Catalyst Acceleration for Gradient-Based Non-Convex Optimization

Courtney Paquette, Hongzhou Lin, Dmitriy Drusvyatskiy, Julien Mairal, Zaid Harchaoui

Catalyst Acceleration for Gradient-Based Non-Convex Optimization*

Courtney Paquette
University of Washington
yumiko88@uw.edu

Hongzhou Lin
Inria
hongzhou.lin@inria.fr

Dmitriy Drusvyatskiy
University of Washington
ddrusv@uw.edu

Julien Mairal
Inria
julien.mairal@inria.fr

Zaid Harchaoui
University of Washington
zaid@uw.edu

June 9, 2017

Abstract

We introduce a generic scheme to solve nonconvex optimization problems using gradient-based algorithms originally designed for minimizing convex functions. When the objective is convex, the proposed approach enjoys the same properties as the Catalyst approach of Lin et al. [22]. When the objective is nonconvex, it achieves the best known convergence rate to stationary points for first-order methods. Specifically, the proposed algorithm does not require knowledge about the convexity of the objective; yet, it obtains an overall worst-case efficiency of $\tilde{O}(\varepsilon^{-2})$ and, if the function is convex, the complexity reduces to the near-optimal rate $\tilde{O}(\varepsilon^{-2/3})$. We conclude the paper by showing promising experimental results obtained by applying the proposed approach to SVRG and SAGA for sparse matrix factorization and for learning neural networks.

1 Introduction

We consider optimization problems of the form

$$\min_{x\in\mathbb{R}^p} \{f(x) := f_0(x) + \psi(x)\}\ , \quad \text{where } f_0(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x)\ . \tag{1}$$

Here, each function $f_i\colon \mathbb{R}^p \to \mathbb{R}$ is smooth, the regularization $\psi\colon \mathbb{R}^p \to \overline{\mathbb{R}}$ may be nonsmooth, and $\overline{\mathbb{R}} := \mathbb{R}\cup\{\infty\}$. By considering extended-real-valued functions, this composite setting also encompasses constrained minimization by letting ψ be the indicator function of the constraints on x. Minimization of regularized empirical risk objectives of form (1) is central in machine learning. Whereas a significant amount of work has been devoted to this composite setting for convex problems, leading in particular to fast incremental algorithms [see, *e.g.*, 12, 20, 23, 35, 37, 38], the question of minimizing efficiently (1) when the functions f_i and ψ may be nonconvex is still largely open today.

Yet, nonconvex problems in machine learning are of high interest. For instance, the variable x may represent the parameters of a neural network, where each term $f_i(x)$ measures the fit between x and a data point indexed by i, or (1) may correspond to a nonconvex matrix factorization problem (see Section 6). Besides, even when the data-fitting functions f_i are convex, it is also typical to consider nonconvex regularization

*C. Paquette and D. Drusvyatskiy were supported by the AFOSR YIP Award FA9550-15-1-0237. H. Lin and J. Mairal were supported by the ERC grant SOLARIS (number 714381) and a grant from ANR (MACARON project ANR-14-CE23-0003-01). Z. Harchaoui was supported by the program "Learning in Machines and Brains" of CIFAR.

functions ψ, for example for feature selection in signal processing [18]. In this work, we address two questions from nonconvex optimization:

1. How to apply a method for convex optimization to a nonconvex problem?
2. How to design an algorithm which does not need to know whether the objective function is convex while obtaining the optimal convergence guarantee if the function is convex?

Several pioneering works attempted to transfer ideas from the convex world to the nonconvex one, see, *e.g.*, [15, 16]. Our paper has a similar goal and studies the extension of Nesterov's acceleration for convex problems [26] to nonconvex composite ones. Unfortunately, the concept of acceleration for nonconvex problems is unclear from a worst-case complexity point of view: gradient descent requires $O(\varepsilon^{-2})$ iterations to guarantee a gradient norm smaller than ε [10, 9]. Under a stronger assumption that the objective function is C^2-smooth, state-of-the-art methods [*e.g.*, 7] achieve a marginal gain with complexity $O(\varepsilon^{-7/4}\log(1/\varepsilon))$, and do not appear to generalize to composite or finite-sum settings. For this reason, our work fits within a broader stream of recent research on methods that *do not perform worse than gradient descent in the nonconvex case* (in terms of worst-case complexity), while *automatically accelerating for minimizing convex functions*. The hope when applying such methods to nonconvex problems is to see acceleration in practice, by heuristically exploiting convexity that is "hidden" in the objective (for instance, local convexity near the optimum, or convexity along the trajectory of iterates).

The main contribution of this paper is a *generic* meta-algorithm, dubbed 4WD-Catalyst, which is able to use a *gradient-based* optimization method $\mathcal{M}$, originally designed for convex problems, and turn it into an accelerated scheme that also applies to nonconvex objective functions. The proposed 4WD-Catalyst can be seen as a **4-W**heel-**D**rive extension of Catalyst [22] to all optimization "terrains" (convex and nonconvex), while Catalyst was originally proposed for convex optimization. Specifically, without knowing whether the objective function is convex or not, our algorithm may take a method $\mathcal{M}$ designed for convex optimization problems with the same structure as (1), *e.g.*, SAGA [12], SVRG [38], and apply $\mathcal{M}$ to a sequence of subproblems such that it asymptotically provides a stationary point of the nonconvex objective. Overall, the number of iterations of $\mathcal{M}$ to obtain a gradient norm smaller than ε is $\tilde{O}(\varepsilon^{-2})$ in the worst case, while automatically reducing to $\tilde{O}(\varepsilon^{-2/3})$ if the function is convex.[1]

Related work. Inspired by Nesterov's acceleration method for convex optimization [27], the first accelerated method performing universally well for nonconvex and convex problems was introduced in [15]. Specifically, the work [15] addresses composite problems such as (1) with $n = 1$, and, provided the iterates are bounded, it performs no worse than gradient descent on nonconvex instances with complexity $O(\varepsilon^{-2})$ on the gradient norm. When the problem is convex, it accelerates with complexity $O(\varepsilon^{-2/3})$. Extensions to accelerated Gauss-Newton type methods were also recently developed in [13]. In a follow-up work [16], a new scheme is proposed, which monotonically interlaces proximal gradient descent steps and Nesterov's extrapolation; thereby achieving similar guarantees as [15] but without the need to assume the iterates to be bounded. Extensions when the gradient of ψ is only Hölder continuous can also be devised.

In [21], a similar strategy is proposed, focusing instead on convergence guarantees under the so-called Kurdyka-Łojasiewicz inequality—a property corresponding to polynomial-like growth of the function, as shown by [5]. Our scheme is in the same spirit as these previous papers, since it monotonically interlaces proximal-point steps (instead of proximal-gradient as in [16]) and extrapolation/acceleration steps. A fundamental difference is that our method is generic and accommodates inexact computations, since we allow the subproblems to be approximately solved by any method we wish to accelerate.

By considering C^2-smooth nonconvex objective functions f with Lipschitz continuous gradient ∇f and Hessian $\nabla^2 f$, Carmon et al. [7] propose an algorithm with complexity $O(\varepsilon^{-7/4}\log(1/\varepsilon))$, based on iteratively solving convex subproblems closely related to the original problem. It is not clear if the complexity of their algorithm improves in the convex setting. Note also that the algorithm proposed in [7] is inherently for

[1] In this section, the notation $\tilde{O}$ only displays the polynomial dependency with respect to ε for the clarity of exposition.

C^2-smooth minimization and requires exact gradient evaluations. This implies that the scheme does not allow incorporating nonsmooth regularizers and can not exploit finite sum structure.

Finally, a stochastic method related to SVRG [19] for minimizing large sums while automatically adapting to the weak convexity constant of the objective function is proposed in [1]. When the weak convexity constant is small (*i.e.*, the function is nearly convex), the proposed method enjoys an improved efficiency estimate. This algorithm, however, does not automatically accelerate for convex problems, in the sense that the overall rate is slower than $O(\varepsilon^{-2/3})$ in terms of target accuracy ε on the gradient norm.

Organization of the paper. Section 2 presents mathematical tools for non-convex and non-smooth analysis, which are used throughout the paper. In Sections 3 and 4, we introduce the main algorithm and important extensions, respectively. Finally, we present experimental results on matrix factorization and training of neural networks in Section 6.

2 Tools for nonconvex and nonsmooth optimization

Convergence results for nonsmooth optimization typically rely on the concept of subdifferential, which does not admit a unique definition in a nonconvex context [6]. In this paper, we circumvent this issue by focusing on a broad class of nonconvex functions known as weakly convex or lower C^2 functions, for which all these constructions coincide. Weakly convex functions cover most of the interesting cases of interest in machine learning and resemble convex functions in many aspects. In this section, we formally introduce them and discuss their subdifferential properties.

Definition 2.1 (Weak convexity)**.** A function $f\colon \mathbb{R}^p \to \overline{\mathbb{R}}$ is $\rho-$*weakly convex* if for any points $x, y \in \mathbb{R}^p$ and $\lambda \in [0,1]$, the approximate secant inequality holds:

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y) + \rho\lambda(1-\lambda)\left\|x-y\right\|^2 .$$

Notice that ρ-weak convexity with $\rho = 0$ is exactly the definition of a convex function. An elementary algebraic manipulation shows that f is ρ-weakly convex if and only if the function $x \mapsto f(x) + \frac{\rho}{2}\left\|x\right\|^2$ is convex. In particular, a C^1-smooth function f is ρ-weakly convex if the gradient ∇f is ρ-Lipschitz, while a C^2-smooth function f is ρ-weakly convex if and only if $\nabla^2 f(x) \succeq -\rho I$ for all x. This closely resembles an equivalent condition for C^2-smooth and μ-strongly convex functions, namely $\nabla^2 f(x) \succeq \mu I$ with $\mu > 0$.

Useful characterizations of ρ-weakly convex functions rely on differential properties. Since the functions we consider in the paper are nonsmooth, we use a generalized derivative construction. We mostly follow the standard monograph on the subject by Rockafellar and Wets [34].

Definition 2.2 (Subdifferential)**.** Consider a function $f\colon \mathbb{R}^p \to \overline{\mathbb{R}}$ and a point x with $f(x)$ finite. The *subdifferential* of f at x is the set

$$\partial f(x) := \{v \in \mathbb{R}^p : f(y) \geq f(x) + v^T(y-x) + o(\|y-x\|) \ \ \forall y \in \mathbb{R}^p\}.$$

Thus, a vector v lies in $\partial f(x)$ whenever the linear function $y \mapsto f(x) + v^T(y-x)$ is a lower-model of f, up to first-order around x. In particular, the subdifferential $\partial f(x)$ of a differentiable function f is the singleton $\{\nabla f(x)\}$, while for a convex function f it coincides with the subdifferential in the sense of convex analysis [see 34, Exercise 8.8]. It is useful to keep in mind that the sum rule, $\partial(f+g)(x) = \partial f(x) + \nabla g(x)$, holds for any differentiable function g.

We are interested in deriving complexity bounds on the number of iterations required by a method $\mathcal{M}$ to guarantee

$$\operatorname{dist}\big(0, \partial f(x)\big) \leq \varepsilon .$$

Recall when $\varepsilon = 0$, we are at a stationary point and satisfy first-order optimality conditions. In our convergence analysis, we will also use the following differential characterization of ρ-weakly convex functions, which generalize classical properties of convex functions. A proof follows directly from Theorem 12.17 of [34] by taking into account that f is ρ-weakly convex if and only if $f + \frac{\rho}{2}\|\cdot\|^2$ is convex.

Theorem 2.3 (Differential characterization of ρ-weakly convex functions)**.**
For any lower-semicontinuous function $f\colon \mathbb{R}^p \to \overline{\mathbb{R}}$, the following properties are equivalent:

1. *f is ρ-weakly convex.*
2. **(subgradient inequality).** *For all x, y in $\mathbb{R}^p$ and v in $\partial f(x)$, we have*
$$f(y) \geq f(x) + v^T(y-x) - \frac{\rho}{2}\|y-x\|^2.$$
3. **(hypo-monotonicity).** *For all x, y in $\mathbb{R}^p$, v in $\partial f(x)$, and w in $\partial f(y)$,*
$$(v-w)^T(x-y) \geq -\rho\|x-y\|^2.$$

Weakly convex functions have appeared in a wide variety of contexts, and under different names. Some notable examples are globally lower-C^2 [33], prox-regular [31], proximally smooth functions [11], and those functions whose epigraph has positive reach [14].

3 The Basic 4WD-Catalyst algorithm for non-convex optimization

We now present a generic scheme (Algorithm 1) for applying a convex optimization method to minimize

$$\min_{x\in\mathbb{R}^p} \; f(x), \tag{2}$$

where f is only ρ-weakly convex. Our goal is to develop a unified framework that automatically accelerates in convex settings. Consequently, the scheme must be agnostic to the constant ρ.

3.1 Basic 4WD-Catalyst : a meta algorithm

At the center of our meta algorithm (Algorithm 1) are two sequences of subproblems obtained by adding simple quadratics to f. The proposed approach extends the Catalyst acceleration of [22] and comes with a simplified convergence analysis. We next describe in detail each step of the scheme.

Two-step subproblems. The proposed acceleration scheme builds two main sequences of iterates $(\bar{x}_k)_k$ and $(\tilde{x}_k)_k$, obtained from approximately solving two subproblems. These subproblems are simple quadratic perturbations of the original problem f having the form:

$$\min_x \left\{ f_\kappa(x; y) := f(x) + \frac{\kappa}{2}\|x-y\|^2 \right\}.$$

Here, κ is a regularization parameter and y is called the *prox-center*. By adding the quadratic, we make the problem more "convex": when f is non convex, with a large enough κ, the subproblem will be convex; when f is convex, we improve the conditioning of the problem.

At the k-th iteration, given a previous iterate x_{k-1} and the extrapolation term v_{k-1}, we construct the two following subproblems.

1. **Proximal point step.** We first perform an inexact proximal point step with prox-center x_{k-1}:
$$\bar{x}_k \approx \operatorname*{argmin}_x f_\kappa(x; x_{k-1}) \quad \text{[Proximal-point step]}$$
2. **Accelerated proximal point step.** Then we build the next prox-center y_k as the convex combination
$$y_k = \alpha_k v_{k-1} + (1-\alpha_k)x_{k-1}. \tag{3}$$

Algorithm 1 Basic 4WD-Catalyst

input Fix a point $x_0 \in \operatorname{dom} f$, real numbers $\kappa > 0$, and an optimization method $\mathcal{M}$.

initialization: $\alpha_1 \equiv 1$, $v_0 \equiv x_0$.

repeat for $k = 1, 2, \ldots$

1. Choose $\bar{x}_k$ using $\mathcal{M}$ such that
$$\bar{x}_k \approx \operatorname*{argmin}_x f_\kappa(x; x_{k-1}) \tag{5}$$
where $\operatorname{dist}\big(0, \partial f_\kappa(\bar{x}_k; x_{k-1})\big) < \kappa \|\bar{x}_k - x_{k-1}\|$ and $f_\kappa(\bar{x}_k; x_{k-1}) \leq f_\kappa(x_{k-1}; x_{k-1})$.

2. Set
$$y_k = \alpha_k v_{k-1} + (1 - \alpha_k) x_{k-1}. \tag{6}$$

3. Choose $\tilde{x}_k$ using $\mathcal{M}$ such that
$$\tilde{x}_k \approx \operatorname*{argmin}_x f_\kappa(x; y_k) \qquad \text{where} \qquad \operatorname{dist}\big(0, \partial f_\kappa(\tilde{x}_k; y_k)\big) < \frac{\kappa}{k+1} \|\tilde{x}_k - y_k\| . \tag{7}$$

4. Set
$$v_k = x_{k-1} + \frac{1}{\alpha_k}(\tilde{x}_k - x_{k-1}). \tag{8}$$

5. Pick $\alpha_{k+1} \in (0, 1)$ satisfying
$$\frac{1 - \alpha_{k+1}}{\alpha_{k+1}^2} = \frac{1}{\alpha_k^2}. \tag{9}$$

6. Choose x_k to be any point satisfying
$$f(x_k) \leq \min\{f(\bar{x}_k), f(\tilde{x}_k)\}. \tag{10}$$

until the stopping criterion $\operatorname{dist}\big(0, \partial f(\bar{x}_k)\big) < \varepsilon$

Next, we use y_k as a prox-center and update the next extrapolation term:

$$\begin{aligned} &\tilde{x}_k \approx \operatorname*{argmin}_x f_\kappa(x; y_k) && \text{[Accelerated proximal-point step]} \\ &v_k = x_{k-1} + \tfrac{1}{\alpha_k}(\tilde{x}_k - x_{k-1}) && \text{[Extrapolation]} \end{aligned} \tag{4}$$

where $\alpha_{k+1} \in (0, 1)$ is a sequence of coefficients satisfying $(1 - \alpha_{k+1})/\alpha_{k+1}^2 = 1/{\alpha_k}^2$. Essentially, the sequences $(\alpha_k)_k, (y_k)_k, (v_k)_k$ are built upon the extrapolation principles of Nesterov [27].

Picking the best. At the end of iteration k, we have at hand two iterates, resp. $\bar{x}_k$ and $\tilde{x}_k$. Following [15], we simply choose the best of the two in terms of their objective values, that is we choose x_k such that

$$f(x_k) \leq \min\{f(\bar{x}_k), f(\tilde{x}_k)\} .$$

The proposed scheme blends the two steps in a synergistic way, allowing us to recover the near-optimal rates of convergence in both worlds: convex and non-convex. Intuitively, when $\bar{x}_k$ is chosen, it means that Nesterov's extrapolation step "fails" to accelerate convergence.

Stopping criterion for the subproblems. In order to derive complexity bounds, it is important to properly define the stopping criterion for the proximal subproblems. When the subproblem is convex, a functional gap like $f_\kappa(z; x) - \inf_z f_\kappa(z; x)$ may be used as a control of the inexactness, as in [22]. Without

convexity, this criterion cannot be used since such quantities can not be easily bounded. In particular, first order methods seek points whose subgradient is small. Since small subgradients do not necessarily imply small function values in a non-convex setting, first order methods only test is for small subgradients. In contrast, in the convex setting, small subgradients imply small function values; thus a first order method in the convex setting can "test" for small function values. Hence, we cannot use a direct application of Catalyst [22] which uses the functional gap as a stopping criteria. Because we are working in the nonconvex setting, we include a stationarity stopping criteria.

We propose to use jointly the following two types of stopping criteria:

1. Descent condition: $f_\kappa(z; y) \leq f_\kappa(y; y)$;
2. Adaptive stationary condition: $\operatorname{dist}\big(0, \partial f_\kappa(z; y)\big) < \kappa \|z - y\|$.

Without the descent condition, the stationarity condition is insufficient for defining a good stopping criterion because of the existence of local maxima in nonconvex problems. In the nonconvex setting, local maxima and local minima satisfy the stationarity condition. The descent condition ensures the iterates generated by the algorithm always decrease the value of objective function f; thus ensuring we move away from local maxima. The second criterion, adaptive stationary condition, provides a flexible relative tolerance on termination of algorithm used for solving the subproblems; a detailed analysis is forthcoming.

In Basic 4WD-Catalyst , we use both the stationary condition and the descent condition as a stopping criteria to produce the point $\bar{x}$:

$$\operatorname{dist}\big(0, \partial f_\kappa(\bar{x}_k; x_{k-1})\big) < \kappa \|\bar{x}_k - x_{k-1}\| \text{ and } f_\kappa(\bar{x}_k; x_{k-1}) \leq f_\kappa(x_{k-1}; x_{k-1}). \tag{11}$$

For the point $\tilde{x}$, our "acceleration" point, we use a modified stationary condition:

$$\operatorname{dist}\big(0, \partial f_\kappa(\tilde{x}_k; y_k)\big) < \frac{\kappa}{k+1} \|\tilde{x}_k - y_k\|. \tag{12}$$

The $k+1$ factor guarantees Basic 4WD-Catalyst accelerates for the convex setting. To be precise, Equation (27) in the proofs of Theorem 3.1 and Theorem 3.2 uses the factor $k+1$ to ensure convergence. Note, we do not need the descent condition for $\tilde{x}$, as the functional decrease in $\bar{x}$ is enough to ensure the sequence $\{f(x_k)\}_{k\geq 1}$ is monotonically decreasing.

3.2 Convergence analysis.

We present here the theoretical properties of Algorithm 1. In this first stage, we do not take into account the complexity of solving the subproblems (5) and (7). For the next two theorems, we assume that the stopping criteria for the proximal subproblems are satisfied at each iteration of Algorithm 1.

Theorem 3.1 (Outer-loop complexity for Basic 4WD-Catalyst; non-convex case)**.** *For any $\kappa > 0$ and $N \geq 1$, the iterates generated by Algorithm 1 satisfy*

$$\min_{j=1,\ldots,N} \operatorname{dist}^2\big(0, \partial f(\bar{x}_j)\big) \leq \frac{8\kappa}{N}(f(x_0) - f^*).$$

It is important to notice that this convergence result is valid for any κ and does not require it to be larger than the weak convexity parameter. As long as the stopping criteria for the proximal subproblems are satisfied, the quantities $\operatorname{dist}(0, \partial f(\bar{x_j}))$ tend to zero. The proof is inspired by that of inexact proximal algorithms [4, 17, 22] and appears in Appendix B.

If the function f turns out to be convex, the scheme achieves a faster convergence rate both in function values and in stationarity:

Theorem 3.2 (Outer-loop complexity, convex case)**.** *If the function f is convex, then for any $\kappa > 0$ and $N \geq 1$, the iterates generated by Algorithm 1 satisfy*

$$f(x_N) - f(x^*) \leq \frac{4\kappa}{(N+1)^2} \|x^* - x_0\|^2, \tag{13}$$

and

$$\min_{j=1,\dots,2N} \operatorname{dist}^2\big(0, \partial f(\bar{x}_j)\big) \leq \frac{32\kappa^2}{N(N+1)^2} \left\|x^* - x_0\right\|^2,$$

where x^* is any minimizer of the function f.

The proof of Theorem 3.2 appears in Appendix B. This theorem establishes a rate of $O(N^{-2})$ for sub-optimality in function value and convergence in $O(N^{-3/2})$ for the minimal norm of subgradients. The first rate is optimal in terms of information-based complexity for the minimization of a convex composite function [27, 29]. The second can be improved to $O(N^{-2}\log(N))$ through a regularization technique, if one knew in advance that the function is convex and had an estimate on the distance of the initial point to an optimal solution [28].

Towards an automatically adaptive algorithm. So far, our analysis has not taken into account the cost of obtaining the iterates $\bar{x}_j$ and $\tilde{x}_j$ by the algorithm $\mathcal{M}$. We emphasize again that the two results above do not require any assumption on κ, which leaves us a degree of freedom. In order to develop the global complexity, we need to evaluate the total number of iterations performed by $\mathcal{M}$ throughout the process. Clearly, this complexity heavily depends on the choice of κ, since it controls the magnitude of regularization we add to improve the convexity of the subproblem. This is the point where a careful analysis is needed, because our algorithm must adapt to ρ without knowing it in advance. The next section is entirely dedicated to this issue. In particular, we will explain how to automatically adapt the parameter κ (Algorithm 2).

4 The 4WD-Catalyst algorithm

In this section, we work towards understanding the global efficiency of Algorithm 1, which automatically adapts to the weak convexity parameter. For this, we must take into account the cost of approximately solving the proximal subproblems to the desired stopping criteria. We expect that once the subproblem becomes strongly convex, the given optimization method $\mathcal{M}$ can solve it efficiently. For this reason, we first focus on the computational cost for solving the sub-problems, before introducing a new algorithm with known worst-case complexity.

4.1 Solving the sub-problems efficiently

When κ is large enough, the subproblems become strongly convex; thus globally solvable. Henceforth, we will assume that $\mathcal{M}$ satisfies the following natural linear convergence assumption.

Linear convergence of $\mathcal{M}$ for strongly-convex problems. We assume that for any $\kappa > \rho$, there exist $A_\kappa \geq 0$ and $\tau_\kappa \in (0,1)$ so that the following hold:

1. For any prox-center $y \in \mathbb{R}^p$ and initial $z_0 \in \mathbb{R}^p$ the iterates $\{z_t\}_{t\geq 1}$ generated by $\mathcal{M}$ on the problem $\min_z f_\kappa(z;y)$ satisfy
$$\operatorname{dist}^2(0, \partial f_\kappa(z_t;y)) \leq A_\kappa(1-\tau_\kappa)^t(f_\kappa(z_0;y) - f_\kappa^*(y)), \tag{14}$$
where $f_\kappa(y)^* := \inf_z f_\kappa(z;y)$. If the method $\mathcal{M}$ is randomized, we require the same inequality to hold in expectation.

2. The rates τ_κ and the constants A_κ are increasing in κ.

Remark 4.1. The linear convergence we assume here for $\mathcal{M}$ differs from the one considered by [22], which was given in terms of function values. However, if the problem is a composite one, both points of view are near-equivalent, as discussed in Section A and the precise relationship is given in Appendix C. We choose the norm of the subgradient as our measurement because the complexity analysis is easier.

Then, a straightforward analysis bounds the computational complexity to achieve an ε-stationary point.

Lemma 4.2. *Let us consider a strongly convex problem $f_\kappa(\cdot; y)$ and a linearly convergent method $\mathcal{M}$ generating a sequence of iterates $\{z_t\}_{t\geq 0}$. Define $T(\varepsilon) = \inf\{t \geq 1, \operatorname{dist}\big(0, \partial f_\kappa(z_t; y)\big) \leq \varepsilon\}$, where ε is the target accuracy; then,*

1. *If $\mathcal{M}$ is deterministic,*
$$T(\varepsilon) \leq \frac{1}{\tau_\kappa} \log\left(\frac{A_\kappa \left(f_\kappa(z_0; y) - f_\kappa^*(y)\right)}{\varepsilon^2}\right).$$
2. *If $\mathcal{M}$ is randomized, then*
$$\mathbb{E}\left[T(\varepsilon)\right] \leq \frac{1}{\tau_\kappa} \log\left(\frac{A_\kappa \left(f_\kappa(z_0; y) - f_\kappa^*(y)\right)}{\tau_\kappa \varepsilon^2}\right).$$

   see Lemma C.1 of [22].

As we can see, we only lose a factor in the log term by switching from deterministic to randomized algorithms. For the sake of simplicity, we perform our analysis only for deterministic algorithms and the analysis for randomized algorithms holds in the same way in expectation.

Bounding the required iterations when $\kappa > \rho$ and restart strategy. Recall that we add a quadratic to f with the hope to make each subproblem convex. Thus, if ρ is known, then we should set $\kappa > \rho$. In this first stage, we show that whenever $\kappa > \rho$, then the number of inner calls to $\mathcal{M}$ can be bounded with a proper initialization. Consider the subproblem

$$\min_{x \in \mathbb{R}^p} \left\{ f_\kappa(x; y) = f(x) + \frac{\kappa}{2} \|x - y\|^2 \right\}, \tag{15}$$

and define the initialization point z_0 by

1. if f is smooth, then set $z_0 = y$;
2. if $f = f_0 + \psi$ is composite, with f_0 L-smooth, then set $z_0 = \operatorname{prox}_{\eta\psi}(y - \eta \nabla f_0(y))$ with $\eta \leq \frac{1}{L+\kappa}$.

Theorem 4.3. *Consider the subproblem (15) and suppose $\kappa > \rho$. Then initializing $\mathcal{M}$ at the previous z_0 generates a sequence of iterates $(z_t)_{t\geq 0}$ such that*

1. *in at most T_κ iterations where*
$$T_\kappa = \frac{1}{\tau_\kappa} \log\left(\frac{8A_\kappa(L+\kappa)}{(\kappa-\rho)^2}\right),$$
   the output z_T satisfies $f_\kappa(z_T; y) \leq f_\kappa(z_0; y)$ (descent condition) and dist$(0, \partial f_\kappa(z_T; y)) \leq \kappa \|z_T - y\|$ (adaptive stationary condition);
2. *in at most $S_\kappa \log(k+1)$ iterations where*
$$S_\kappa \log(k+1) = \frac{1}{\tau_\kappa} \log\left(\frac{8A_\kappa(L+\kappa)(k+1)^2}{(\kappa-\rho)^2}\right),$$
   the output z_S satisfies dist$(0, \partial f_\kappa(z_S; y)) \leq \frac{\kappa}{k+1} \|z_S - y\|$ (modified adaptive stationary condition).

The proof is technical and is presented in Appendix D. The lesson we learn here is that as soon as the subproblem becomes strongly convex, it can be solved in almost a constant number of iterations. Herein arises a problem–the choice of the smoothing parameter κ. On one hand, when f is already convex, we may want to choose κ small in order to obtain the desired optimal complexity. On the other hand, when the problem is non convex, a small κ may not ensure the strong convexity of the subproblems. Because of such different behavior according to the convexity of the function, we introduce an additional parameter κ_{cvx} to handle the regularization of the extrapolation step. Moreover, in order to choose a $\kappa > \rho$ in the nonconvex case, we need to know in advance an estimate of ρ. This is not an easy task for large scale machine learning problems such as neural networks. Thus we propose an adaptive step to handle it automatically.

Algorithm 2 4WD-Catalyst

input Fix a point $x_0 \in \operatorname{dom} f$, real numbers $\kappa_0, \kappa_{\text{cvx}} > 0$ and $T, S > 0$, and an opt. method $\mathcal{M}$.

initialization: $\alpha_1 = 1$, $v_0 = x_0$.

repeat for $k = 1, 2, \ldots$

1. Compute
$$(\bar{x}_k, \kappa_k) = \text{Auto-adapt } (x_{k-1}, \kappa_{k-1}, T).$$
2. Compute $y_k = \alpha_k v_{k-1} + (1 - \alpha_k) x_{k-1}$ and apply $S \log(k+1)$ iterations of $\mathcal{M}$ to find
$$\tilde{x}_k \approx \operatorname*{argmin}_{x \in \mathbb{R}^p} f_{\kappa_{\text{cvx}}}(x, y_k), \tag{16}$$
by using the initialization strategy described below (15).
3. Update v_k and α_{k+1} by
$$v_k = x_{k-1} + \tfrac{1}{\alpha_k}(\tilde{x}_k - x_{k-1}) \quad \text{and} \quad \alpha_{k+1} = \frac{\sqrt{\alpha_k^4 + 4\alpha_k^2} - \alpha_k^2}{2}.$$
4. Choose x_k to be any point satisfying $f(x_k) = \min\{f(\bar{x}_k), f(\tilde{x}_k)\}$.

until the stopping criterion $\operatorname{dist}\big(0, \partial f(\bar{x}_k)\big) < \varepsilon$

4.2 4WD-Catalyst: adaptation to weak convexity

We now introduce 4WD-Catalyst, presented in Algorithm 2, which can automatically adapt to the *unknown weak convexity* constant of the objective. The algorithm relies on a procedure to automatically adapt to ρ, described in Algorithm 3.

The idea is to fix in advance a number of iterations T, let $\mathcal{M}$ run on the subproblem for T iterations, output the point z_T, and check if a sufficient decrease occurs. We show that if we set $T = \tilde{O}(\tau_L^{-1})$, where the notation $\tilde{O}$ hides logarithmic dependencies in L and A_L, where L is the Lipschitz constant of the smooth part of f; then, if the subproblem were convex, the following conditions would be guaranteed:

1. Descent condition: $f_\kappa(z_T; x) \le f_\kappa(x; x)$;
2. Adaptive stationary condition: $\operatorname{dist}\big(0, \partial f_\kappa(z_T; x)\big) \le \kappa \left\| z_T - x \right\|$.

Thus, if either condition is not satisfied, then the subproblem is deemed not convex and we double κ and repeat. The procedure yields an estimate of ρ in a logarithmic number of increases; see Lemma D.3.

Relative stationarity and predefining S. One of the main differences of our approach with the Catalyst algorithm of [22] is to use a *pre-defined* number of iterations, T and S, for solving the subproblems. We introduce κ_{cvx}, a $\mathcal{M}$ dependent smoothing parameter and set it in the same way as the smoothing parameter in [22]. The automatic acceleration of our algorithm when the problem is convex is due to extrapolation steps in Step 2-3 of Basic 4WD-Catalyst. We show that if we set $S = \tilde{O}\left(\tau_{\kappa_{\text{cvx}}}^{-1}\right)$, where $\tilde{O}$ hides logarithmic dependencies in L, κ_κ, and $A_{\kappa_{\text{cvx}}}$, then we can be sure that, for convex objectives,

$$\operatorname{dist}\big(0, \partial f_{\kappa_{\text{cvx}}}(\tilde{x}_k; y_k)\big) < \frac{\kappa_{\text{cvx}}}{k+1} \left\| \tilde{x}_k - y_k \right\|. \tag{17}$$

This relative stationarity of $\tilde{x}_k$, including the choice of κ_{cvx}, shall be crucial to guarantee that the scheme accelerates in the convex setting. An additional $k+1$ factor appears compared to the previous adaptive stationary condition because we need higher accuracy for solving the subproblem to achieve the accelerated rate in $1/\sqrt{\varepsilon}$.

We shall see in the experiments that our strategy of predefining T and S works quite well. The theoretical bounds we derive are, in general, too conservative; we observe in our experiments that one may choose T

and S significantly smaller than the theory suggests and still retain the stopping criteria.

Algorithm 3 Auto-adapt (y, κ, T)

input $y \in \mathbb{R}^p$, method $\mathcal{M}$, $\kappa > 0$, number of iterations T.

Repeat Compute

$$z_T \approx \operatorname*{argmin}_{z \in \mathbb{R}^p} f_\kappa(z; y).$$

by running T iterations of $\mathcal{M}$ by using the initialization strategy described below (15).

If $f_\kappa(z_T; y) > f_\kappa(y; y)$ or $\operatorname{dist}(\partial f_\kappa(z_T; y), 0) > \kappa \|z_T - y\|$,

then go to repeat with $\kappa \to 2\kappa$.

else go to output.

output (z_T, κ).

To derive the global complexity results for 4WD-Catalyst that match optimal convergence guarantees, we make a distinction between the regularization parameter κ in the proximal point step and in the extrapolation step. For the proximal point step, we apply Algorithm 3 to adaptively produce a sequence of κ_k initializing at $\kappa_0 > 0$, an initial guess of ρ. The resulting $\bar{x}_k$ and κ_k satisfy both the following inequalities:

$$\operatorname{dist}\big(0, \partial f_{\kappa_k}(\bar{x}_k; x_{k-1})\big) < \kappa_k \|\bar{x}_k - x_k\| \text{ and } f_{\kappa_k}(\bar{x}_k; x_{k-1}) \leq f_{\kappa_k}(x_{k-1}; x_{k-1}). \tag{18}$$

For the extrapolation step, we introduce the parameter κ_{cvx} which essentially depends on the Lipschitz constant L. The choice is the same as the smoothing parameter in [22] and depends on the method $\mathcal{M}$. With a similar predefined iteration strategy, the resulting $\tilde{x}_k$ satisfies the following inequality if the original objective is convex,

$$\operatorname{dist}\big(0, \partial f_{\kappa_{\text{cvx}}}(\tilde{x}_k; y_k)\big) < \frac{\kappa_{\text{cvx}}}{k+1} \|\tilde{x}_k - y_k\|. \tag{19}$$

4.3 Convergence analysis

Let us next postulate that T and S are chosen large enough to guarantee that $\bar{x}_k$ and $\tilde{x}_k$ satisfy conditions (18) and (19) for the corresponding subproblems, and see how the outer algorithm complexity resembles the guarantees of Theorem 3.1 and Theorem 3.2. The main technical difference is that κ changes at each iteration k, which requires keeping track of the effects of κ_k and κ_{cvx} on the proof.

Theorem 4.4 (Outer-loop complexity, 4WD-Catalyst)**.** *Fix real constants $\kappa_0, \kappa_{\text{cvx}} > 0$, and $x_0 \in \operatorname{dom} f$. Set $\kappa_{\max} := \max_{k \geq 1} \kappa_k$. Suppose that the number of iterations T is such that $\bar{x}_k$ satisfies* (18)*. Define $f^* := \lim_{k \to \infty} f(x_k)$. Then for any $N \geq 1$, the iterates generated by Algorithm 2 satisfy,*

$$\min_{j=1,\ldots,N} \operatorname{dist}^2\big(0, \partial f(\bar{x}_j)\big) \leq \frac{8\kappa_{\max}}{N}(f(x_0) - f^*).$$

If in addition the function f is convex and S_k is chosen so that $\tilde{x}_k$ satisfies (19)*, then*

$$\min_{j=1,\ldots,2N} \operatorname{dist}^2\big(0, \partial f(\bar{x}_j)\big) \leq \frac{32\kappa_{\max}\kappa_{\text{cvx}}}{N(N+1)^2} \|x^* - x_0\|^2,$$

and

$$f(x_N) - f(x^*) \leq \frac{4\kappa_{\text{cvx}}}{(N+1)^2} \|x^* - x_0\|^2, \tag{20}$$

where x^* is any minimizer of the function f.

Inner-loop Complexity In light of Theorem 4.4, we must now understand how to choose T and S as small as possible, while guaranteeing that $\bar{x}_k$ and $\tilde{x}_k$ satisfy (18) and (19) hold for each k. The quantities T and S depend on the method $\mathcal{M}$'s convergence rate parameter τ_κ which only depends on L and κ. For example, the convergence rate parameter $\tau_\kappa^{-1} = (L+\kappa)/\kappa$ for gradient descent and $\tau_\kappa^{-1} = n + (L+\kappa)/\kappa$ for SVRG. The values of T and S must be set beforehand without knowing the true value of the weak convexity constant ρ. Using Theorem 4.3, we assert the following choices for T and S.

Theorem 4.5 (Inner complexity for 4WD-Catalyst : determining the values T and S)**.** *Suppose the stopping criteria are* (18) *and* (19) *as in in Theorem 4.4, and choose T and S in Algorithm 2 to be the smallest numbers satisfying*

$$T \geq \frac{1}{\tau_L} \log\left(\frac{40 A_{4L}}{L}\right),$$

and

$$S \log(k+1) \geq \frac{1}{\tau_{\kappa_{\text{cvx}}}} \log\left(\frac{8 A_{\kappa_{\text{cvx}}} (\kappa_{\text{cvx}} + L)(k+1)^2}{\kappa_{\text{cvx}}^2}\right),$$

for all k. In particular,

$$T = O\left(\frac{1}{\tau_L} \log\left(A_{4L}, L\right)\right),$$
$$S = O\left(\frac{1}{\tau_{\kappa_{\text{cvx}}}} \log(A_{\kappa_{\text{cvx}}}, L, \kappa_{\text{cvx}})\right).$$

Then $\kappa_{\max} \leq 4L$ and the following hold for any index $k \geq 1$:

1. *Generating $\bar{x}_k$ in Algorithm 2 requires at most $\tilde{O}\left(\tau_L^{-1}\right)$ iterations of $\mathcal{M}$;*
2. *Generating $\tilde{x}_k$ in Algorithm 2 requires at most $\tilde{O}\left(\tau_{\kappa_{\text{cvx}}}^{-1}\right)$ iterations of $\mathcal{M}$.*

where $\tilde{O}$ hides universal constants and logarithmic dependencies on k, L, κ_{cvx}, A_L, and $A_{\kappa_{\text{cvx}}}$.

Appendix D is devoted to proving Theorem 4.5, but we outline below the general procedure and state the two main propositions (see Proposition 4.6 and Proposition 4.7).

We summarize the proof of Theorem 4.5 as followed:

1. When $\kappa > \rho + L$, we compute the number of iterations of $\mathcal{M}$ to produce a point satisfying (18). Such a point will become $\bar{x}_k$.
2. When the function f is convex, we compute the number of iterations of $\mathcal{M}$ to produce a point which satisfies the (19) condition. Such a point will become the point $\tilde{x}_k$.
3. We compute the smallest number of times we must double κ_0 until it becomes larger than $\rho + L$. Thus eventually the condition $4L \geq \kappa > \rho + L$ will occur.
4. We always set the number of iterations of $\mathcal{M}$ to produce $\bar{x}_k$ and $\tilde{x}_k$ as in Step 1 and Step 2, respectively, regardless of whether $f_\kappa(\cdot; x_k)$ is convex or f is convex.

The next proposition shows that Auto-adapt terminates with a suitable choice for $\bar{x}_k$ after T number of iterations.

Proposition 4.6 (Inner complexity for $\bar{x}_k$)**.** *Suppose $\rho + L < \kappa \leq 4L$. By initializing the method $\mathcal{M}$ using the strategy suggested in Algorithm 2 for solving*

$$\min_z \left\{ f_\kappa(z; x) := f(z) + \frac{\kappa}{2} \|z - x\|^2 \right\}$$

we may run the method $\mathcal{M}$ for at least T iterations, where

$$T \geq \frac{1}{\tau_L} \log\left(\frac{40 A_{4L}}{L}\right);$$

then, the output z_T satisfies $f_\kappa(z_T; x) \leq f_\kappa(x; x)$ and $dist\big(0, \partial f_\kappa(z_T; x)\big) \leq \kappa \|z_T - x\|$.

Under the additional assumption that the function f is convex, we produce a point with (19) when the number of iterations S is chosen sufficiently large.

Proposition 4.7 (Inner-loop complexity for $\tilde{x}_k$)**.** *Consider the method $\mathcal{M}$ with the initialization strategy suggested in Algorithm 2 for minimizing $f_{\kappa_{\text{cvx}}}(\cdot; y_k)$ with linear convergence rates of the form* (14)*. Suppose the function f is convex. If the number of iterations of $\mathcal{M}$ is greater than*

$$S = O\left(\frac{1}{\tau_{\kappa_{\text{cvx}}}} \log(A_{\kappa_{\text{cvx}}}, L, \kappa_{\text{cvx}})\right)$$

such that

$$S \log(k+1) \geq \frac{1}{\tau_{\kappa_{\text{cvx}}}} \log\left(\frac{8 A_{\kappa_{\text{cvx}}} (\kappa_{\text{cvx}} + L)(k+1)^2}{\kappa_{\text{cvx}}^2}\right), \tag{21}$$

then, the output $\tilde{z}_S = \tilde{x}_k$ satisfies $\|\partial f_{\kappa_{\text{cvx}}}(\tilde{z}_S)\| < \frac{\kappa_{\text{cvx}}}{k+1} \|\tilde{z}_{S_k} - y_k\|$ for all $k \geq 1$.

We can now derive global complexity bounds by combining Theorem 4.4 and Theorem 4.5, and a good choice for the constant κ_{cvx}.

Theorem 4.8 (Global complexity bounds for 4WD-Catalyst)**.** *Choose Choose T and S as in Theorem 4.5. We let $\tilde{O}$ hide universal constants and logarithmic dependencies in A_L, $A_{\kappa_{\text{cvx}}}$, L, ε, κ_0, κ_{cvx}, and $\|x^* - x_0\|^2$. Then, the following statements hold.*

1. *Algorithm 2 generates a point x satisfying $dist\big(0, \partial f(x)\big) \leq \varepsilon$ after at most*

$$\tilde{O}\left(\left(\tau_L^{-1} + \tau_{\kappa_{\text{cvx}}}^{-1}\right) \cdot \frac{L(f(x_0) - f^*)}{\varepsilon^2}\right)$$

   iterations of the method $\mathcal{M}$.

2. *If f is convex, then Algorithm 2 generates a point x satisfying $dist\big(0, \partial f(x)\big) \leq \varepsilon$ after at most*

$$\tilde{O}\left(\left(\tau_L^{-1} + \tau_{\kappa_{\text{cvx}}}^{-1}\right) \cdot \frac{L^{1/3}\left(\kappa_{\text{cvx}} \|x^* - x_0\|^2\right)^{1/3}}{\varepsilon^{2/3}}\right)$$

   iterations of the method $\mathcal{M}$.

3. *If f is convex, then Algorithm 2 generates a point x satisfying $f(x) - f^* \leq \varepsilon$ after at most*

$$\tilde{O}\left(\left(\tau_L^{-1} + \tau_{\kappa_{\text{cvx}}}^{-1}\right) \cdot \frac{\sqrt{\kappa_{\text{cvx}} \|x^* - x_0\|^2}}{\sqrt{\varepsilon}}\right)$$

   iterations of the method $\mathcal{M}$.

Remark 4.9. In general, the linear convergence parameter of $\mathcal{M}$, τ_κ, depends on the condition number of the problem f_κ. Here, τ_L and $\tau_{\kappa_{\text{cvx}}}$ are precisely given by plugging in $\kappa = L$ and κ_{cvx} respectively into τ_κ. To clarify, let $\mathcal{M}$ be SVRG, τ_κ is given by $\frac{1}{n + \frac{\kappa + L}{\kappa}}$ which yields $\tau_L = 1/(n+2)$. A more detailed computation is given in Table 5.1. For all the incremental methods we considered, these parameters τ_L and τ_κ are on the order of $1/n$.

Remark 4.10. If $\mathcal{M}$ is a first order method, the convergence guarantee in the convex setting is *near-optimal*, up to logarithmic factors, when compared to $O(1/\sqrt{\varepsilon})$ [22, 37]. In the non-convex setting, our approach matches, up to logarithmic factors, the best known rate for this class of functions, namely $O(1/\varepsilon^2)$ [10, 9]. Moreover, our rates dependence on the dimension and Lipschitz constant equals, up to log factors, the best known dependencies in both the convex and nonconvex setting. These logarithmic factors may be the price we pay for having a generic algorithm.

5 Applications to Existing Algorithms

We now show how to accelerate existing algorithms $\mathcal{M}$ and compare the convergence guaranties before and after 4WD-Catalyst. In particular, we focus on the gradient descent algorithm and on the incremental methods SAGA and SVRG. For all the algorithms considered, we state the convergence guaranties in terms of the *total number of iterations* (in expectation, if appropriate) to reach an accuracy of ε; in the convex setting, the accuracy is stated in terms of functional error, $f(x) - \inf f < \varepsilon$ and in the nonconvex setting, the appropriate measure is stationarity, namely $\operatorname{dist}(0, \partial f(x)) < \varepsilon$. All the algorithms considered have formulations for the composite setting with analogous convergence rates. Table 5 presents convergence rates for SAGA [12], (prox) SVRG [38], and gradient descent (FG).

	$\rho > 0$		$\rho = 0$	
	Original	4WD-Catalyst	Original	4WD-Catalyst
FG	$\mathcal{O}\left(n\frac{L}{\varepsilon^2}\right)$	$\tilde{O}\left(n\frac{L}{\varepsilon^2}\right)$	$O\left(n\frac{L}{\varepsilon}\right)$	$\tilde{O}\left(n\sqrt{\frac{L}{\varepsilon}}\right)$
SVRG [38]	not avail.	$\tilde{O}\left(n\frac{L}{\varepsilon^2}\right)$	not avail.	$\tilde{O}\left(\sqrt{n}\sqrt{\frac{L}{\varepsilon}}\right)$
SAGA [12]	not avail.	$\tilde{O}\left(n\frac{L}{\varepsilon^2}\right)$	$O\left(n\frac{L}{\varepsilon}\right)$	$\tilde{O}\left(\sqrt{n}\sqrt{\frac{L}{\varepsilon}}\right)$

Table 1: Comparison of rates of convergence, before and after the 4WD-Catalyst , resp. in the non-convex and convex cases. For the comparision, in the convex case, we only present the number of iterations to obtain a point x satisfying $f(x) - f^* < \varepsilon$. In the non-convex case, we show the number of iterations to obtain a point x satisfying $\operatorname{dist}(0, \partial f(x)) < \varepsilon$.

The original SVRG [38] has no guarantees for nonconvex functions; however, there is a nonconvex extension of SVRG in [32]. Their convergence rate achieves a better dependance on n compared to our results, namely $O(\frac{n^{2/3}L}{\varepsilon^2})$. This is done by performing a strategy of minibatching. In order to achieve a similar dependency on n, we require a tighter bound for SVRG with minibatching applied to μ-strongly convex problems, namely $O\left(\left(n^{2/3} + \frac{L}{\mu}\right)\log\left(\frac{1}{\varepsilon}\right)\right)$. To the best of our knowledge, such a rate is currently unknown.

5.1 Practical parameters choices and convergence rates

The smoothing parameter κ_{cvx} drives the convergence rate of 4WD-Catalyst in the convex setting. To determine κ_{cvx}, we pretend $\rho = 0$ and compute the global complexity of our scheme. As such, we end up with the same complexity result as Catalyst [22]. Following their work, the rule of thumb is to maximize the ratio $\tau_\kappa/\sqrt{L+\kappa}$ for convex problems. On the other hand, the choice of κ_0 is independent of $\mathcal{M}$; it is an initial lower estimate for the weak convexity constant ρ. In practice, we typically choose $\kappa_0 = \kappa_{\text{cvx}}$; For incremental approaches a natural heuristic is also to choose $S = T = n$, meaning that S iterations of $\mathcal{M}$ performs one pass over the data. In Table 5.1, we present the values of κ_{cvx} used for various algorithms, as well as other quantities that are useful to derive the convergence rates.

Full gradient method. A first illustration is the algorithm obtained when accelerating the regular "full" gradient (FG). Here, the optimal choice for κ_{cvx} is L. In the convex setting, we get an accelerated rate of $O(n\sqrt{L/\varepsilon}\log(1/\varepsilon))$ which agrees with Nesterov's accelerated variant (AFG) up to logarithmic factors.

On the other hand, in the nonconvex setting, our approach achieves no worse rate than $O(nL/\varepsilon^2 \log(1/\varepsilon))$, which agrees with the standard gradient descent up to logarithmic factors. We note that under stronger assumptions, namely C^2-smoothness of the objective, the accelerated algorithm in [8] achieves the same rate as (AFG) for the convex setting and $O(\varepsilon^{-7/4} \log(1/\varepsilon))$ for the nonconvex setting. Their approach, however, does not extend to composite setting nor to stochastic methods. Our marginal loss is the price we pay for considering a much larger class of functions.

Randomized incremental gradient. We now consider randomized incremental gradient methods such as SAGA [12] and (prox) SVRG [38]. Here, the optimal choice for κ_{cvx} is $O(L/n)$. Under the convex setting, we achieve an accelerated rate of $O(\sqrt{n}\sqrt{L/\varepsilon}\log(1/\varepsilon))$. A direct application of SVRG and SAGA have no convergence guarantees in the non-convex setting. With our approach, the resulting algorithm matches the guarantees for FG up to log factors.

Variable	Description	GD	SVRG	SAGA
$1/\tau_L$	linear convergence parameter with $\kappa = L$	2	$n+2$	$4n$
κ_{cvx}	smoothing parameter for convex setting	L	$L/(n-1)$	$3L/(4n-3)$
$1/\tau_{\kappa_{\text{cvx}}}$	linear convergence parameter with κ_{cvx}	2	$2n$	$4n$
A_{4L}	constant from the convergence rate of $\mathcal{M}$	$8L$	$8L$	$8Ln$

Table 2: Values of various quantities that are useful to derive the convergence rate of the different optimization methods.

5.2 Detailed derivation of convergence rates

Using the values of Table 5.1, we may now specialize our convergence results to different methods.

Gradient descent. The number of iterations in the inner loop are

$$T \geq 2\log(320)$$
$$S\log(k+1) \geq 2\log\left(64(k+1)^2\right)$$

The global complexity for gradient descent is

1. Algorithm 2 will generate a point x satisfying $\operatorname{dist}\big(0, \partial f(x)\big) \leq \varepsilon$ after at most

$$O\left[\frac{nL(f(x_0)-f^*)}{\varepsilon^2} \cdot \log\left(\frac{L^2(f(x_0)-f^*)^2}{\varepsilon^4}\right) + n\log\left(\frac{L}{\kappa_0}\right)\right]$$

   gradient computations.

2. If f is convex, then Algorithm 2 will generate a point x satisfying $\operatorname{dist}\big(0, \partial f(x)\big) \leq \varepsilon$ after at most

$$O\left[\frac{nL^{2/3}\left\|x_0-x^*\right\|^{2/3}}{\varepsilon^{2/3}} \cdot \log\left(\frac{L^{4/3}\left\|x_0-x^*\right\|^{4/3}}{\varepsilon^{4/3}}\right) + n\log\left(\frac{L}{\kappa_0}\right)\right]$$

   gradient computations.

3. If f is convex, then Algorithm 2 will generate a point x satisfying $f(x) - f^* \leq \varepsilon$ after at most

$$O\left[\frac{n\sqrt{L}\left\|x^*-x_0\right\|}{\sqrt{\varepsilon}} \cdot \log\left(\frac{L\left\|x_0-x^*\right\|^2}{\varepsilon}\right) + n\log\left(\frac{L}{\kappa_0}\right)\right]$$

   gradient computations.

SVRG. For SVRG, the number of iterations in the inner loop are

$$T \geq (n+2)\log(320)$$
$$S\log(k+1) \geq 2n\log\left(64\cdot n^2\cdot(k+1)^2\right).$$

The global complexity for SVRG when n is sufficiently large is

1. Algorithm 2 will generate a point x satisfying $\operatorname{dist}\big(0,\partial f(x)\big)\leq\varepsilon$ after at most

$$O\left[\frac{nL(f(x_0)-f^*)}{\varepsilon^2}\cdot\log\left(\frac{n^2L^2(f(x_0)-f^*)^2}{\varepsilon^4}\right)+n\log\left(\frac{L}{\kappa_0}\right)\right]$$

gradient computations.

2. If f is convex, then Algorithm 2 will generate a point x satisfying $\operatorname{dist}\big(0,\partial f(x)\big)\leq\varepsilon$ after at most

$$O\left[\frac{n^{2/3}L^{2/3}\left\|x^*-x_0\right\|^{2/3}}{\varepsilon^{2/3}}\log\left(\frac{n^{4/3}L^{4/3}\left\|x^*-x_0\right\|^{4/3}}{\varepsilon^{4/3}}\right)+n^{2/3}\log\left(\frac{L}{\kappa_0}\right)\right]$$

gradient computations.

3. If f is convex, then Algorithm 2 will generate a point x satisfying $f(x)-f^*\leq\varepsilon$ after at most

$$O\left[\frac{\sqrt{nL}\left\|x^*-x_0\right\|}{\sqrt{\varepsilon}}\cdot\log\left(\frac{nL\left\|x_0-x^*\right\|^2}{\varepsilon}\right)+\sqrt{n}\log\left(\frac{L}{\kappa_0}\right)\right]$$

gradient computations.

SAGA We observe that the variables for SAGA are the same as for SVRG up to a multiplicative factors. Therefore, the global complexities results for SAGA are, up to constant factors, the same as SVRG.

6 Experiments

We investigate the performance of 4WD-Catalyst in two standard non-convex problems in machine learning. We report experimental results of 4WD-Catalyst when applied to two different algorithms: SVRG [38] and SAGA [12]. We compare the following algorithms:

- Nonconvex SVRG/SAGA [32]: stepsize $\eta = 1/Ln^{2/3}$;
- Convex SVRG/SAGA [12, 38]: stepsize $\eta = 1/2L$;
- 4WD-Catalyst SVRG/SAGA: stepsize $\eta = 1/2L$.

The original version of SVRG (resp. SAGA), convex SVRG (resp. SAGA), was designed for minimizing convex objectives. We report their results, while there is no theoretical guarantee on their behavior when venturing into nonconvex terrains. We also report the results of recently proposed variants, Nonconvex SVRG/SAGA, designed for minimizing nonconvex objectives. The proposed algorithms 4WD-Catalyst SVRG and 4WD-Catalyst SAGA enjoy the strong theoretical guarantees stated in Sec. 3.

Parameter settings We start from an initial estimate of the Lipschitz constant L and use the theoretically recommended $\kappa_0 = \kappa_{\text{cvx}} = 2L/n$. The number of inner iterations is to $T = S = n$ in all experiments, which boils down to making one pass at most over the data for solving each sub-problem. We simply drop the $\log(k)$ dependency while solving the subproblem in (16). These choices turn out to be justified *a posteriori*, as both SVRG and SAGA have a much better convergence rate in practice than the theoretical rate derived from a worst-case analysis. Indeed, in all experiments, one pass over the data to solve each sub-problem is enough to guarantee sufficient descent.

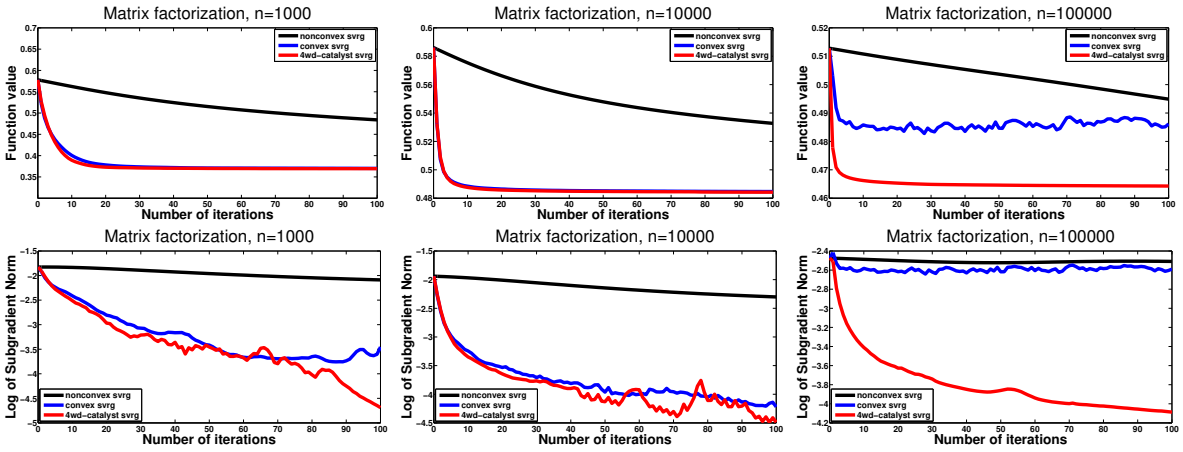

Figure 1: Dictionary learning experiments using SVRG. We plot the function value (top) and the subgradient norm (bottom). From left to right, we vary the size of dataset from $n = 1\,000$ to $n = 100\,000$.

Sparse matrix factorization a.k.a. dictionary learning. Dictionary learning consists of representing a dataset $X = [x_1, \cdots, x_n] \in \mathbb{R}^{m \times n}$ as a product $X \approx DA$, where D in $\mathbb{R}^{m \times p}$ is called a dictionary, and A in $\mathbb{R}^{p \times n}$ is a sparse matrix. The classical non-convex formulation [see 24] is

$$\min_{D \in \mathcal{C}, A \in \mathbb{R}^{p \times n}} \sum_{i=1}^{n} \frac{1}{2} \|x_i - D\alpha_i\|_2^2 + \psi(\alpha_i),$$

where $A = [\alpha_1 \cdots \alpha_n]$ carries the decomposition coefficients of signals $x_1 \cdots x_n$, ψ is a sparsity-inducing regularization and $\mathcal{C}$ is chosen as the set of matrices whose columns are in the ℓ_2-ball. An equivalent point of view is the finite-sum problem $\min_{D \in \mathcal{C}} \frac{1}{n} \sum_{i=1}^{n} f_i(D)$ with

$$f_i(D) := \min_{\alpha \in \mathbb{R}^p} \frac{1}{2} \|x_i - D\alpha\|_2^2 + \psi(\alpha). \tag{22}$$

We consider the elastic-net regularization $\psi(\alpha) = \frac{\mu}{2}\|\alpha\|^2 + \lambda\|\alpha\|_1$ of [39], which has a sparsity-inducing effect, and report the corresponding results in Figures 1 and 2, learning a dictionary in $\mathbb{R}^{m \times p}$ with $p = 256$ elements, on a set of whitened normalized image patches of size $m = 8 \times 8$. Parameters are standard ones in this literature [24]—that is, a small value $\mu = 1e-5$, and $\lambda = 0.25$, leading to sparse matrices A (on average ≈ 4 non-zero coefficients per column of A). Note that our implementations are based on the open-source SPAMS toolbox [25].[2]

Neural networks. We consider now simple binary classification problems for learning neural networks. Assume that we are given a training set $\{a_i, b_i\}_{i=1}^n$, where the variables b_i in $\{-1, +1\}$ represent class labels, and a_i in $\mathbb{R}^p$ are feature vectors. The estimator of a label class is now given by a two-layer neural network $\hat{b} = \text{sign}(W_2^\top \sigma(W_1^\top a))$, where W_1 in $\mathbb{R}^{p \times d}$ represents the weights of a hidden layer with d neurons, W_2 in $\mathbb{R}^d$ carries the weight of the network's second layer, and $\sigma(u) = \log(1 + e^u)$ is a non-linear function, applied pointwise to its arguments. We fix the number of hidden neurons to $d = 100$ and use the logistic loss to fit the estimators to the true labels. Since the memory required by SAGA becomes n times larger than SVRG for nonlinear models, which is problematic for large n, we can only perform experiments with SVRG. The experimental results are reported on two datasets alpha and covtype in Figures 3 and 4.

[2] available here `http://spams-devel.gforge.inria.fr`.

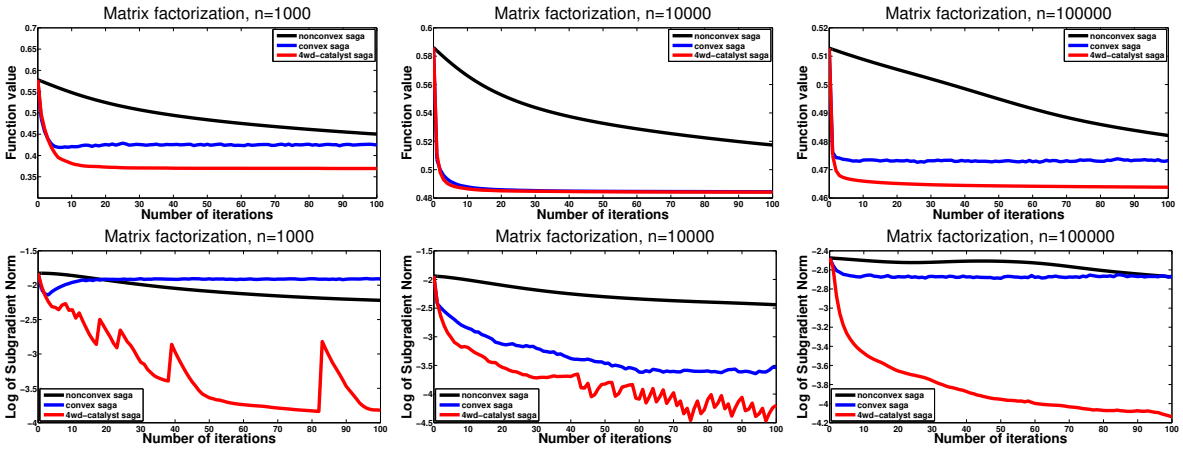

Figure 2: Dictionary learning experiments using SAGA. We plot the function value (top) and the subgradient norm (bottom). From left to right, we vary the size of dataset from $n = 1\,000$ to $n = 100\,000$.

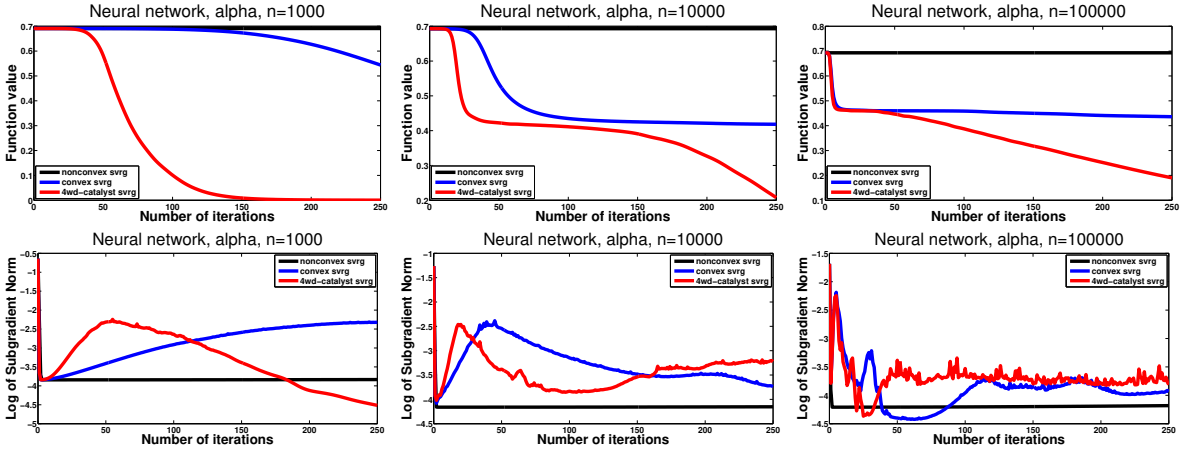

Figure 3: Neural network experiments on subsets of dataset alpha. From left to right, we vary the size of the dataset's subset from $n = 1\,000$ to $n = 100\,000$.

Initial estimates of L. The proposed algorithm 4WD-Catalyst requires an initial estimate of the Lipschitz constant L. In the problems we are considering, there is no simple closed form formula available to compute an estimate of L. We use following heuristics to estimate L:

1. For matrix factorization, it can be shown that the function f_i defined in (22) is differentiable according to Danskin's theorem [see Bertsekas [3], Proposition B.25] and its gradient is given by

$$\nabla_D f_i(D) = -(x_i - D\alpha_i(D))\alpha_i(D)^T \quad \text{where} \quad \alpha_i(D) \in \operatorname*{argmin}_{\alpha \in \mathbb{R}^p} \frac{1}{2} \|x_i - D\alpha\|^2 + \psi(\alpha).$$

   If the coefficients α_i were fixed, the gradient would be linear in D and thus admit $\|\alpha_i\|^2$ as Lipschitz constant. Therefore, when initializing our algorithm at D_0, we find $\alpha_i(D_0)$ for any $i \in [1, n]$ and use $\max_{i \in [1,n]} \|\alpha_i(D_0)\|^2$ as an estimate of L.

2. For neural networks, the formulation we are considering is actually differentiable. We randomly gen-

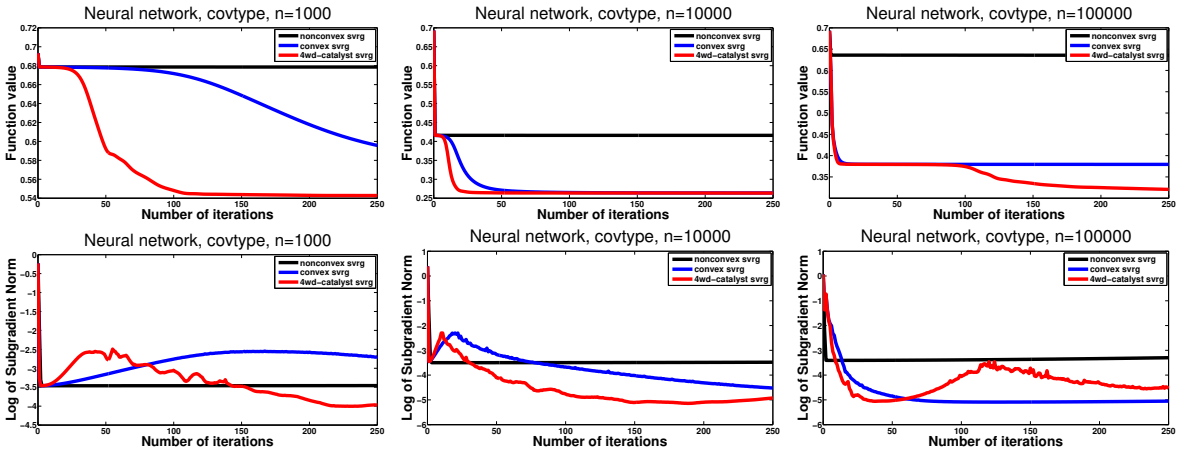

Figure 4: Neural network experiments on subsets of datasets alpha (top) and covtype (bottom).

erates two pairs of weight vectors (W_1, W_2) and (W_1', W_2') and use the quantity

$$\max_{i\in[1,n]} \left\{ \frac{\|\nabla f_i(W_1, W_2) - \nabla f_i(W_1', W_2)\|}{\|W_1 - W_1'\|}, \frac{\|\nabla f_i(W_1, W_2) - \nabla f_i(W_1, W_2')\|}{\|W_2 - W_2'\|} \right\}$$

as an estimate of the Lipschitz constant, where f_i denotes the loss function respect to i-th training sample (a_i, b_i). We separate weights in each layer to estimate the Lipschitz constant *per layer*. Indeed the scales of the weights can be quite different across layers.

Computational cost. For the Convex-SVRG and Nonconvex-SVRG, one iteration corresponds to one pass over the data in the plots. On the one hand, since 4WD-Catalyst-SVRG solves two sub-problems per iteration, the cost per iteration is twice that of the Convex-SVRG and Nonconvex-SVRG. On the other hand, in the experiments, we observe that, every time acceleration occurs, then $\tilde{x}_k$ is almost always preferred to $\bar{x}_k$ in step 4 of 4WD-Catalyst, hence half of the computations are in fact not performed when running 4WD-Catalyst-SVRG.

We report in Figure 5 an experimental study where we vary S on the neural network example. In terms of number of iterations, of course, the larger S_k the better the performance. This is not surprising as we solve each subproblem more accurately. Nevertheless, in terms of number of gradient evaluations, the relative performance is reversed. There is clearly no benefit to take larger S_k. This justifies in hindsight our choice of setting $S = 1$.

Experimental conclusions. In matrix factorization experiments, we observe that 4WD-Catalyst-SVRG always outperforms the competing algorithms. Nonconvex-SVRG has slower convergence in objective values and Convex-SVRG is not always converging; see in particular right panel in Fig. 1. Therefore 4WD-Catalyst-SVRG offers a more stable option than Convex-SVRG for minimizing nonconvex objectives. Furthermore, in these experiments 4WD-Catalyst-SVRG enjoys a faster convergence in objective values. This confirms the remarkable ability of 4WD-Catalyst-SVRG to adapt to nonconvex terrains. Similar conclusions hold when applying 4WD-Catalyst to SAGA, which demonstrates how general 4WD-Catalyst is.

In neural network experiments, we observe that 4WD-Catalyst-SVRG converges much faster in terms of objective values than the competing algorithms. Nonconvex-SVRG with the theoretically recommended sequence of step-sizes [32] compares unfavorably here, which implies that the recommended step-sizes are too pessimistic hence too small. We also observe an interesting phenomenon: the subgradient norm may increase at some point then decrease, while the function value keeps decreasing, as the algorithm proceeds. This suggests that the extrapolation step, or the Auto-adapt procedure, is helpful to escape bad stationary

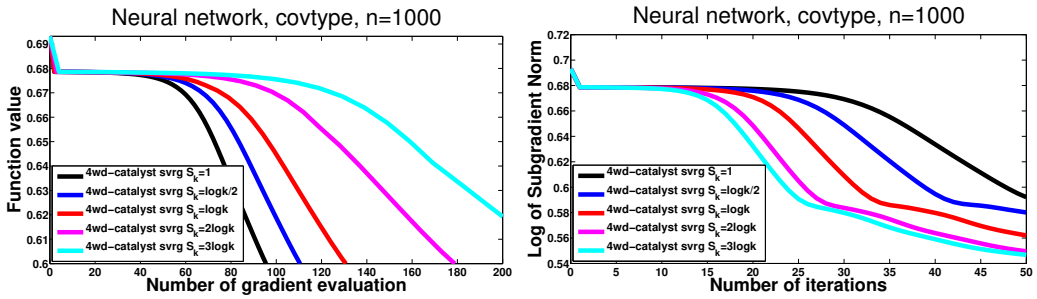

Figure 5: We run 50 iterations of 4WD-Catalyst SVRG with different choice of S on two-layer neural network. The data is a subset of dataset covtype. The x-axis is the number of gradient evaluations on the left, which is $T + S_k$ per iteration with $T = 1$; and the number of iterations on the right.

points, *e.g.*, saddle-points. A more systematic study is required to confirm such observation, we leave it as a potential direction of future work.

A Convergence rates in strongly-convex composite minimization

We now briefly discuss convergence rates, which are typically given in different forms in the convex and non-convex cases. If the weak-convex constant is known, we can form a strongly convex approximation similar to [22]. For that purpose, we consider a strongly-convex composite minimization problem

$$\min_{x\in\mathbb{R}^p} \; h(x) := f_0(x) + \psi(x),$$

where $f_0 \colon \mathbb{R}^p \to \mathbb{R}$ is μ-strongly convex and smooth with L-Lipschitz continuous gradient ∇f_0, and $\psi \colon \mathbb{R}^p \to \overline{\mathbb{R}}$ is a closed convex function with a computable proximal map

$$\operatorname{prox}_{\beta\psi}(y) := \operatorname*{argmin}_{z\in\mathbb{R}^p} \left\{ \psi(y) + \tfrac{1}{2\beta}\|z-y\|^2 \right\}.$$

Let x^* be the minimizer of h and h^* be the minimal value of h. In general, there are three types of measures of optimality that one can monitor: $\|x - x^*\|^2$, $h(x) - h^*$, and $\operatorname{dist}(0, \partial h(x))$.

Since h is strongly convex, the three of them are equivalent in terms of convergence rates if one can take an extra *prox-gradient step*:

$$[x]_L := \operatorname{prox}_{\psi/L}(x - L^{-1}\nabla f_0(x)).$$

To see this, define the *displacement vector*, also known as the gradient mapping, $g_L(x) := L(x - [x]_L)$, and notice the inclusion $g_L(x) \in \partial h([x]_L)$. In particular $g_L(x) = 0$ if and only if x is the minimizer of h. These next inequalities follow directly from Theorem 2.2.7 in [27]:

$$\begin{aligned}
\tfrac{1}{2L}\|g_L(x)\| \le &\|x - x^*\| \le \tfrac{2}{\mu}\|g_L(x)\| \\
\tfrac{\mu}{2}\|x - x^*\|^2 \le &h(x) - h^* \le \tfrac{1}{2\mu}|\partial h(x)|^2 \\
2\mu(h([x]_L) - h^*) \le &\|g_L(x)\|^2 \le 2L(h(x) - h([x]_L))
\end{aligned}$$

Thus, an estimate of any one of the four quantities $\|x - x^*\|$, $h(x) - h^*$, $\|g_L(x)\|$, or $\operatorname{dist}(0, \partial h(x))$ directly implies an estimate of the other three evaluated either at x or at $[x]_L$.

B Theoretical analysis of the basic algorithm

We present here proofs of the theoretical results of the paper. Althroughout the proofs, we shall work under the Assumptions on f stated in Section 3 and the Assumptions on $\mathcal{M}$ stated in Section 4.

B.1 Convergence guarantee of Basic 4WD-Catalyst

In Theorem 3.1 and Theorem 3.2 under an appropriate tolerance policy on the proximal subproblems (5) and (7), Basic 4WD-Catalyst performs no worse than an exact proximal point method in general, while automatically accelerating when f is convex. For this, we need the following observations.

Lemma B.1 (Growth of (α_k))**.** *Suppose the sequence $\{\alpha_k\}_{k\geq 1}$ is produced by Algorithm 1. Then, the following bounds hold for all $k \geq 1$:*

$$\frac{\sqrt{2}}{k+2} \leq \alpha_k \leq \frac{2}{k+1}.$$

Proof. This result is noted without proof in a remark of [36]. For completeness, we give below a simple proof using induction. Clearly, the statement holds for $k = 1$. Assume the inequality on the right-hand side holds for k. By using the induction hypothesis, we get

$$\alpha_{k+1} = \frac{\sqrt{\alpha_k^4 + 4\alpha_k^2} - \alpha_k^2}{2} = \frac{2}{\sqrt{1 + 4/\alpha_k^2} + 1} \leq \frac{2}{\sqrt{1 + (k+1)^2} + 1} \leq \frac{2}{k+2},$$

as claimed and the expression for α_{k+1} is given by explicitly solving (9). To show the lower bound, we note that for all $k \geq 1$, we have

$$\alpha_{k+1}^2 = (1 - \alpha_{k+1})\alpha_k^2 = \prod_{i=2}^{k+1}(1 - \alpha_i)\alpha_1^2 = \prod_{i=2}^{k+1}(1 - \alpha_i).$$

Using the established upper bound $\alpha_k \leq \frac{2}{k+1}$ yields

$$\alpha_{k+1}^2 \geq \prod_{i=2}^{k+1}\left(1 - \frac{2}{i+1}\right) = \frac{2}{(k+2)(k+1)} \geq \frac{2}{(k+2)^2}.$$

The result follows. □

Lemma B.2 (Prox-gradient and near-stationarity)**.** *Suppose y^+ satisfies dist$(0, \partial f_\kappa(y^+; y)) < \varepsilon$. Then, the inequality holds:*

$$\operatorname{dist}\big(0, \partial f(y^+)\big) \leq \varepsilon + \left\|\kappa(y^+ - y)\right\|.$$

Proof. We can find $\xi \in \partial f_\kappa(y^+; y)$ with $\|\xi\| \leq \varepsilon$. Taking into account $\partial f_\kappa(y^+; y) = \partial f(y^+) + \kappa(y^+ - y)$ the result follows. □

Next we establish convergence guarantees of Theorem 3.1 and Theorem 3.2 for Basic 4WD-Catalyst .

Proof of Theorem 3.2 and Theorem 3.2. The proof of Theorem 3.1 follows the analysis of inexact proximal point method [22, 17, 4]. The descent condition in (11) implies $\{f(x_k)\}_{k\geq 0}$ are monotonically decreasing. From this, we deduce

$$f(x_{k-1}) = f_\kappa(x_{k-1}; x_{k-1}) \geq f_\kappa(\bar{x}_k; x_{k-1}) \geq f(x_k) + \frac{\kappa}{2}\left\|\bar{x}_k - x_{k-1}\right\|^2. \tag{23}$$

Using the adaptive stationarity condition (11), we apply Lemma B.2 with $y = x_{k-1}$, $y^+ = \bar{x}_k$ and $\varepsilon = \kappa\left\|\bar{x}_k - x_{k-1}\right\|$; hence we obtain

$$\operatorname{dist}(0, \partial f(\bar{x}_k)) \leq 2\left\|\kappa(\bar{x}_k - x_{k-1})\right\|.$$

We combine the above inequality with (23) to deduce

$$\operatorname{dist}^2(0, \partial f(\bar{x}_k)) \leq 4 \left\| \kappa(\bar{x}_k - x_{k-1}) \right\|^2 \leq 8\kappa \left(f(x_{k-1}) - f(x_k) \right). \tag{24}$$

Summing $j = 1$ to N, we conclude

$$\begin{aligned} \min_{j=1,\ldots,N} \left\{ \operatorname{dist}^2(0, \partial f(\bar{x}_j)) \right\} &\leq \frac{4}{N} \sum_{j=1}^{N} \left\| \kappa(\bar{x}_k - x_{k-1}) \right\|^2) \\ &\leq \frac{8\kappa}{N} \left(\sum_{j=1}^{N} f(x_{j-1}) - f(x_j) \right) \\ &\leq \frac{8\kappa}{N} \left(f(x_0) - f^* \right). \end{aligned}$$

Next, suppose the function f is convex. Our analysis is similar to that of [36, 2]. Using the stopping criteria (12), fix an $\xi_k \in \partial f_\kappa(\tilde{x}_k; y_k)$ with $\|\xi_k\| < \frac{\kappa}{k+1} \|\tilde{x}_k - y_k\|$. For any $x \in \mathbb{R}^n$, Equation (10), and the strong convexity of the function $f_\kappa(\cdot; y_k)$ yields

$$f(x_k) \leq f(\tilde{x}_k) \leq f(x) + \frac{\kappa}{2} \left(\|x - y_k\|^2 - \|x - \tilde{x}_k\|^2 - \|\tilde{x}_k - y_k\|^2 \right) + \xi_k^T \left(\tilde{x}_k - x \right).$$

We substitute $x = \alpha_k x^* + (1 - \alpha_k) x_{k-1}$ where x^* is any minimizer of f. Using the convexity of f, the norm of ξ_k, and Equations (6) and (8), we deduce

$$\begin{aligned} f(x_k) \leq \alpha_k f(x^*) + (1 - \alpha_k) f(x_{k-1}) + \frac{\alpha_k^2 \kappa}{2} \left(\|x^* - v_{k-1}\|^2 - \|x^* - v_k\|^2 \right) \\ - \frac{\kappa}{2} \|\tilde{x}_k - y_k\|^2 + \frac{\alpha_k \kappa}{k+1} \|\tilde{x}_k - y_k\| \|x^* - v_k\|. \end{aligned} \tag{25}$$

Set $\theta_k = \frac{1}{k+1}$. Completing the square on Equation (25), we obtain

$$\frac{-\kappa}{2} \|\tilde{x}_k - y_k\|^2 + \alpha_k \theta_k \kappa \|\tilde{x}_k - y_k\| \|x^* - v_k\| \leq \frac{\kappa}{2} \left(\alpha_k \theta_k \right)^2 \|x^* - v_k\|^2.$$

Hence, we deduce

$$\begin{aligned} f(x_k) - f^* &\leq (1 - \alpha_k)(f(x_{k-1}) - f^*) + \frac{\alpha_k^2 \kappa}{2} \left(\|x^* - v_{k-1}\|^2 - \|x^* - v_k\|^2 \right) \\ &\quad + \frac{\kappa}{2} \left(\alpha_k \theta_k \right)^2 \|x^* - v_k\|^2. \\ &= (1 - \alpha_k)(f(x_{k-1}) - f^*) + \frac{\alpha_k^2 \kappa}{2} \left(\|x^* - v_{k-1}\|^2 - \left(1 - \theta_k^2\right) \|x^* - v_k\|^2 \right) \end{aligned}$$

Denote $A_k := 1 - \theta_k^2$. Subtracting f^* from both sides and using the inequality $\frac{1-\alpha_k}{\alpha_k^2} = \frac{1}{\alpha_{k-1}^2}$ and $\alpha_1 \equiv 1$, we derive the following recursion argument:

$$\begin{aligned} \frac{f(x_k) - f^*}{\alpha_k^2} + \frac{A_k \kappa}{2} \|x^* - v_k\|^2 &\leq \frac{1 - \alpha_k}{\alpha_k^2} \left(f(x_{k-1}) - f^* \right) + \frac{\kappa}{2} \|x^* - v_{k-1}\|^2 \\ &\leq \frac{1}{A_{k-1}} \left(\frac{f(x_{k-1}) - f^*}{\alpha_{k-1}^2} + \frac{A_{k-1} \kappa}{2} \|x^* - v_{k-1}\|^2 \right). \end{aligned}$$

The last inequality follows because $0 < A_{k-1} \leq 1$. Iterating N times,we deduce

$$\frac{f(x_N) - f^*}{\alpha_N^2} \leq \prod_{j=2}^{N} \frac{1}{A_{j-1}} \left(\frac{\kappa}{2} \|x^* - v_0\|^2 \right). \tag{26}$$

We note

$$\prod_{j=2}^{N} \frac{1}{A_{j-1}} = \frac{1}{\prod_{j=2}^{N} \left(1 - \frac{1}{(j+1)^2}\right)} \leq 2; \tag{27}$$

thereby concluding the result. Summing up (24) from $j = N+1$ to $2N$, we obtain

$$\begin{aligned}
\min_{j=1,\ldots,2N} \left\{\operatorname{dist}^2(0, \partial f(\bar{x}_j))\right\} &\leq \frac{4}{N} \sum_{j=N+1}^{2N} \|\kappa(\bar{x}_k - x_{k-1})\|^2) \\
&\leq \frac{8\kappa}{N} \left(\sum_{j=N+1}^{2N} f(x_{j-1}) - f(x_j) \right) \\
&\leq \frac{8\kappa}{N} \left(f(x_N) - f^*\right)
\end{aligned}$$

Combining this inequality with (26), the result is shown.

□

C Analysis of 4WD-Catalyst and Auto-adapt

Linear convergence interlude. Our assumption on the linear rate of convergence of $\mathcal{M}$ (see (14)) may look strange at first sight. Nevertheless, most linearly convergent first-order methods $\mathcal{M}$ for composite minimization either already satisfy this assumption or can be made to satisfy it by introducing an extra prox-gradient step. To see this, recall the convex composite minimization problem from Section A

$$\min_{z \in \mathbb{R}^p} \; h(z) := f_0(z) + \psi(z),$$

where

1. $f_0 \colon \mathbb{R}^p \to \mathbb{R}$ is convex and C^1-smooth with the gradient ∇f_0 that is L-Lipschitz,
2. $\psi \colon \mathbb{R}^p \to \overline{\mathbb{R}}$ is a closed convex function with a computable proximal map

$$\operatorname{prox}_{\beta\psi}(y) := \operatorname*{argmin}_z \; \{\psi(y) + \tfrac{1}{2\beta}\|z - y\|^2\}.$$

See [30] for a survey of proximal maps. Typical linear convergence guarantees of an optimization algorithm assert existence of constants $A \in \mathbb{R}$ and $\tau \in (0, 1)$ satisfying

$$h(z_t) - h^* \leq A(1 - \tau)^t (h(z_0) - h^*) \tag{28}$$

for each $t = 0, 1, 2, \ldots, \infty$. To bring such convergence guarantees into the desired form (14), define the prox-gradient step

$$[z]_L := \operatorname{prox}_{\psi/L}(z - L^{-1}\nabla f_0(z)),$$

and the displacement vector

$$g_L(z) = L(z - [z]_L),$$

and notice the inclusion $g_L(z) \in \partial h([z]_L)$. The following inequality follows from [29]:

$$\|g_L(z)\|^2 \leq 2L(h(z) - h([z]_L)) \leq 2L(h(z) - h^*).$$

Thus, the linear rate of convergence (28) implies

$$\|g_L(z_t)\|^2 \leq 2LA(1 - \tau)^t (h(z_0) - h^*),$$

which is exactly in the desired form (14).

C.1 Convergence analysis of the adaptive algorithm: 4WD-Catalyst

First, under some reasonable assumptions on the method $\mathcal{M}$ (see Section 4.1), the sub-method Auto-adapt terminates.

Lemma C.1 (Auto-adapt terminates)**.** *Assume that $\tau_\kappa \to 1$ when $\kappa \to +\infty$. The procedure Auto-adapt$(x, \kappa, \varepsilon, T)$ terminates after finitely many iterations.*

Proof. Due to our assumptions on $\mathcal{M}$ and the expressions $f_\kappa(x; x) = f(x)$ and $f_\kappa^*(x) \geq f^*$, we have

$$\operatorname{dist}^2\big(0, \partial f_\kappa(z_T; x)\big) \leq A(1-\tau_\kappa)^T\big(f(x) - f_\kappa^*(x)\big) \leq A(1-\tau_\kappa)^T\big(f(x) - f^*)\big). \tag{29}$$

Since τ_κ tends to one, for all sufficiency large κ, we can be sure that the right-hand-side is smaller than ε^2. On the other hand, for $\kappa > \rho$, the function $f_\kappa(\cdot; x)$ is $(\kappa - \rho)$-strongly convex and therefore we have $\operatorname{dist}^2(0, \partial f_\kappa(z_T; x)) \geq 2(\kappa-\rho)(f_\kappa(z_T; x) - f_\kappa^*(x))$. Combining this with (29), we deduce

$$f_\kappa(z_T; x) - f_\kappa^*(x) \leq \frac{A(1-\tau_\kappa)^T}{2(\kappa-\rho)}\big(f(x) - f_\kappa^*(x)\big).$$

Letting $\kappa \to \infty$, we deduce $f_\kappa(z_T; x) \leq f(x)$, as required. Thus the loop indeed terminates. □

We prove the main result, Theorem 4.4, for 4WD-Catalyst.

Proof of Theorem 4.4. The proof closely resembles the proofs of Theorem 3.2 and Theorem 3.2, so we omit some of the details. The main difference in the proof is that we keep track of the effects the parameters κ_{cvx} and κ_0 have on the inequalities as well as the sequence of κ_k. Since $\{f(x_k)\}_{k\geq 0}$ are monotonically decreasing, we deduce

$$f(x_{k-1}) = f_{\kappa_k}(x_{k-1}; x_{k-1}) \geq f_{\kappa_k}(\bar{x}_k; x_{k-1}) \geq f(x_k) + \frac{\kappa_k}{2}\left\|\bar{x}_k - x_{k-1}\right\|^2. \tag{30}$$

Using the adaptive stationary condition (18), we apply Lemma B.2 with $\varepsilon = \kappa_k \left\|\bar{x}_k - x_{k-1}\right\|$; hence we obtain

$$\operatorname{dist}(0, \partial f(\bar{x}_k)) \leq 2\left\|\kappa_k(\bar{x}_k - x_{k-1})\right\|.$$

We combine the above inequality with (30) to deduce

$$\operatorname{dist}^2(0, \partial f(\bar{x}_k)) \leq 4\left\|\kappa_k(\bar{x}_k - x_{k-1})\right\|^2 \leq 8\kappa_{\max}\left(f(x_{k-1}) - f(x_k)\right). \tag{31}$$

Summing $j = 1$ to N, we conclude

$$\begin{aligned}
\min_{j=1,\ldots,N}\ \left\{\operatorname{dist}^2(0, \partial f(\bar{x}_j))\right\} &\leq \frac{4}{N}\sum_{j=1}^{N} 2\left\|\kappa_k(\bar{x}_k - x_{k-1})\right\|^2) \\
&\leq \frac{8\kappa_{\max}}{N}\left(\sum_{j=1}^{N} f(x_{j-1}) - f(x_j)\right) \\
&\leq \frac{8\kappa_{\max}}{N}\left(f(x_0) - f^*\right).
\end{aligned}$$

Suppose the function f is convex. Using in the stopping criteria (17) in replacement of (11), we deduce a similar expression as (25):

$$\begin{aligned}
f(x_k) \leq \alpha_k f(x^*) + (1-\alpha_k) f(x_{k-1}) + \frac{\alpha_k^2\kappa_{\text{cvx}}}{2}\left(\left\|x^* - v_{k-1}\right\|^2 - \left\|x^* - v_k\right\|^2\right) \\
- \frac{\kappa_{\text{cvx}}}{2}\left\|\tilde{x}_k - y_k\right\|^2 + \frac{\alpha_k\kappa_{\text{cvx}}}{k+1}\left\|\tilde{x}_k - y_k\right\|\left\|x^* - v_k\right\|.
\end{aligned}$$

Denote $\theta_k = \frac{1}{k+1}$. Completing the square, we obtain

$$\frac{-\kappa_{\text{cvx}}}{2}\left\|\tilde{x}_k - y_k\right\|^2 + \alpha_k\theta_k\kappa_{\text{cvx}}\left\|\tilde{x}_k - y_k\right\|\left\|x^* - v_k\right\| \leq \frac{\kappa_{\text{cvx}}}{2}\left(\alpha_k\theta_k\right)^2\left\|x^* - v_k\right\|^2.$$

Hence, we deduce

$$\begin{aligned} f(x_k) - f^* &\leq (1-\alpha_k)(f(x_{k-1}) - f^*) + \frac{\alpha_k^2\kappa_{\text{cvx}}}{2}\left(\left\|x^* - v_{k-1}\right\|^2 - \left\|x^* - v_k\right\|^2\right) \\ &\quad + \frac{\kappa_{\text{cvx}}}{2}\left(\alpha_k\theta_k\right)^2\left\|x^* - v_k\right\|^2. \\ &= (1-\alpha_k)(f(x_{k-1}) - f^*) + \frac{\alpha_k^2\kappa_{\text{cvx}}}{2}\left(\left\|x^* - v_{k-1}\right\|^2 - \left(1-\theta_k^2\right)\left\|x^* - v_k\right\|^2\right) \end{aligned}$$

Denote $A_k := 1 - \theta_k^2$. Following the standard recursion argument as in the proofs of Theorem 3.2 and Theorem 3.2, we conclude

$$\begin{aligned} \frac{f(x_k) - f^*}{\alpha_k^2} + \frac{A_k\kappa_{\text{cvx}}}{2}\left\|x^* - v_k\right\|^2 &\leq \frac{1-\alpha_k}{\alpha_k^2}\left(f(x_{k-1}) - f^*\right) + \frac{\kappa_{\text{cvx}}}{2}\left\|x^* - v_{k-1}\right\|^2 \\ &\leq \frac{1}{A_{k-1}}\left(\frac{f(x_{k-1}) - f^*}{\alpha_{k-1}^2} + \frac{A_{k-1}\kappa_{\text{cvx}}}{2}\left\|x^* - v_{k-1}\right\|^2\right). \end{aligned}$$

The last inequality follows because $0 < A_{k-1} \leq 1$. Iterating N times, we deduce

$$\frac{f(x_N) - f^*}{\alpha_N^2} \leq \prod_{j=2}^{N}\frac{1}{A_{j-1}}\left(\frac{\kappa_{\text{cvx}}}{2}\left\|x^* - v_0\right\|^2\right). \tag{32}$$

We note

$$\prod_{j=2}^{N}\frac{1}{A_{j-1}} = \frac{1}{\prod_{j=2}^{N}\left(1 - \frac{1}{(j+1)^2}\right)} \leq 2;$$

thus the result is shown. Summing up (31) from $j = N+1$ to $2N$, we obtain

$$\begin{aligned} \min_{j=1,\ldots,2N}\left\{\text{dist}^2(0, \partial f(\bar{x}_j))\right\} &\leq \frac{4}{N}\sum_{j=N+1}^{2N}\left\|\kappa_k(\bar{x}_k - x_{k-1})\right\|^2) \\ &\leq \frac{8\kappa_{\max}}{N}\left(\sum_{j=N+1}^{2N} f(x_{j-1}) - f(x_j)\right) \\ &\leq \frac{8\kappa_{\max}}{N}\left(f(x_N) - f^*\right) \end{aligned}$$

Combining this inequality with (32), the result is shown. □

D Inner-loop complexity: proof of Theorem 4.5

Recall, the following notation

$$\begin{aligned} f_0(x; y) &= \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \frac{\kappa}{2}\left\|x - y\right\|^2 \\ y^0 &= \text{prox}_{1/(\kappa+L)f_0}\left(y - \frac{1}{\kappa+L}\nabla f_0(y; y)\right). \end{aligned} \tag{33}$$

Lemma D.1 (Relationship between function values and iterates of the prox)**.** *Assuming $\psi(x)$ is convex and the parameter $\kappa > \rho$, then*

$$f_\kappa(y^0; y) - f_\kappa^*(y) \leq \frac{\kappa + L}{2} \|y^* - y\|^2 \tag{34}$$

where y^* is a minima of $f_\kappa(\cdot; y)$ and $f_\kappa^*(y)$ is the optimal value.

Proof. As the κ is chosen sufficiently large, we know $f_0(\cdot; y)$ is convex and differentiable with $(\kappa+L)$-Lipschitz continuous gradient. Hence, we deduce for all x

$$f_0(y; y) + \nabla f_0(y; y)^T (x - y) \leq f_0(x; y). \tag{35}$$

Using the definition of y^0 and the $(\kappa + L)$-Lip. continuous gradient of $f_0(\cdot; y)$, we conclude for all x

$$\begin{aligned} f_\kappa(y^0; y) = f_0(y^0; y) + \psi(y^0) &\leq f_0(y; y) + \nabla f_0(y; y)^T (y_0 - y) + \frac{\kappa + L}{2} \|y_0 - y\|^2 + \psi(y_0) \\ &\leq f_0(y; y) + \nabla f_0(y; y)^T (x - y) + \frac{\kappa + L}{2} \|x - y\|^2 + \psi(x). \end{aligned} \tag{36}$$

By setting $x = y^*$ in both (35) and (36) and combining these results, we conclude

$$f_\kappa(y^0; y) \leq f_\kappa^*(y) + \frac{\kappa + L}{2} \|y^* - y\|^2 .$$

□

Note that if we are not in the composite setting and $\kappa > \rho$, then $f_\kappa(\cdot, y)$ is $(\kappa+L)$-strongly convex. Using standard bounds for strongly convex functions, Equation (34) follows (see [27]). We next show an important lemma for deducing the inner complexities.

Lemma D.2. *Assume $\kappa > \rho$. Given any $\varepsilon \leq \frac{\kappa-\rho}{2}$, if an iterate z satisfies dist$(0, \partial f_\kappa(z; y)) \leq \varepsilon \|y^* - y\|$, then*

$$\text{dist}(0, \partial f_\kappa(z; y)) \leq 2\varepsilon \|z - y\| . \tag{37}$$

Proof. Since $\kappa > \rho$, we know $f_\kappa(\cdot; y)$ is $(\kappa - \rho)$-strongly convex. Therefore, by [27], we know

$$\|z - y^*\| \leq \frac{1}{\kappa - \rho} \text{dist}(0, \partial f_\kappa(z; y)). \tag{38}$$

By the triangle inequality and Equation (38), we deduce

$$\begin{aligned} \text{dist}(0, \partial f_\kappa(z; y)) \leq \varepsilon \|y^* - y\| &\leq \varepsilon \big(\|y^* - z\| + \|z - y\| \big) \\ &\leq \frac{\varepsilon}{\kappa - \rho} \cdot \text{dist}(0, \partial f_\kappa(z; y)) + \varepsilon \|z - y\| \\ &\leq \frac{1}{2} \cdot \text{dist}(0, \partial f_\kappa(z; y)) + \varepsilon \|z - y\| . \end{aligned}$$

The last inequality follows because of the assumption $\varepsilon \leq \frac{\kappa-\rho}{2}$. Rearranging the terms above, we get the desired result. □

These two lemmas together give us Theorem 4.3.

Proof of Theorem 4.3. First, we prove that z_T satisfies both adaptive stationary condition and the descent condition. Recall, the point y^0 is defined to be the prox or y depending on if $f_\kappa(\cdot; y)$ is a composite form or smooth, respectively (see statement of Theorem 4.3). By Lemma D.1 (or the remark following it), the starting y^0 satisfies

$$f_\kappa(y^0; y) - f_\kappa^*(y) \leq \frac{\kappa + L}{2} \|y^* - y\|^2 .$$

By the linear convergence assumption of $\mathcal{M}$ (see (14)) and the above equation, after $T := T_\kappa$ iterations initializing from y^0, we have

$$\begin{aligned}
\operatorname{dist}^2(0, \partial f_\kappa(z_T; y)) &\leq A_\kappa (1-\tau_\kappa)^T \left(f_\kappa(y^0; y) - f_\kappa^*(y)\right) \\
&\leq A_\kappa e^{-T\cdot\tau_\kappa} \left(f_\kappa(y^0; y) - f_\kappa^*(y)\right) \\
&\leq \frac{(\kappa-\rho)^2}{8(L+\kappa)} \cdot \frac{L+\kappa}{2} \|y^* - y\|^2 \\
&\leq \frac{(\kappa-\rho)^2}{16} \|y^* - y\|^2 .
\end{aligned} \tag{39}$$

Take the square root and apply Lemma D.2 yields

$$\operatorname{dist}(0, \partial f_\kappa(z_T; y)) \leq \frac{\kappa-\rho}{2} \|z_T - y\| \leq \kappa \|z_T - y\| ,$$

which gives the adaptive stationary condition. Next, we show the descent condition. Let $v \in \partial f_\kappa(z_T; y)$ such that $\|v\| \leq (\kappa-\rho)\|z_T - y\|/2$, by the $(\kappa-\rho)$-strong convexity of $f_\kappa(\cdot; y)$, we deduce

$$\begin{aligned}
f_\kappa(y; y) &\geq f_\kappa(z_T; y) + \langle v, y - z_T\rangle + \frac{\kappa-\rho}{2}\|z_T - y\|^2 \\
&\geq f_\kappa(z_T; y) - \|v\|\,\|y - z_T\| + \frac{\kappa-\rho}{2}\|z_T - y\|^2 \\
&\geq f_\kappa(z_T; y).
\end{aligned}$$

This yields the descent condition which completes the proof for T. The proof for S_κ is similar to T_κ, so we omit many of the details. In this case, we only need to show the adaptive stationary condition. For convenience, we denote $S = S_\kappa$. Following the same argument as in Equation (39) but with $S\log(k+1)$ number of iterations, we deduce

$$\operatorname{dist}^2(0, \partial f_\kappa(z_S; y)) \leq \frac{(\kappa-\rho)^2}{16(k+1)^2} \|y^* - y\|^2 .$$

By applying Lemma D.2, we obtain

$$\operatorname{dist}(0, \partial f_\kappa(z_S; y)) \leq \frac{(\kappa-\rho)}{2(k+1)} \|z_T - y\| \leq \frac{\kappa}{k+1} \|z_S - y\| ,$$

which proves the desired result for z_S. □

Assuming Proposition 4.6 and Proposition 4.7 hold as well as Lemma D.3, we begin by providing the proof of Theorem 4.5.

Proof of Theorem 4.5. We consider two cases: (i) the function f is non-convex and (ii) the function f is convex. First, we consider the non-convex setting. To produce $\bar{x}_k$, the method $\mathcal{M}$ is called

$$T \log\left(\tfrac{4L}{\kappa_0}\right) / \log(2) \tag{40}$$

number of times. This follows from Proposition 4.6 and Lemma D.3. The reasoning is that once $\kappa > \rho + L$, which only takes at most $\log(4L/\kappa_0)$ number of increases of κ to reach, then the iterate $\bar{x}_k$ satisfies the stopping criteria (18). Each time we increase κ we run $\mathcal{M}$ for T iterations. Therefore, the total number of iterations of $\mathcal{M}$ is given by multiplying T with $\log(4L/\kappa_0)$. To produce $\tilde{x}_k$, the method $\mathcal{M}$ is called $S\log(k+1)$ number of times. (Note: the proof of Theorem 4.4 does not need $\tilde{x}_k$ to satisfy (17) in the non-convex case).

Next, suppose the function f is convex. As before, to produce $\bar{x}_k$ the method $\mathcal{M}$ is called (40) times. To produce $\tilde{x}_k$, the method $\mathcal{M}$ is called $S\log(k+1)$ number of times. By Proposition 4.7, the iterate $\tilde{x}_k$ satisfies (17); a key ingredient in the proof of Theorem 4.4. □

D.1 Inner complexity for $\bar{x}_k$: proof of Proposition 4.6

Next, we supply the proof of Proposition 4.6 which shows that by choosing κ large enough, Algorithm 3 terminates.

Proof of Proposition 4.6. The idea is to apply Theorem 4.3. Since the parameter A_κ increases with κ, then we upper bound it by $A_{\kappa_k} \leq A_{4L}$. Moreover, we have $\kappa - \rho \geq \rho + L - \rho = L$. Lastly, since τ_κ is increasing in κ, we know $\frac{1}{\tau_\kappa} \leq \frac{1}{\tau_L}$. Plugging these bound into Theorem 4.3, we see that for any smoothing parameter κ satisfying $\rho + L < \kappa < 4L$, we get the desired result. □

Next, we compute the maximum number of times we must double κ until $\kappa > \rho + L$.

Lemma D.3 (Doubling κ)**.** *If we set T and S according to Theorem 4.5, then the doubling of κ_0 will terminate as soon as $\kappa > \rho + L$. Thus the number of times κ_0 must be doubled in Algorithm 3 is at most*

$$\frac{\log\left(\frac{2(\rho+L)}{\kappa_0}\right)}{\log(2)} \leq \left\lceil \frac{\log\left(\frac{4L}{\kappa_0}\right)}{\log(2)} \right\rceil.$$

Since κ is doubled (Algorithm 3) and T is chosen as in Proposition 4.6 , the maximum the value κ, $\kappa_{\max}$, takes is $2(\rho + L) \leq 4L$.

D.2 Inner complexity for $\tilde{x}_k$: proof of Proposition 4.7

In this section, we prove Proposition 4.7, an inner complexity result for the iterates $\tilde{x}_k$. Recall that the inner-complexity analysis for $\tilde{x}_k$ is important only when f is convex (see Section 4). Therefore, we assume throughout this section that the function f is convex. We are now ready to prove Proposition 4.7.

Proof of Proposition 4.7. The proof immediately follows from Theorem 4.3 by setting $\kappa = \kappa_{\text{cvx}}$ and $\rho = 0$ as the function f is convex. □

References

[1] Z. Allen-Zhu. Natasha: Faster stochastic non-convex optimization via strongly non-convex parameter. *arXiv:1702.00763*, 2016.

[2] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[3] D. P. Bertsekas. *Nonlinear programming.* Athena scientific Belmont, 1999.

[4] D. P. Bertsekas. *Convex Optimization Algorithms.* Athena Scientific, 2015.

[5] J. Bolte, T.P. Nguyen, J. Peypouquet, and B. Suter. From error bounds to the complexity of first-order descent methods for convex functions. *Mathematical Programming, Serie A*, 2016.

[6] J.M. Borwein and A.S. Lewis. *Convex analysis and nonlinear optimization: Theory and examples.* Springer Verlag, 2006.

[7] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Accelerated methods for non-convex optimization. *arXiv:1611.00756*, 2016.

[8] Y. Carmon, O. Hinder, J. C. Duchi, and A. Sidford. "Convex until proven guilty": Dimension-free acceleration of gradient descent on non-convex functions. *preprint arXiv:1705.02766*, 2017.

[9] C. Cartis, N. I. M. Gould, and P. L. Toint. On the complexity of finding first-order critical points in constrained nonlinear optimization. *Mathematical Programming*, 2014.

[10] C. Cartis, N. I. M. Gould, and Ph. L. Toint. On the complexity of steepest descent, newton's and regularized newton's methods for nonconvex unconstrained optimization problems. *SIAM Journal on Optimization*, 20(6):2833–2852, 2010.

[11] F. H. Clarke, R. J. Stern, and P. R. Wolenski. Proximal smoothness and the lower-C^2 property. *Journal of Convex Analysis*, 2(1-2):117–144, 1995.

[12] A. J. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[13] D. Drusvyatskiy and C. Paquette. Efficiency of minimizing compositions of convex functions and smooth maps. *Preprint arXiv:1605.00125*, 2016.

[14] H. Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93:418–491, 1959.

[15] S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2, Ser. A):59–99, 2016.

[16] S. Ghadimi, G. Lan, and H. Zhang. Generalized uniformly optimal methods for nonlinear programming. *arXiv:1508.07384*, 2015.

[17] O. Güler. On the convergence of the proximal point algorithm for convex minimization. *SIAM Journal on Control and Optimization*, 29(2):403–419, 1991.

[18] T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical Learning With Sparsity: The Lasso And Generalizations*. CRC Press, 2015.

[19] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[20] G. Lan. An optimal randomized incremental gradient method. *arXiv:1507.02000*, 2015.

[21] H. Li and Z. Lin. Accelerated proximal gradient methods for nonconvex programming. In *Advances in Neural Information Processing Systems (NIPS)*. 2015.

[22] H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for first-order optimization. *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[23] J. Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

[24] J. Mairal, F. Bach, and J. Ponce. Sparse modeling for image and vision processing. *Foundations and Trends in Computer Graphics and Vision*, 8(2-3):85–283, 2014.

[25] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research (JMLR)*, 11:19–60, 2010.

[26] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[27] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2004.

[28] Y. Nesterov. How to make the gradients small. *OPTIMA, MPS Newsletter*, (88):10–11, 2012.

[29] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[30] N. Parikh and S.P. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2014.

[31] R. A. Poliquin and R. T. Rockafellar. Prox-regular functions in variational analysis. *Transactions of the American Mathematical Society*, 348(5):1805–1838, 1996.

[32] S. J. Reddi, S. Sra, B. Poczos, and A. J. Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[33] R. T. Rockafellar. Favorable classes of Lipschitz-continuous functions in subgradient optimization. In *Progress in nondifferentiable optimization*, volume 8 of *IIASA Collaborative Proc. Ser. CP-82*, pages 125–143. Internat. Inst. Appl. Systems Anal., Laxenburg, 1982.

[34] R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*, volume 317 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1998.

[35] M. Schmidt, N. Le Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1):83–112, 2017.

[36] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. Technical report, 2008.

[37] B. E. Woodworth and N. Srebro. Tight complexity bounds for optimizing composite objectives. In *Advances in Neural Information Processing Systems (NIPS)*. 2016.

[38] L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

[39] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.