$ there exists a constant $\mu_{\mathcal{X}} > 0$ such that $|f'(x) - f'(y)| \geqslant \mu_{\mathcal{X}}|x - y|$ for any $x, y \in \mathcal{X}$.
4. $|f'(x)| \leqslant G$ with $G = 2$.
5. f is 2-Lipschitz.

Proof. First, let us find f' and f'':

$$f'(x) = \begin{cases} x, & x \in [-1,1] \\ \frac{ax}{1+|x|}, & x \notin [-1,1] \end{cases}, \qquad f''(x) = \begin{cases} 1, & x \in (-1,1) \\ \frac{a}{(1+|x|)^2}, & x \notin [-1,1] \end{cases},$$

so indeed $a = 2$ and $b = 2\log 2 - \frac{3}{2}$. Convexity of f follows from the fact that $f''(x) > 0$ for any x. Lipschitzness of f' follows directly from the bound $f''(x) \leqslant 1$ for all x. Similarly, local strong convexity follows from the bound $f''(x) \geq \frac{1}{\max_{z\in\mathcal{X}}(1+|z|)^2} =: \mu_{\mathcal{X}}$ for any $x \in \mathcal{X}$. Finally, the last two properties trivially follow from the expression for $f'(x)$. ■

Proof of Theorem 1. Let us choose $x^0 = r + 2$ with a sufficiently large $r > 6c$. This readily implies that

$$x^1 = x^0 - \frac{2x^0}{1+x^0} = \frac{x^0(x^0-1)}{x^0+1} > x^0 - 2 = r.$$

Our goal is to show that the iterates follow a very specific pattern. Namely, we prove that for all $k \geqslant 0$,

$$\operatorname{sign}(x^{2k}) = \operatorname{sign}(x^{2k+1}), \quad \operatorname{sign}(x^{2k+2}) \neq \operatorname{sign}(x^{2k}), \quad |x^{2k+2}| > 2|x^{2k+1}| > |x^{2k}|.$$

If this condition holds true, then the sequence (x^k) must be divergent.

First, observe that if $|x^k|, |x^{k-1}| \geqslant r$ and $\operatorname{sign}(x^k) = \operatorname{sign}(x^{k-1})$, then the smoothness estimate admits a simple expression:

$$\begin{aligned} L_k &= \frac{|f'(x^k) - f'(x^{k-1})|}{|x^k - x^{k-1}|} = \frac{2\left|\frac{x^k}{1+|x^k|} - \frac{x^{k-1}}{1+|x^{k-1}|}\right|}{|x^k - x^{k-1}|} = \frac{2\left|\frac{|x^k|}{1+|x^k|} - \frac{|x^{k-1}|}{1+|x^{k-1}|}\right|}{||x^k| - |x^{k-1}||} \\ &= \frac{2}{(1+|x^k|)(1+|x^{k-1}|)} < \frac{2}{r(1+|x^k|)}. \end{aligned}$$

Therefore, in that case $\alpha_k|f'(x^k)| > \frac{r(1+|x^k|)}{2c}|f'(x^k)| = \frac{r|x^k|}{c} > 3|x^k|$. Since $\operatorname{sign}(f'(x^k)) = \operatorname{sign}(x^k)$, it implies that $|x^{k+1}| > 2|x^k|$ and $\operatorname{sign}(x^{k+1}) \neq \operatorname{sign}(x^k)$.

Next, if $|x^k|, |x^{k-1}| \geqslant r > 3$ with $|x^k| \geqslant 2|x^{k-1}|$ and $\operatorname{sign}(x^k) \neq \operatorname{sign}(x^{k-1})$, then we have $\frac{2|x^k|}{1+|x^k|} > \frac{3}{2}$ and

$$\begin{aligned} L_k &= \frac{|f'(x^k) - f'(x^{k-1})|}{|x^k - x^{k-1}|} = \frac{2\left|\frac{x^k}{1+|x^k|} - \frac{x^{k-1}}{1+|x^{k-1}|}\right|}{|x^k - x^{k-1}|} = \frac{2\left(\frac{|x^k|}{1+|x^k|} + \frac{|x^{k-1}|}{1+|x^{k-1}|}\right)}{|x^k| + |x^{k-1}|} \\ &> \frac{3}{|x^k| + |x^{k-1}|} > \frac{3}{|x^k| + \frac{1}{2}|x^k|} \geqslant \frac{2}{|x^k|}. \end{aligned}$$

https://doi.org/10.52202/079017-3193

This implies $\alpha_k < \frac{|x^k|}{2c} \leqslant \frac{|x^k|}{2}$. Since $\operatorname{sign}(f'(x^k)) = \operatorname{sign}(x^k)$ and $\frac{\alpha_k}{1+|x^k|} \leqslant \frac{|x^k|}{2(1+|x^k|)} < \frac{1}{2}$, we conclude that $\operatorname{sign}(x^{k+1}) = \operatorname{sign}(x^k)$ and

$$|x^{k+1}| = \left|x^k - \alpha_k \frac{x^k}{1+|x^k|}\right| = |x^k| \left(1 - \frac{\alpha_k}{1+|x^k|}\right) > \frac{1}{2}|x^k|.$$

As x^0 and x^1 satisfy the first case, by induction we deduce that all iterates (x^k) follow the described pattern. ■

B Analysis of Algorithm 2

B.1 Initial stepsize (expanded discussion)

Algorithm 2 requires an initial stepsize α_0. While the algorithm converges for any value $\alpha_0 > 0$ with the rate

$$\min_{i\in[k]}(f(x^i) - f_*) \leqslant \frac{R^2}{2\sum_{i=1}^k \alpha_i},$$

(see eq. (17) for the definition of R), the choice of α_0 will impact further steps due to the bound $\alpha_k \leqslant \sqrt{2/3 + \theta_{k-1}}\alpha_{k-1}$. Because of this reason, we do not want to choose α_0 too small. On the other hand, too large α_0 will make R large. To counterbalance these two extremes, we suggest to do the following:

$$\text{choose } \alpha_0 \text{ such that } \alpha_0 L_1 \in \left[\frac{1}{\sqrt{2}}, 2\right]. \tag{25}$$

The upper bound ensures that α_0 is not too large, while the lower ensures that it is not too small either. In most scenarios, this requires to run a linesearch, but we emphasize one more time: it is only needed for the first iteration. In some sense, our condition (16) is similar to classical Goldstein's rule [Gol62] on selecting the stepsize: not too small and not too big.

Of course, if we start with a very small α_0, only the first bound for α_k will be active for some time, and we will eventually reach a reasonable range for a stepsize. However, linesearch with a more aggressive factor (say, 10) will allow us to reach this range faster. If we start with $\alpha_0 = 10^{-8}$ when in fact a reasonable range for steps in this region is $[1, 10]$, then we will need at least 100 iterations of our method, while linesearch with a factor 10 will find it in less than 10 iterations.

It may happen that the problem is degenerated in the sense that for any α_0, $\alpha_0 L_1 < \frac{1}{\sqrt{2}}$. In other words, increasing α_0 leads to decreasing L_1 and linesearch may never stop. In this case we should terminate a linesearch after α_0 reaches any prescribed value, say 1.

B.2 Analysis of Algorithm 2

Lemma 4. For iterates (x^k) of Algorithm 2 it holds

$$\|x^{k+1} - x^k\|^2 \leqslant \frac{1}{2}\|x^k - x^{k-1}\|^2 + \frac{3}{2}\alpha_k\theta_k(f(x^{k-1}) - f(x^k)). \tag{26}$$

Before we continue, let us give some intuition for this lemma. Its analysis follows mostly the same steps as in (12). However, now we will split $\alpha_k^2\|\nabla f(x^{k-1})\|^2$ into two parts and use one of it to improve the smoothness bound for α_k.

Proof. We start from the third line in (12) and then apply the above-mentioned splitting:

$$\begin{aligned}
\|x^{k+1} - x^k\|^2 &= \alpha_k^2 L_k^2\|x^k - x^{k-1}\|^2 - \alpha_k^2\|\nabla f(x^{k-1})\|^2 + 2\alpha_k^2\langle\nabla f(x^k), \nabla f(x^{k-1})\rangle \\
&= \left(\alpha_k^2 L_k^2 - \frac{\alpha_k^2}{2\alpha_{k-1}^2}\right)\|x^k - x^{k-1}\|^2 - \frac{\alpha_k^2}{2}\|\nabla f(x^{k-1})\|^2 + 2\alpha_k^2\langle\nabla f(x^k), \nabla f(x^{k-1})\rangle \\
&\overset{(15)\&(11)}{\leqslant} \frac{1}{2}\|x^k - x^{k-1}\|^2 + \frac{3}{2}\alpha_k^2\langle\nabla f(x^k), \nabla f(x^{k-1})\rangle \\
&= \frac{1}{2}\|x^k - x^{k-1}\|^2 + \frac{3}{2}\alpha_k\theta_k\langle\nabla f(x^k), x^{k-1} - x^k\rangle.
\end{aligned} \tag{27}$$

Convexity of f completes the proof. ■

Lemma 5. For iterates (x^k) of Algorithm 2 and any solution x^* it holds

$$\begin{aligned}&\|x^{k+1}-x^*\|^2+\|x^{k+1}-x^k\|^2+\alpha_k(2+3\theta_k)(f(x^k)-f_*)\\&\leqslant\|x^k-x^*\|^2+\|x^k-x^{k-1}\|^2+3\alpha_k\theta_k(f(x^{k-1})-f_*).\end{aligned}\tag{28}$$

Proof. From (26) we have

$$\|x^{k+1}-x^k\|^2\leqslant\|x^k-x^{k-1}\|^2-\|x^{k+1}-x^k\|^2+3\alpha_k\theta_k(f(x^{k-1})-f(x^k)).\tag{29}$$

Using this inequality in (10), we get

$$\begin{aligned}&\|x^{k+1}-x^*\|^2+\|x^{k+1}-x^k\|^2+2\alpha_k(f(x^k)-f_*)\\&\leqslant\|x^k-x^*\|^2+\|x^k-x^{k-1}\|^2+3\alpha_k\theta_k(f(x^{k-1})-f(x^k)),\end{aligned}$$

which is equivalent to (28). ■

Proof of Lemma 2. The first bound for α_k in Algorithm 2 gives us $3\alpha_k\theta_k\leqslant(2+3\theta_{k-1})\alpha_{k-1}$. We use it in (28) and telescope then to obtain

$$\begin{aligned}&\|x^{k+1}-x^*\|^2+\|x^{k+1}-x^k\|^2+\alpha_k(2+3\theta_k)(f(x^k)-f_*)\\&\leqslant\|x^1-x^*\|^2+\|x^1-x^0\|^2+\alpha_0(2+3\theta_0)(f(x^0)-f_*).\end{aligned}\tag{30}$$

This immediately implies that (x^k) is bounded, but we would like to obtain the bound without an intermediate iterate x^1. From (10) we know that

$$\|x^1-x^*\|\leqslant\|x^0-x^*\|^2+\alpha_0^2\|\nabla f(x^0)\|^2-2\alpha_0(f(x^0)-f_*).$$

Combining it with (30), we deduce

$$\begin{aligned}&\|x^{k+1}-x^*\|^2+\|x^{k+1}-x^k\|^2+\alpha_k(2+3\theta_k)(f(x^k)-f_*)\\&\leqslant\|x^0-x^*\|^2+2\alpha_0^2\|\nabla f(x^0)\|^2+3\theta_0\alpha_0(f(x^0)-f_*).\end{aligned}$$

Using that $\theta_0=\frac{1}{3}$ completes the proof. ■

Remark 3. We could have used $\theta_0=0$ as we did in Algorithm 1 which would have improved the final constant R. However, since the first bound for α_k is worse this time, we would need a more complicated initial bound for α_0. We decided to keep it simple.

Notation. For brevity, we write $\alpha_k\mapsto 1$ to denote that in the k-th iteration α_k satisfies the first bound, that is $\alpha_k=\sqrt{\frac{2}{3}+\theta_{k-1}}$. Similarly, for $\alpha_k\mapsto 2$. Also let L be the Lipschitz constant of ∇f over the set $B(x^*,R)$. This means that $L_k\leqslant L$ for all k.

Next few statements are not very important for the first reading, as they only concern with a lower bound of $\sum_{i=1}^k\alpha_i$. The main statement in Theorem 2 is valid independently of them, so the reader can go directly there.

Lemma 6. If $\alpha_k\mapsto 2$, then $\alpha_k\geqslant\frac{1}{\sqrt{2}L}$ and $\alpha_{k-1}+\alpha_k\geqslant\frac{2}{L}$.

Proof. Note that in this case $\alpha_k=\frac{\alpha_{k-1}}{\sqrt{2\alpha_{k-1}^2L_k^2-1}}$, and hence $\frac{1}{\alpha_{k-1}^2}+\frac{1}{\alpha_k^2}=2L_k^2$. This implies that $\alpha_k\geqslant\frac{1}{\sqrt{2}L_k}\geqslant\frac{1}{\sqrt{2}L}$. By AM-GM inequality,

$$\left(\frac{1}{\alpha_{k-1}^2}+\frac{1}{\alpha_k^2}\right)(\alpha_{k-1}+\alpha_k)^2\geqslant\frac{2}{\alpha_{k-1}\alpha_k}\cdot 4\alpha_{k-1}\alpha_k=8$$

and the conclusion $\alpha_{k-1}+\alpha_k\geqslant\sqrt{\frac{8}{2L_k^2}}=\frac{2}{L_k}$ follows. ■

Lemma 7. If α_0 satisfies (16), then $\alpha_k\geqslant\frac{1}{\sqrt{3}L}$ for all $k\geqslant 1$.

https://doi.org/10.52202/079017-3193

Proof. We use induction. For $k = 1$, we have either $\alpha_1 = \sqrt{\frac{2}{3} + \theta_0}\alpha_0 \geqslant \frac{1}{\sqrt{2}L}$ or $\alpha_1 \mapsto 2$, which in view of Lemma 6 also implies $\alpha_1 \geqslant \frac{1}{\sqrt{2}L}$.

Suppose that $\alpha_{k-1} \geqslant \frac{1}{\sqrt{3}L}$ and we must show that $\alpha_k \geqslant \frac{1}{\sqrt{3}L}$. If $\alpha_k \mapsto 2$, then we are done. Therefore, suppose that $\alpha_k \mapsto 1$. Consider two options for α_{k-1}. If $\alpha_{k-1} \mapsto 1$, then $\theta_{k-1} \geqslant \sqrt{2/3}$. Thus, for α_k we have that

$$\alpha_k \geqslant \sqrt{\frac{2}{3} + \sqrt{\frac{2}{3}}}\alpha_{k-1} \geqslant \alpha_{k-1} \geqslant \frac{1}{\sqrt{3}L}.$$

If $\alpha_{k-1} \mapsto 2$, then $\alpha_{k-1} \geqslant \frac{1}{\sqrt{2}L}$ and hence

$$\alpha_k = \sqrt{\frac{2}{3} + \theta_{k-1}}\alpha_{k-1} \geqslant \sqrt{\frac{2}{3}} \cdot \frac{1}{\sqrt{2}L} = \frac{1}{\sqrt{3}L},$$

which completes the proof. ■

Remark 4. It is clear from above proof that condition $\alpha_0 \geqslant \frac{1}{\sqrt{2}L_1}$ from (16) was used only to give us the basis for induction. Without that condition, one can still show in the same way that $\alpha_k \geqslant \min\{\alpha_0, \frac{1}{\sqrt{3}L}\}$.

Summing this result from 1 to k yields $\sum_{i=1}^k \alpha_i \geqslant \frac{k}{\sqrt{3}L}$. The stepsize in the previous section is lower bounded by a $\frac{k}{\sqrt{2}L}$, so it is natural to wonder: why does the current section contain a "larger stepsize"? The answer is that while we cannot show that each individual step is larger, we still show in the next theorem that its *total* length will be lower bounded by the same quantity.

Proof of Theorem 2. We proceed in the same way as in Lemma 2, but this time we keep all the terms that were discarded earlier. Specifically, summing (28) over all k yields

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 + \|x^{k+1} - x^k\|^2 \\
&+ \alpha_k(2 + 3\theta_k)(f(x^k) - f_*) + \sum_{i=1}^{k-1}(\alpha_i(2 + 3\theta_i) - 3\alpha_{i+1}\theta_{i+1})(f(x^i) - f_*) \\
&\leqslant \|x^1 - x^*\|^2 + \|x^1 - x^0\|^2 + 3\alpha_1\theta_1(f(x^0) - f_*) \\
&\leqslant \|x^0 - x^*\|^2 + 2\alpha_0^2\|\nabla f(x^0)\|^2 + \alpha_0(f(x^0) - f_*) = R^2,
\end{aligned} \tag{31}$$

where the last two bounds follow from the same arguments as in Lemma 2. Note that each factor $(\alpha_k(2 + 3\theta_k) - 3\alpha_{k+1}\theta_{k+1})$ is nonnegative and their sum is

$$\alpha_k(2 + 3\theta_k) + \sum_{i=1}^{k-1}(\alpha_i(2 + 3\theta_i) - 3\alpha_{i+1}\theta_{i+1}) = 2\sum_{i=1}^{k}\alpha_i + 3\theta_1\alpha_1 \geqslant 2\sum_{i=1}^{k}\alpha_i.$$

Hence, we readily obtain that

$$\min_{i\in[k]}(f(x^i) - f_*) \leqslant \frac{R^2}{2\sum_{i=1}^k \alpha_i}.$$

In particular, if α_0 satisfies (16), then inequality (19) is a direct consequence of Lemma 11, which we prove in the next section.

It remains to prove that (x^k) converges to a solution. The next arguments will be similar to the ones in [MM20]. We have already proved that (x^k) is bounded. As f is L-smooth over $B(x^*, R)$, we have

$$f(x^*) - f(x^k) \geqslant \langle \nabla f(x^k), x^* - x^k\rangle + \frac{1}{2L}\|\nabla f(x^k)\|^2.$$

Using this sharper bound instead of plain convexity in (10) and repeating the same arguments as in Lemma 5, we end up with the same inequality plus the extra term

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 + \|x^{k+1} - x^k\|^2 + \alpha_k(2 + 3\theta_k)(f(x^k) - f_*) + \frac{\alpha_k}{L}\|\nabla f(x^k)\|^2 \\
&\leqslant \|x^k - x^*\|^2 + \|x^k - x^{k-1}\|^2 + 3\alpha_k\theta_k(f(x^{k-1}) - f_*).
\end{aligned} \tag{32}$$

Now, by telescoping this inequality we infer that $\sum_{i=1}^{k} \frac{\alpha_i}{L}\|\nabla f(x^i)\|^2 \leqslant R^2$. Since the sequence (α_k) is separated from 0 (note that this is independent of condition (16) by Remark 4), we conclude that $\nabla f(x^k) \to 0$ as $k \to \infty$. Hence, all limit points of (x^k) are solutions. Applying $3\theta_k\alpha_k \leqslant (2+3\theta_{k-1})\alpha_{k-1}$ in (32) we get

$$\|x^{k+1}-x^*\|^2 + b_{k+1} \leqslant \|x^k - x^*\|^2 + b_k,$$

where $b_k = \|x^k - x^{k-1}\|^2 + \alpha_{k-1}(2+3\theta_{k-1})(f(x^{k-1}) - f_*)$. Then the convergence of (x^k) to a solution follows from the standard Opial-type arguments. ■

B.3 Better bounds for the sum of stepsizes

In this section, we prove the bound $\sum_{i=1}^{k} \alpha_i \geqslant \frac{k}{\sqrt{2}L}$.

Lemma 8. If $\theta_k < \frac{1}{3}$, then $\alpha_k \mapsto 2$ and $\alpha_{k-1}L_k > \sqrt{5}$, $\alpha_{k-2}L_k \geqslant \frac{3}{2}$, $\alpha_{k-3}L_k \geqslant 1$.

Proof. By definition, $\alpha_k \mapsto 1$ means that $\alpha_k = \sqrt{\frac{2}{3}+\theta_{k-1}}\alpha_{k-1}$ and thus $\theta_k \geqslant \sqrt{\frac{2}{3}}$. Hence, $\alpha_k \mapsto 2$. Then we have that $\theta_k = \frac{1}{\sqrt{2\alpha_{k-1}^2 L_k^2 - 1}} < \frac{1}{3}$ which implies $\alpha_{k-1}L_k > \sqrt{5}$. Since we get a large α_{k-1}, the first bound on stepsizes does not allow previous steps to be much smaller. That is the idea we shall use.

For any k, we have that $\theta_k \leqslant \sqrt{\frac{2}{3}+\theta_{k-1}}$. As $\theta_0 \leqslant 1$, it is trivial to prove that $\theta_k \leqslant \frac{1+\sqrt{\frac{11}{3}}}{2} =: t_0$, which is the root of $t - \sqrt{\frac{2}{3}+t} = 0$. From $\alpha_{k-1}L_k > \sqrt{5}$, it follows that

$$\sqrt{5} < \alpha_{k-1}L_k \leqslant \sqrt{\frac{2}{3}+\theta_{k-2}}\alpha_{k-2}L_k \leqslant t_0\alpha_{k-2}L_k.$$

Hence, to prove $\alpha_{k-2}L_k \geqslant \frac{3}{2}$, it only remains to check that $\frac{\sqrt{5}}{t_0} \geqslant \frac{3}{2}$.

Similarly, we have

$$\frac{3}{2} \leqslant \alpha_{k-2}L_k \leqslant \sqrt{\frac{2}{3}+\theta_{k-3}}\alpha_{k-3}L_k \leqslant t_0\alpha_{k-3}L_k.$$

And to prove $\alpha_{k-3}L_k \geqslant 1$, we must check that $\frac{3}{2t_0} \geqslant 1$. ■

Given the sequence $(\alpha_k)_{k\geq 1}$, we call its element α_m a *breakpoint*, if $\theta_m < \frac{1}{3}$ and $\alpha_m < \frac{1}{L}$. The next lemma says that a small step can only occur shortly after a breakpoint.

Lemma 9. If $\alpha_k < \frac{1}{\sqrt{2}L}$, then exactly one of the following holds

(i) α_{k-1} is a breakpoint;

(ii) $\alpha_{k-1} < \alpha_k$ and α_{k-2} is a breakpoint.

Proof. In view of Lemma 6, the statement implies that $\alpha_k \mapsto 1$. Suppose that α_{k-1} is not a breakpoint, since otherwise we are done. This means that either (a) $\alpha_{k-1} \geqslant \frac{1}{L}$ or (b) $\alpha_{k-1} < \frac{1}{L}$ and $\theta_{k-1} \geqslant \frac{1}{3}$. In the first case we immediately get a contradiction, since $\alpha_k = \sqrt{\frac{2}{3}+\theta_{k-1}}\alpha_{k-1} \geqslant \sqrt{\frac{2}{3}}\frac{1}{L} > \frac{1}{\sqrt{2}L}$. Then if we consider (b), the bound $\theta_{k-1} \geqslant \frac{1}{3}$ implies that $\alpha_{k-1} \leqslant \alpha_k < \frac{1}{\sqrt{2}L}$. Then we can apply the same arguments as above, but to α_{k-1}. This means that either α_{k-2} will be a breakpoint or we will have a chain $\alpha_{k-2} \leqslant \alpha_{k-1} \leqslant \alpha_k < \frac{1}{\sqrt{2}L}$. However, the latter option cannot occur, because using $\theta_{k-1} \geqslant 1$ and $\alpha_{k-1} \geqslant \frac{1}{\sqrt{3}L}$ ensure us that

$$\alpha_k = \sqrt{\frac{2}{3}+\theta_{k-1}}\alpha_{k-1} \geqslant \sqrt{\frac{2}{3}+1}\frac{1}{\sqrt{3}L} = \frac{\sqrt{5}}{3L} > \frac{1}{\sqrt{2}L}.$$

■

 https://doi.org/10.52202/079017-3193

Although a breakpoint indicates that we are in the region with a small stepsize, Lemma 8 guarantees that previous steps were quite large. The next lemma shows that in total we make significant progress.

Lemma 10. If α_m is a breakpoint, then $\sum_{j=-2}^{2} \alpha_{m+j} > \frac{5}{L}$.

Proof. If α_m is a breakpoint, then on one hand Lemma 8 implies that $\alpha_{m-1} \geqslant \frac{\sqrt{5}}{L_m}$, $\alpha_{m-2} \geqslant \frac{3}{2L_m}$. On the other hand, we have that $\alpha_m \geqslant \frac{1}{\sqrt{2}L}$, $\alpha_{m+1} \geqslant \frac{1}{\sqrt{3}L}$, and $\alpha_{m+2} \geqslant \frac{1}{\sqrt{3}L}$. Combining, we get

$$\sum_{j=-2}^{2} \alpha_{m+j} L \geqslant \frac{3}{2} + \sqrt{5} + \frac{1}{\sqrt{2}} + \frac{2}{\sqrt{3}} > 5.59.$$

■

Lemma 11. If α_0 satisfies (16), then for any $k \geqslant 1$ we have

$$\sum_{i=1}^{k} \alpha_i \geqslant \frac{k}{\sqrt{2}L}. \tag{33}$$

Proof. Let $\mathcal{M} = \{m \text{ is a breakpoint: } \alpha_{m+1} < \frac{1}{\sqrt{2}L}\}$. We can split $\sum_{i=1}^{k} \alpha_i$ into two terms as

$$\sum_{i=1}^{k} \alpha_i = \sum_{m \in \mathcal{M}} \sum_{j=-2}^{2} \alpha_{m+j} + \text{rest}. \tag{34}$$

We claim that elements in the "rest" are greater or equal than $\frac{1}{\sqrt{2}L}$. Indeed, if $\alpha_i < \frac{1}{\sqrt{2}L}$ is in the "rest" term, then either α_{i-1} is a breakpoint or $\alpha_{i-1} < \frac{1}{\sqrt{2}L}$ and α_{i-2} is a breakpoint, as Lemma 9 suggests. In either case, α_i must be included in the first sum, by the definition of $\mathcal{M}$.

Now let us estimate both terms. The first sum in (34) is greater than $\frac{5|\mathcal{M}|}{L} > \frac{5|\mathcal{M}|}{\sqrt{2}L}$, by Lemma 10. The total sum in the "rest" term is not less than $\frac{k-5|\mathcal{M}|}{\sqrt{2}L}$. Hence, the desired inequality follows. It has to be only noted that if $k-1 \in \mathcal{M}$, we have to additionally consider the sum $\sum_{j=-2}^{1} \alpha_{k-1+j} \geqslant \frac{4}{\sqrt{2}L}$, for which the bound follows from the same arguments as in Lemma 10. ■

Remark 5. It is obvious that our analysis was not optimal. For instance, whenever $\alpha_k \mapsto 2$, we could use $\alpha_{k-1} + \alpha_k \geqslant \frac{2}{L}$ instead of more conservative $\frac{2}{\sqrt{2}L}$. Similarly, we got a much better bound for every breakpoint. However, we did not want to overcomplicate an already tedious examination. We leave it as an open question if one can provide a bound closer to $\frac{k}{L}$ (or better?) with a readable proof.

C Adaptive proximal gradient method

Recall that the second bound for the stepsize α_k is equivalent to

$$\alpha_k^2 \left(L_k^2 - \frac{1}{2\alpha_{k-1}^2} \right) \leqslant \frac{1}{2}. \tag{35}$$

We can rewrite $x^{k+1} = \operatorname{prox}_{\alpha_k g}(x^k - \alpha_k \nabla f(x^k))$ as an implicit equation

$$x^{k+1} = x^k - \alpha_k (\nabla f(x^k) + \widetilde{\nabla} g(x^{k+1})), \tag{36}$$

where $\widetilde{\nabla} g(x^{k+1})$ is a certain subgradient of g at x^{k+1}, that is $\widetilde{\nabla} g(x^{k+1}) \in \partial g(x^{k+1})$. For this particular subgradient we will also use the notation

$$\widetilde{\nabla} F(x^k) = \nabla F(x^k) + \widetilde{\nabla} g(x^k).$$

First, we adapt our basic inequality (10) to the more general case. By prox-inequality, we have

$$\langle x^{k+1} - x^k + \alpha_k \nabla f(x^k), x - x^{k+1} \rangle \geqslant \alpha_k (g(x^{k+1}) - g(x)), \quad \forall x. \tag{37}$$

Then we set $x = x^*$ above and transform it into

$$\|x^{k+1} - x^*\|^2 + 2\alpha_k(g(x^{k+1}) - g(x^*)) \leqslant \|x^k - x^*\|^2 + 2\alpha_k\langle \nabla f(x^k), x^* - x^{k+1}\rangle - \|x^{k+1} - x^k\|^2. \tag{38}$$

This standard inequality is at the heart of the analysis of the proximal gradient method. To complete the full proof, or rather to get the final inequality, the classical analysis only requires applying one convexity inequality and one descent lemma to function f. Our analysis, however, will be different. The main nuisance is that in the k-th iteration the proximal map yields us $g(x^{k+1}) - g(x^*)$ term, while our adaptivity approach works with $f(x^k) - f(x^*)$, as we remember from before. Thus, our first obstacle is to understand how to combine these two terms.

First, we estimate the term $\langle \nabla f(x^k), x^* - x^{k+1}\rangle$ in the RHS of (38). We have

$$\begin{aligned}
\langle \nabla f(x^k), x^* - x^{k+1}\rangle &= \langle \nabla f(x^k), x^* - x^k\rangle + \langle \nabla f(x^k), x^k - x^{k+1}\rangle \\
&= \langle \nabla f(x^k), x^* - x^k\rangle + \langle \nabla f(x^k) + \widetilde{\nabla} g(x^k), x^k - x^{k+1}\rangle + \langle \widetilde{\nabla} g(x^k), x^{k+1} - x^k\rangle \\
&\leqslant f(x^*) - f(x^k) + \langle \nabla f(x^k) + \widetilde{\nabla} g(x^k), x^k - x^{k+1}\rangle + g(x^{k+1}) - g(x^k),
\end{aligned}$$

where in the last inequality we used separately convexity of f and g. Applying this inequality in (38) yields

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 + 2\alpha_k(F(x^k) - F(x^*)) \\
&\leqslant \|x^k - x^*\|^2 + 2\alpha_k\langle \nabla f(x^k) + \widetilde{\nabla} g(x^k), x^k - x^{k+1}\rangle - \|x^{k+1} - x^k\|^2 \\
&\leqslant \|x^k - x^*\|^2 + \alpha_k^2\|\nabla f(x^k) + \widetilde{\nabla} g(x^k)\|^2.
\end{aligned} \tag{39}$$

As we see, the final inequality is very much in the spirit of (10).

Lemma 12 (Compare to Lemma 1)**.** For iterates (x^k) with arbitrary stepsizes, it holds

$$\langle \nabla f(x^k) + \widetilde{\nabla} g(x^k), \nabla f(x^{k-1}) + \widetilde{\nabla} g(x^k)\rangle \leqslant \|\nabla f(x^{k-1}) + \widetilde{\nabla} g(x^k)\|^2. \tag{40}$$

Proof. As before, this is just monotonicity of ∇f in disguise:

$$\begin{aligned}
&\|\nabla f(x^{k-1}) + \widetilde{\nabla} g(x^k)\|^2 - \langle \nabla f(x^k) + \widetilde{\nabla} g(x^k), \nabla f(x^{k-1}) + \widetilde{\nabla} g(x^k)\rangle \\
&\quad = \langle \nabla f(x^{k-1}) - \nabla f(x^k), \nabla f(x^{k-1}) + \widetilde{\nabla} g(x^k)\rangle \\
&\quad = \frac{1}{\alpha_{k-1}}\langle \nabla f(x^{k-1}) - \nabla f(x^k), x^{k-1} - x^k\rangle \geqslant 0.
\end{aligned}$$

■

The next lemma is special for the composite case. Although it looks like this fact should be known in the literature, we were not able to identify it.

Lemma 13. For iterates (x^k) of the proximal gradient method with arbitrary stepsizes, it holds

$$\|\nabla f(x^k) + \widetilde{\nabla} g(x^{k+1})\| \leq \|\nabla f(x^k) + \widetilde{\nabla} g(x^k)\|. \tag{41}$$

Proof. This time it is just a monotonicity of ∂g in disguise:

$$\begin{aligned}
\|\nabla f(x^k) + \widetilde{\nabla} g(x^k)\|^2 &= \|\nabla f(x^k) + \widetilde{\nabla} g(x^{k+1}) + \widetilde{\nabla} g(x^k) - \widetilde{\nabla} g(x^{k+1})\|^2 \\
&\overset{(36)}{=} \left\|\frac{1}{\alpha_k}(x^k - x^{k+1}) + \widetilde{\nabla} g(x^k) - \widetilde{\nabla} g(x^{k+1})\right\|^2 \\
&= \frac{1}{\alpha_k^2}\|x^k - x^{k+1}\|^2 + \frac{2}{\alpha_k}\langle x^k - x^{k+1}, \widetilde{\nabla} g(x^k) - \widetilde{\nabla} g(x^{k+1})\rangle + \|\widetilde{\nabla} g(x^k) - \widetilde{\nabla} g(x^{k+1})\|^2 \\
&\geqslant \frac{1}{\alpha_k^2}\|x^k - x^{k+1}\|^2 + \frac{2}{\alpha_k}\langle x^k - x^{k+1}, \widetilde{\nabla} g(x^k) - \widetilde{\nabla} g(x^{k+1})\rangle \\
&\geqslant \|\nabla f(x^k) + \widetilde{\nabla} g(x^{k+1})\|^2,
\end{aligned}$$

where the last inequality follows from monotonicity of ∂g and (36). ■

https://doi.org/10.52202/079017-3193

In Section 3 we estimated $\|x^{k+1}-x^k\|^2 = \alpha_k^2\|\nabla f(x^k)\|^2$. This time, $\|x^{k+1}-x^k\|^2$ and $\alpha_k^2\|\widetilde{\nabla}F(x^k)\|^2$ are different and it is the latter term that matters to us.

Lemma 14 (Compare to Lemma 4)**.** For iterates (x^k) of Algorithm 3 it holds

$$\alpha_k^2\|\widetilde{\nabla}F(x^k)\|^2 \leqslant \frac{\alpha_{k-1}^2}{2}\|\widetilde{\nabla}F(x^{k-1})\|^2 + \frac{3}{2}\alpha_k\theta_k(F(x^{k-1})-F(x^k)).$$

Proof. The main idea of the proof is exactly the same as in Lemma 4. However, the presence of $\widetilde{\nabla}g(x^k)$ make it slightly more cumbersome. The previous two lemmata are instrumental on our way. We have

$$\begin{aligned}
&\alpha_k^2\|\nabla f(x^k)+\widetilde{\nabla}g(x^k)\|^2 = \alpha_k^2\|\nabla f(x^k)-\nabla f(x^{k-1})\|^2 - \alpha_k^2\|\nabla f(x^{k-1})+\widetilde{\nabla}g(x^k)\|^2\\
&\qquad + 2\alpha_k^2\langle\nabla f(x^k)+\widetilde{\nabla}g(x^k), \nabla f(x^{k-1})+\widetilde{\nabla}g(x^k)\rangle\\
&= \alpha_k^2L_k^2\|x^k-x^{k-1}\|^2 - \frac{\alpha_k^2}{2\alpha_{k-1}^2}\|x^k-x^{k-1}\|^2 - \frac{\alpha_k^2}{2}\|\nabla f(x^{k-1})+\widetilde{\nabla}g(x^k)\|^2\\
&\qquad + 2\alpha_k^2\langle\nabla f(x^k)+\widetilde{\nabla}g(x^k), \nabla f(x^{k-1})+\widetilde{\nabla}g(x^k)\rangle\\
&\overset{(40)}{\leqslant} \alpha_k^2\left(L_k^2 - \frac{1}{2\alpha_{k-1}^2}\right)\|x^k-x^{k-1}\|^2 + \frac{3\alpha_k^2}{2}\langle\nabla f(x^k)+\widetilde{\nabla}g(x^k), \nabla f(x^{k-1})+\widetilde{\nabla}g(x^k)\rangle\\
&\overset{(35)\&(36)}{\leqslant} \frac{1}{2}\|x^k-x^{k-1}\|^2 + \frac{3}{2}\alpha_k\theta_k\langle\nabla f(x^k)+\widetilde{\nabla}g(x^k), x^{k-1}-x^k\rangle\\
&\overset{(36)}{=} \frac{\alpha_{k-1}^2}{2}\|\nabla f(x^{k-1})+\widetilde{\nabla}g(x^k)\|^2 + \frac{3}{2}\alpha_k\theta_k\langle\widetilde{\nabla}F(x^k), x^{k-1}-x^k\rangle\\
&\overset{(41)}{\leqslant} \frac{\alpha_{k-1}^2}{2}\|\nabla f(x^{k-1})+\widetilde{\nabla}g(x^{k-1})\|^2 + \frac{3}{2}\alpha_k\theta_k\langle\widetilde{\nabla}F(x^k), x^{k-1}-x^k\rangle.
\end{aligned}$$

Convexity of F completes the proof. ■

Lemma 15 (Compare to Lemma 5)**.** For iterates (x^k) of Algorithm 3 and any solution x^* it holds

$$\begin{aligned}
&\|x^{k+1}-x^*\|^2 + \alpha_k^2\|\widetilde{\nabla}F(x^k)\|^2 + \alpha_k(2+3\theta_k)(F(x^k)-F_*)\\
&\leqslant \|x^k-x^*\|^2 + \alpha_{k-1}^2\|\widetilde{\nabla}F(x^{k-1})\|^2 + 3\alpha_k\theta_k(F(x^{k-1})-F_*).
\end{aligned} \tag{42}$$

Proof. The same as in Lemma 5. ■

Recall that we define R as

$$R^2 = \|x^0-x^*\|^2 + 2\alpha_0^2\|\widetilde{\nabla}F(x^0)\|^2 + \alpha_0(F(x^0)-F_*). \tag{43}$$

Lemma 16. The sequence (x^k) is bounded. In particular, for any solution x^* of (20) we have $x^k \in B(x^*, R)$.

Proof. The same as in Lemma 2. We use (42) to telescope until $k=1$ and then apply (39) with $k=0$ to bound $\|x^1-x^*\|^2$. ■

Proof of Theorem 3. The proof of inequalities (22) and (23) is almost identical to the one in Theorem 2. The proof of convergence of (x^k) to a solution is, however, more nuanced. The nontrivial part is to show that all limit points of (x^k) are solutions. While on the surface, it should be no harder than before, the fact that $\lim_{k\to+\infty}\alpha_k$ can be $+\infty$ complicates things a bit.

Let x^* be a solution of (20). By L-smoothness of f over $B(x^*, R)$, we have

$$f(x^*) - f(x^k) \geqslant \langle\nabla f(x^k), x^*-x^k\rangle + \frac{1}{2L}\|\nabla f(x^k)-\nabla f(x^*)\|^2.$$

Using this improved bound, similarly to how it was done in (32), we get

$$\begin{aligned}
&\|x^{k+1}-x^*\|^2 + \alpha_k^2\|\widetilde{\nabla}F(x^k)\|^2 + \alpha_k(2+3\theta_k)(F(x^k)-F_*) + \frac{\alpha_k}{L}\|\nabla f(x^k)-\nabla f(x^*)\|^2\\
&\leqslant \|x^k-x^*\|^2 + \alpha_{k-1}^2\|\widetilde{\nabla}F(x^{k-1})\|^2 + 3\alpha_k\theta_k(F(x^{k-1})-F_*).
\end{aligned} \tag{44}$$

By telescoping this inequality as before, we can now additionally infer that

$$\sum_{k=1}^{\infty} \alpha_k \|\nabla f(x^k) - \nabla f(x^*)\|^2 < +\infty \tag{45}$$

and thus, $\|\nabla f(x^k) - \nabla f(x^*)\| \to 0$. Specifically, this implies $\nabla f(x^k) - \nabla f(x^{k-1}) \to 0$ as $k \to \infty$.

We want to prove that all limit points of (x^k) are solutions. To this end, we will use prox-inequality (37) rewritten as

$$\frac{1}{\alpha_k}\langle x^{k+1} - x^k, x - x^{k+1}\rangle + \langle \nabla f(x^k), x - x^{k+1}\rangle \geqslant g(x^{k+1}) - g(x), \forall x \tag{46}$$

which in turn, by convexity of f, leads to

$$\frac{1}{\alpha_k}\langle x^{k+1} - x^k, x - x^{k+1}\rangle + \langle \nabla f(x^k) - \nabla f(x^{k+1}), x - x^{k+1}\rangle \geqslant F(x^{k+1}) - F(x). \tag{47}$$

The left-hand side has two terms, and the second term evidently tends to 0 as $\nabla f(x^{k+1}) - \nabla f(x^k) \to 0$. If we can show the same for the first term, it will imply that all limit points of (x^k) are solutions.

Consider (46) again, but this time we set $x = x^k$. This yields

$$-\frac{1}{\alpha_k}\|x^{k+1} - x^k\|^2 + \langle \nabla f(x^k), x^k - x^{k+1}\rangle \geqslant g(x^{k+1}) - g(x^k).$$

We manipulate the inequality above as follows

$$\begin{aligned}
\frac{1}{\alpha_k}\|x^{k+1} - x^k\|^2 &\leqslant \langle \nabla f(x^k), x^k - x^{k+1}\rangle + g(x^k) - g(x^{k+1}) \\
&= \langle \nabla f(x^k) - \nabla f(x^*), x^k - x^{k+1}\rangle + \underbrace{\langle \nabla f(x^*), x^k - x^{k+1}\rangle + g(x^k) - g(x^{k+1})}_{\delta_k} \\
&\leqslant \frac{\alpha_k}{2}\|\nabla f(x^k) - \nabla f(x^*)\|^2 + \frac{1}{2\alpha_k}\|x^{k+1} - x^k\|^2 + \delta_k,
\end{aligned}$$

where in the last inequality we applied Cauchy-Schwarz and Young's inequalities. From this we deduce that

$$\frac{1}{\alpha_k}\|x^{k+1} - x^k\|^2 \leqslant \alpha_k \|\nabla f(x^k) - \nabla f(x^*)\|^2 + 2\delta_k.$$

Note that the sequence $\left(\alpha_k \|\nabla f(x^k) - \nabla f(x^*)\|^2\right)$ is summable by (45). Also, the sequence (δ_k) is summable, since (x^k) is bounded and $g(x^k)$ is lower-bounded: $g(x^k) \geqslant F_* - f(x^k) > -\infty$ for all k. Hence, $\sum_k \frac{1}{\alpha_k}\|x^{k+1} - x^k\|^2 < +\infty$ and, thus, $\frac{1}{\alpha_k}\|x^{k+1} - x^k\|^2 \to 0$ as $k \to +\infty$. Given that (α_k) is separated from zero, it immediately follows that $\frac{1}{\alpha_k}\|x^{k+1} - x^k\| \to 0$ as well.

Therefore, we have proved that all limit points of (x^k) are solutions. The proof of convergence of the whole sequence (x^k) runs as before in Theorem 2. ■

Remark 6. We didn't derive a linear convergence of the adaptive proximal gradient, when F is strongly convex. We only mention that it is quite straightforward and goes along the same lines as the original AdGD in [MM20, Theorem 2] in the strongly convex regime.

D Additional experiments

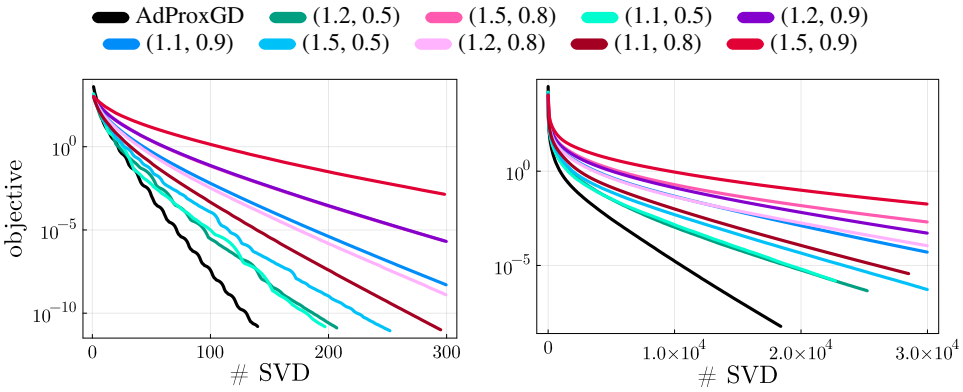
Low-rank matrix completion. We consider a famous low-rank matrix completion problem in the form

$$\min_{X \in \mathbb{R}^{n\times n}} \frac{1}{2}\|P_\Omega(X - A)\|_F^2 \quad \text{subject to} \quad \|X\|_* \leqslant r, \tag{48}$$

where Ω is a subset of indices (i, j) and r is the supposed maximum rank. To project onto the spectral ball $\mathcal{C} = \{X : \|X\|_* \leqslant r\}$, computing the singular value decomposition (SVD) is required, making it the most computationally expensive operation in this setting.

We created matrix A by multiplying matrices U and $V^\top$, where U and V are n-by-r matrices with entries sampled from a normal distribution. The subset Ω was randomly chosen as a fraction of $\frac{1}{5}n^2$ entries from $[n] \times [n]$. The obtained results are depicted in Figure 2, where we solely compared the number of computed SVDs. For two scenarios we generated, the proposed method was always faster than any of the linesearch versions.

https://doi.org/10.52202/079017-3193



(a) $n = 100$, $r = 20$ (b) $n = 200$, $r = 20$

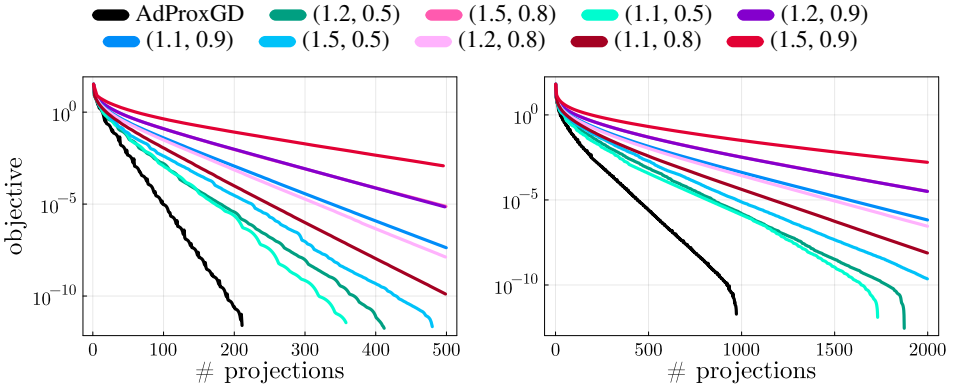
Figure 2: Low-rank matrix completion, problem (48)

Minimal length piecewise-linear curve subject to linear constraints. We consider [BV04, Example 10.4], where we want to minimize the length of a piecewise-linear curve passing through n points in $\mathbb{R}^2$ with coordinates $(1, x_1), \dots, (n, x_n)$ while satisfying linear constraints $Ax = b$, where $x = (x_1, \dots, x_n)$. Given $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, this can be modeled as

$$\min_{x \in \mathbb{R}^n} (1 + x_1^2)^{1/2} + \sum_{i=1}^{n-1} (1 + (x_{i+1} - x_i)^2)^{1/2} \quad \text{subject to} \quad Ax = b. \tag{49}$$

While applying the proximal gradient method, the most computationally expensive operation is computing the projection onto $\mathcal{C} = \{x \colon Ax = b\}$. Assuming that A is full rank with $m \leqslant n$, this projection can be computed as $P_{\mathcal{C}} z = z - A^\top (AA^\top)^{-1}(Az - b)$.

In comparison, we focused solely on the number of computed projections. We generated a random m-by-n matrix A and random vector w with entries sampled from a normal distribution and set $b = Aw$. In Figure 3, we can see again that the proposed method converged faster than any of the linesearch versions.



(a) $m = 50$, $n = 200$ (b) $m = 50$, $n = 500$

Figure 3: Minimal length piecewise-linear curve, problem (49)

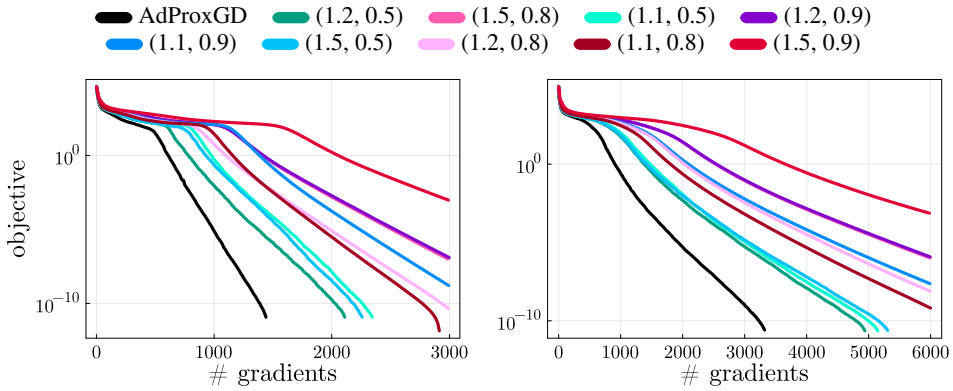
Nonnegative matrix factorization. We want to solve the matrix factorization problem subject to nonnegative constraints:

$$\min_{U, V \in \mathbb{R}_+^{n \times r}} f(U, V) = \frac{1}{2} \|UV^\top - A\|_F^2, \tag{50}$$

where A is a given n-by-n low-rank matrix. Although nonconvex, this problem is famously well-tackled by first-order methods. In each iteration, the gradient $\nabla f(x)$ involves 3 matrix-matrix

multiplications, whereas evaluating the objective $f(x)$ only requires 1. Note that for the last iteration of the linesearch, the computed matrix product can be reused to compute the gradient for the next iteration.

We created matrix A by multiplying matrices B and $C^\top$, where B and C are n-by-r matrices with entries sampled from a normal distribution. Negative entries in both matrices B and C were then set to zero. The results are presented in Figure 4, where the number of gradients roughly means the number of 3 matrix-matrix multiplications. In both cases we generated, the proposed method converged faster than any of the linesearch versions.



(a) $n = 100$, $r = 20$ (b) $n = 100$, $r = 30$

Figure 4: Nonnegative matrix factorization, problem (50)

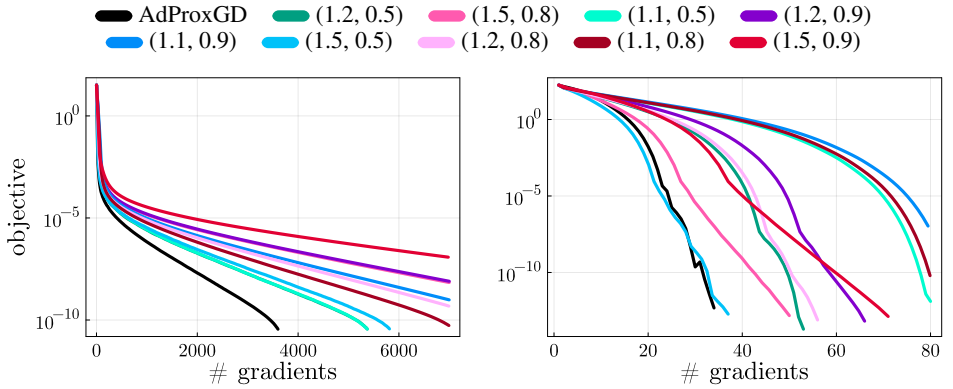
Dual of the entropy maximization problem. Consider the entropy maximization problem subject to linear constraints

$$\min \sum_{i=1}^{n} x_i \log x_i \quad \text{subject to } Ax \leqslant b, \quad \sum_{i=1}^{n} x_i = 1, \text{ and } x_i > 0, \tag{51}$$

where $A \in \mathbb{R}^{m \times n}$. Its dual problem is given by

$$\min_{\lambda \in \mathbb{R}^m_+, \mu \in \mathbb{R}} e^{-\mu - 1} \sum_{i=1}^{n} e^{-a_i^\top \lambda} + \langle b, \lambda \rangle + \mu, \tag{52}$$

where $a_i \in \mathbb{R}^m$ is the i-th column of A (the derivation is provided in [BV04, Chapter 5.1.6]).



(a) $m = 500$, $n = 100$ (b) $m = 100$, $n = 500$

Figure 5: Dual of the entropy maximization, problem (52)

It is the latter problem (52) that we solved. We generated m-by-n matrix A with entries sampled from a normal distribution. Then we generated a random $w \in \mathbb{R}^n$ from the unit simplex and set

https://doi.org/10.52202/079017-3193

$b = Aw$. Each gradient requires two matrix-vector multiplications, while the objective evaluation requires only one (and, as before, the last one can be reused for the next gradient). The results are presented in Figure 5, where the number of gradients roughly means the number of 2 matrix-vector multiplications. In the first scenario, the proposed method is faster than all the linesearch versions, while in the second one, only one version of linesearch was comparable in performance.

Conclusion. Based on our preliminary experiments, it is evident that AdProxGD indeed performs better. To our surprise, a few specific pairs (r, s) consistently outperform the rest among ProxGD with linesearch. We are not aware of any theoretical finding that would confirm this evidence. Also, from a numerical point of view, the linesearch implementation is not always robust. In particular, when we are already close to a solution, the linesearch condition can sometimes fail because the numbers it operates on are all very small.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction describes our contributions in details.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in the Literature and Discussion section, as well as in other various parts of the paper.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

https://doi.org/10.52202/079017-3193

Justification: Each statement has a proof either in the main text or in Appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe in details how a random data was generated for the experiments. The code is publicly available.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code is publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: It is described in details which parameters were used.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: There are no error bars for the experiments. The paper is mostly of theoretical interest.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

https://doi.org/10.52202/079017-3193

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: This is not important for our experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: This is a theoretical paper.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This is a theoretical work that does not have any foreseeable societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This is a theoretical paper.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: All experiments were randomly generated.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

 https://doi.org/10.52202/079017-3193

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: There are no new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not contain a study involving human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not contain a study involving human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.