A characterization of linearly convergent algorithms in convex and composite optimization

Andrea Martin, Ian R. Manchester, and Luca Furieri

***Abstract*—We consider the problem of designing iterative optimization algorithms whose performance is targeted at specific classes of composite optimization problems while inheriting the worst-case convergence guarantees of a general-purpose legacy algorithm. Towards this goal, we characterize the class of iterative optimization algorithms that achieve linear convergence for classes of nonsmooth composite optimization problems. This is achieved by starting from any algorithm known to converge linearly and deriving all – and only – the modifications to its update rule that preserve this convergence rate. Our results apply to evolving legacy algorithms such as gradient descent for nonconvex, gradient-dominated functions; Nesterov's accelerated method for strongly convex functions; projected methods for polytopic-constrained optimization; and the alternating direction method of multipliers (ADMM) for distributed convex and composite optimization. Beyond the theoretical scope, these results find direct application in learned optimization: we show how to learn over classes of exponentially decaying update rules to enhance the empirical performance of classical, linearly convergent optimizers over optimization problems of interest.**

I. Introduction

Guarantees of fast convergence are crucial whenever optimization must be executed under tight computational budgets, as in large-scale machine learning or real-time model predictive control (MPC). Worst-case linear convergence guarantees have been developed for iterative optimization algorithms over several classes of objective functions, whose structure – e.g., strong convexity, smoothness, or gradient dominance – can be exploited by first-order schemes such as standard gradient descent and Nesterov's accelerated method [1]. A growing body of work leverages the analogy between worst-case convergence rates and robust-control techniques such as integral quadratic constraints (IQCs), leading to characterizations of accelerated algorithms with provably optimal rates across families of convex functions [2]–[4].

Worst-case rate guarantees are crucial, as they establish a baseline performance in terms of the number of iterations required to achieve a certain level of precision. However, how well an algorithm performs in a specific application does not depend solely on its worst-case convergence rate. First, there exist fundamental trade-offs between the speed of convergence and the robustness of algorithmic behaviour; see, for instance, the speed/covariance trade-off for accelerated methods in strongly convex optimization analysed in [5]. This raises the question of how to appropriately define algorithm performance. A second challenge is that scenarios encountered in applications rarely span the entirety of the space of problem instances for which the worst-case guarantee is tight, resulting in overly conservative average performance. This introduces another trade-off: how to tailor performance to these specific instances without compromising the original uniform guarantees. A prime example of such situation is MPC [6], where the dynamics and cost are fixed, and only the initial state varies. In such cases, a solver tailored to this sub-family of problems could converge in significantly fewer iterations without compromising worst-case guarantees.

The learning to optimize (L2O) literature addresses the challenge of adopting user-defined performance metrics beyond mere convergence rates and designs algorithms that are tailored to such metrics using learning algorithms. For instance, [7] proposes an algorithm performance metric that balances convergence speed with solution precision and learns neural network update rules. However, general-purpose neural network update rules come with no guarantees. Convergence with learned updates has been addressed through conservative safeguarding mechanisms [8], or by exploiting an ML component for optimal tuning of parameters, such as learning initializations of classical algorithms [9], or tuning the hyperparameters of ADMM for accelerated quadratic optimization via reinforcement learning [10]. These approaches demonstrate performance exceeding that of state-of-the-art classical algorithms upon training and inherit their convergence guarantees – at the cost of restricting the learning-based design to parameter tuning.

Beyond the optimal tuning of classical algorithms, another line of research seeks to use machine learning to design entirely new convergent update rules, aiming to discover application-specific shortcuts unknown to classical update rules. This has been achieved by taking simple gradient descent as a baseline and enhancing it through learned optimal deviations from such gradient-based updates. The work [11] characterizes the class of all and only those deviation functions that ensure convergence to stationary points in nonconvex, unconstrained smooth optimization, enabling learned optimization for user-defined performance metrics and outperforming optimally tuned ADAM [12] in neural network training. The work [13] uses deep learning to train deviations from gradient descent and saturates these updates with the norm of measured gradients, ensuring convergence for composite convex optimization. The numerical studies of [11], [13] empirically demonstrate that convergence rates superior to

This manuscript is a preliminary version. The numerical results presented during the European Control Conference 2025 in Thessaloniki will appear in the final preprint.

A. Martin is with the Division of Decision and Control Systems of the Kungliga Tekniska Högskolan (KTH), and he is grateful to Digital Futures for the Postdoc Fellowship funding. E-mail: andrmar@kth.se

I. R. Manchester is with Australian Centre for Robotics and School of Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, Australia. E-mail: ian.manchester@sydney.edu.au.

L. Furieri is with the Department of Engineering Sciences of the University of Oxford, and he is grateful to the Swiss National Science Foundation (SNSF) for the grant PZ00P2_208951. E-mail: luca.furieri@eng.ox.ac.uk

those of classical algorithms can be achieved through training on gradient descent deviations. However, there is no theoretical guarantee that this improvement will always occur.

Motivated as outlined above, in this paper, we characterize the class of all linearly convergent algorithms in convex and composite optimization. One of our main goals is to address a question that has remained open in the literature of learned optimization. Given any state-of-the-art algorithm for solving a class of optimization problems – such as the optimally tuned Nesterov method for strongly convex smooth optimization – how can we improve its average performance over a class of instances of interest without sacrificing its convergence rate over the entire class of problems?

We believe that a theoretical study of these trade-offs is fundamental to making learned optimization a standard and reliable component of algorithm design.

Contributions: Given any existing optimization algorithm that achieves linear convergence to fixed points at a specified rate – henceforth the *base algorithm* – our main contributions are as follows. First, we characterize the conditions on the base algorithm under which adding exponentially decaying perturbations preserves the same linear convergence rate, up to a higher-order polynomial term. Second, we relax these conditions and identify fundamental trade-offs between the frequency of perturbations and their worst-case impact on the linear convergence rate. Third, we establish a completeness result for linearly convergent optimization: every update rule that converges linearly at a given rate can be written as the sum of the base algorithm and a suitably designed exponentially decaying perturbation function. Finally, we characterize the classes of all linearly convergent update rules in the context of classical algorithms adapted to various smooth and nonsmooth optimization settings. Notably, these include:

1) Gradient descent for nonconvex Polyak–Łojasiewicz (PL) functions;
2) Accelerated methods for strongly convex optimization;
3) Proximal gradient methods for convex optimization with polytopic constraints;
4) ADMM for distributed convex and strongly convex optimization.

Notation: The set of all sequences $\mathbf{x} = (x_0, x_1, x_2, \dots)$ where $x_t \in \mathbb{R}^n$ for all $t \in \mathbb{N}$ is denoted as ℓ^n. For $\mathbf{x} \in \ell^n$, we denote by $z\mathbf{x} = (x_1, x_2, x_3, \dots)$ the sequence shifted one time-step forward. Moreover, $\mathbf{x}$ belongs to $\ell_2^n \subset \ell^n$ if $\|\mathbf{x}\|_2 = \sqrt{\sum_{t=0}^{\infty} |x_t|^2} < \infty$, where $|\cdot|$ denotes any vector norm. When clear from the context, we omit the superscript n from ℓ^n and ℓ_2^n. For a function $g : \mathbb{R}^n \to \mathbb{R}^m$, we write $g(\mathbf{x}) = (g(x_0), g(x_1), \dots) \in \ell^m$. For $m \in \mathbb{N}$, we use $\mathcal{P}_m(x)$ to denote the set of positive and monotonically non-decreasing polynomials of degree at most m in the variable x. We define the set of fixed points of an operator π, assumed non-empty, as $\texttt{Fix}_\pi$. For $m \in \mathbb{N}$ and $\gamma \in (0,1)$, we denote by $\ell_{exp}(m, \gamma)$ the class of signals $\mathbf{x}$ for which there exists a polynomial $p_m(t) \in \mathcal{P}_m(t)$ such that $|x_t| \leq p_m(t)\gamma^t$.

II. Problem Formulation

We consider composite optimization problems of the form

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

where $x \in \mathbb{R}^d$ is the decision variable, and the objective function is such that $f : \mathbb{R}^d \to \mathbb{R}$ is proper and β-smooth, and $g : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is convex, proper, and lower semi-continuous, but potentially nonsmooth. We let $F(x) = f(x) + g(x)$ for brevity, and we assume that the set of optimizers $\mathcal{X}^\star = \arg\min_{x \in \mathbb{R}^d} F(x)$ is non-empty. In particular, we note that (1) subsumes constrained optimization problems of the form

$$\min_{x \in \mathbb{R}^d} \quad f_0(x) \tag{2a}$$

$$\text{subject to} \quad f_i(x) \leq 0\,, \quad \forall i \in [1, M]\,, \tag{2b}$$

where $f_0 : \mathbb{R}^d \to \mathbb{R}$ is β-smooth, and each function $f_i : \mathbb{R}^d \to \mathbb{R}$ with $i \in [1, M]$ defines a non-empty convex feasibility set $\mathcal{X}_i \subseteq \mathbb{R}^d$. In fact, one can rewrite (2) as an instance of (1) letting $f(x) = f_0(x)$ and $g(x) = \max_{i \in [1,M]} \mathbb{I}_{\mathcal{X}_i}(x)$.

A standard method to solve problem (1) is to analytically construct iterations of the form:

$$\xi_{t+1} = \pi(F, \xi_t)\,, \quad \xi_0 \in \mathbb{R}^n\,, \quad t \in \mathbb{N}\,, \tag{3}$$

where $\xi_t \in \mathbb{R}^n$, and the operator π is designed so that its set of fixed points $\texttt{Fix}_\pi$, that is, the points $\xi^\star$ such that $\pi(F, \xi^\star) = \xi^\star$ is closely related to $\mathcal{X}^\star$; that is, either $\mathcal{X}^\star \equiv \texttt{Fix}_\pi$ or a point $x^\star \in \mathcal{X}^\star$ can be easily reconstructed from a point $\xi^\star \in \texttt{Fix}_\pi$.

A key metric for the performance of algorithms (3) when applied to a class of problems $F \in \mathcal{F}$ is how fast they converge to $\texttt{Fix}_\pi$. Classical optimization algorithms often come with convergence guarantees that hold for the worst-case instance of $F \in \mathcal{F}$. However, optimal control methods such as MPC require efficiently finding solutions to the instances of (1) that are encountered during deployment, where the objective F is drawn from a specific distribution $\mathbb{D}_\mathcal{F}$ over the class $\mathcal{F}$. Motivated as such, in this work we investigate the following question.

Given a legacy algorithm π to solve (1)*, how can we improve its average performance on specific functions $F \sim \mathbb{D}_\mathcal{F}$, while retaining worst-case convergence guarantees over the entire class $\mathcal{F}$?*

In particular, this paper characterizes algorithms $\boldsymbol{\nu}$ that achieve *linear convergence* to $\texttt{Fix}_\pi$ for classes of functions $F \in \mathcal{F}$.

Definition 1: An algorithm $\xi_{t+1} = \nu_t(F, \xi_{t:0})$ is linearly convergent to $\texttt{Fix}_\pi$ for $\mathcal{F}$ with rate $\gamma \in (0,1)$ if and only if there exists a polynomial $p(t) \in \mathcal{P}_m(t)$ such that

$$\operatorname{dist}(\xi_t, \texttt{Fix}_\pi) \leq p(t)\gamma^t \operatorname{dist}(\xi_0, \texttt{Fix}_\pi)\,,\ \forall \xi_0 \in \mathbb{R}^n\,,\ \forall F \in \mathcal{F}\,, \tag{4}$$

at all times, where $\operatorname{dist}(\cdot,\cdot)$ is a distance function. In this case, we write that $\boldsymbol{\nu} \in \texttt{pExp}^\pi_\mathcal{F}(m, \gamma)$. When the focus is not on the polynomial order and only on the exponential convergence rate, we write $\boldsymbol{\nu} \in \widehat{\texttt{pExp}}^\pi_\mathcal{F}(\gamma)$. Additionally, if (4) holds with a constant polynomial $p(t) = 1$, then we say that $\boldsymbol{\nu}$ is monotonically linearly convergent to $\texttt{Fix}_\pi$ and we write $\boldsymbol{\nu} \in \texttt{Exp}^\pi_\mathcal{F}(\gamma)$.

To enhance the average performance of a base algorithm π on problem instances $F \sim \mathbb{D}_{\mathcal{F}}$, we aim to design algorithm updates $v_t \in \mathbb{R}^n$ that do not jeopardize its convergence guarantees. Specifically, we consider the evolved algorithm defined by the iterations.

$$\xi_{t+1} = \nu_t(F, \xi_{t:0}) = \pi(F, \xi_t) + v_t(F, \xi_{t:0}), \ \xi_0 \in \mathbb{R}^n, \quad (5)$$

and we establish conditions ensuring that such evolved algorithm $\boldsymbol{\nu}$ is linearly convergent as per (4).

III. Main Results

In this section, we establish our main results on how introducing an evolution term v_t affects the worst-case linear convergence guarantees of a given base optimization algorithm π. We first abstract away from the specific form of π and the class of functions $\mathcal{F}$ it is designed to optimize; we only assume that π is a linearly convergent fixed-point algorithm as per Definition 1. In Section III-B, we present several classes of problems (1) and corresponding base algorithms π that are compatible with our framework, making them amenable to learning-based enhancement of average performance.

A. Characterizations of linearly convergent algorithms and their completeness under (5)

Consider a base algorithm π that achieves linear convergence as per Definition (1). The property (4) implies that the signal $\operatorname{dist}(\xi_t, \texttt{Fix}_\pi)$ decays exponentially up to a polynomial factor $p(t)$, for any initial condition ξ_0 and any objective $F \in \mathcal{F}$. We first characterize to what extent injecting exponentially decaying signals v_t in the iterates of (5) can deteriorate the convergence guarantee of π.

Theorem 1: Consider the recursion (5) and assume that $\pi \in \texttt{pExp}^{\pi}_{\mathcal{F}}(m, \gamma)$. Choose any $N \in \mathbb{N}$ such that $\rho = p(N)\gamma^N < 1$ and any auxiliary signal $\mathbf{w} \in \ell_{exp}(m, \rho)$. For every $t \in \mathbb{N}$ construct the evolution signal v_t in (5) as follows:

$$v_t = \begin{cases} w_{\frac{t+1}{N}-1} & \text{if } t+1 \bmod N = 0, \\ 0 & \text{otherwise}. \end{cases} \quad (6)$$

Then, the iterates of (5) satisfy:

$$\operatorname{dist}(\boldsymbol{\xi}, \texttt{Fix}_\pi) \in \ell_{exp}(m+1, \sqrt[N]{p(N)}\gamma). \quad (7)$$

We report the proof of Theorem 1 in Appendix A. Theorem 1 establishes a trade-off between how often we inject an exponentially decaying perturbation – as measured by $N \in \mathbb{N}$ – and the degradation of the convergence rate. In particular, when $N = 1$, we observe that the asymptotic rate γ does not change, as only the order of the polynomial factor in (4) is affected. For the general case where $N > 1$, the convergence rate increases at most to the value $\sqrt[N]{p(N)}\gamma \in (\gamma, 1)$. As expected, when N tends to infinity, we recover the original rate of the base algorithm because $\lim_{N\to\infty} \sqrt[N]{p(N)} = 1$; this corresponds to the case $v_t = 0$ for all t.

A first challenge is that more frequent learned updates of the base algorithm require a lower value for N, resulting in a deteriorated convergence rate according to (7), that is $\boldsymbol{\nu} \notin \widehat{\texttt{pExp}}^{\pi}_{\mathcal{F}}(\gamma)$. Second, it is crucial to understand how large is the class of linearly convergent algorithms $\xi_t = \nu_t(F, \xi_{t:0})$ that can be achieved by perturbing a base algorithm π with an exponentially decaying learned update v_t as per (5). Our next result establishes conditions on the base algorithm π that simultaneously address the two concerns above. First, we ensure that the evolved algorithm $\boldsymbol{\nu}$ lies in $\widehat{\texttt{pExp}}^{\pi}_{\mathcal{F}}(\gamma)$ for any $\mathbf{v}$ that exponentially decays with rate γ. Second, we guarantee that *any* algorithm in $\texttt{pExp}^{\pi}_{\mathcal{F}}(m, \gamma)$ can be represented – provided that such target algorithm satisfies the following regularity condition.

Definition 2: Define the sequence of updates $u_t = \xi_{t+1} - \xi_t$ associated with a linearly convergent algorithm $\boldsymbol{\nu} \in \texttt{pExp}^{\pi}_{\mathcal{F}}(m, \gamma)$. We say that $\boldsymbol{\nu}$ is *regular*, if and only if the sequence of updates vanishes with the same exponential rate, that is,

$$\mathbf{u} = z\boldsymbol{\xi} - \boldsymbol{\xi} \in \ell_{exp}(m, \gamma).$$

In other words, the definition above excludes pathological cases of linearly convergent algorithms that can cycle indefinitely among the points in $\texttt{Fix}_\pi$ even when $\operatorname{dist}(\xi_t, \texttt{Fix}_\pi) = 0$. We are ready to present our completeness result.

Theorem 2: Let $\pi \in \texttt{Exp}^{\pi}_{\mathcal{F}}(\gamma)$ be a base algorithm such that $\pi(F, \xi)$ is Lipschitz continuous in ξ. Consider the evolved algorithm $\boldsymbol{\nu}$ with iterates ξ_t defined as per (5), and any target algorithm $\chi_{t+1} = \sigma_t(F, \chi_{t:0})$ such that $\boldsymbol{\sigma} \in \texttt{pExp}^{\pi}_{\mathcal{F}}(m, \gamma)$. If $\boldsymbol{\sigma}$ is regular, there exists a sequence $\mathbf{v}(F, \boldsymbol{\xi}) \in \ell_{exp}(m, \gamma)$ such that the iterations of $\boldsymbol{\nu}$ initialized with $\xi_0 = \chi_0$ are equivalent to those of $\boldsymbol{\sigma}$. Additionally, the evolved algorithm $\boldsymbol{\nu}$ belongs to $\widehat{\texttt{pExp}}^{\pi}_{\mathcal{F}}(\gamma)$ for any $\mathbf{v} \in \ell_{exp}(m, \gamma)$ with $m \in \mathbb{N}$.

A few remarks are in order. First, the completeness property of Theorem 2 is key in the context of automating the design of evolved algorithms, as it implies that (5) encompasses all linearly convergent algorithms in $\texttt{pExp}^{\pi}_{\mathcal{F}}(m, \gamma)$. Second, when we learn an evolution term $\mathbf{v} \in \ell_{exp}(m, \gamma)$ by searching over the entire space of exponentially decaying updates, it is crucial that the base algorithm satisfy the stronger condition $\pi \in \texttt{Exp}^{\pi}_{\mathcal{F}}(\gamma)$. If instead $\pi \in \texttt{pExp}^{\pi}_{\mathcal{F}}(m, \gamma\ ) \setminus \texttt{Exp}^{\pi}_{\mathcal{F}}(\gamma)$, then Theorem 1 only guarantees linear convergence – with the degraded rate $\sqrt[N]{p(N)}\,\gamma$ – for those $\mathbf{v}$ chosen exactly as in (6). In other words, without $\pi \in \texttt{Exp}^{\pi}_{\mathcal{F}}(\gamma)$, most perturbations in $\ell_{exp}(m, \gamma)$ would not preserve linear convergence, significantly limiting the designer ability to freely explore the space of updates. Third, the assumption that $\pi(F, \xi)$ is Lipschitz continuous in ξ is mild; we will show in the next section that this condition is satisfied for important base algorithms widely used for convex and composite optimization.

B. Application to convex and composite optimization

Here, we show how Theorem 1 and Theorem 2 can be used to evolve existing solvers for convex and composite optimization problems in the form (1) drawn from specific classes $\mathcal{F}$.

The case of smooth optimization: We first consider the case (1) where $g(x) = 0$ for all $x \in \mathbb{R}^d$, leaving us with the task of minimizing a β-smooth function $F(x) = f(x)$. Our first result focuses on classes of possibly nonconvex functions for which standard gradient descent achieves monotonic linear convergence.

Corollary 1: Let $\mathcal{F}_{RSI}^{\beta,\mu}$ be the class of β-smooth functions satisfying the restricted secant inequality (RSI) with constant $\mu > 0$, that is, those for which it holds

$$\nabla F(x)^\top (x - x^\star) \geq \frac{\mu}{2} \operatorname{dist}(x, \mathcal{X}^\star)^2 \,, \ \forall x \in \mathbb{R}^d \,, \qquad (8)$$

for any $x^\star$ in $\arg\min_{y \in \mathcal{X}^\star} \operatorname{dist}(x, y)^2$. Let π be the gradient descent update rule $\pi(F, \xi_t) = \xi_t - \eta \nabla F(\xi_t)$ with $\eta = \frac{\mu}{\beta^2}$, and $\gamma = \sqrt{1 - \frac{\mu^2}{\beta^2}}$. Then, any regular algorithm $\boldsymbol{\sigma} \in \mathtt{pExp}^{\pi}_{\mathcal{F}_{RSI}^{\beta,\mu}}(m, \gamma)$ can be written as

$$x_{t+1} = \nu_t(F, x_{t:0}) = x_t - \eta \nabla F(x_t) + v_t(F, x_{t:0}) \,, \qquad (9)$$

with $\mathbf{v} \in \ell_{exp}(m, \gamma)$. Viceversa, for any $\mathbf{v} \in \ell_{exp}(m, \gamma)$, the algorithm (9) is such that $\boldsymbol{\nu} \in \widehat{\mathtt{pExp}}^{\pi}_{\mathcal{F}_{RSI}^{\beta,\mu}}(\gamma)$.

Proof: By Theorem 2.1 of [14], it holds that (4) holds for the gradient descent algorithm $\xi_{t+1} = \pi(F, \xi_t) = \xi_t - \eta \nabla F(\xi_t)$ with $\eta = \frac{\mu}{\beta^2}$ with $\gamma = \sqrt{1 - \frac{\mu^2}{\beta^2}} \in (0, 1)$. Further, we have that $\pi(F, \xi_t)$ is Lipschitz continuous since $|\pi(F, x) - \pi(F, y)| = |x - y - \eta \nabla F(x) + \eta \nabla F(y)| \leq (1 + \eta\beta)|x - y|$, where the last inequality follows from the β-smoothness of $F \in \mathcal{F}_{RSI}^{\beta,\mu}$. The result then follows by applying Theorem 2.

The result of Corollary 1 enables learning over the class of *all* the linearly convergent regular algorithms in $\mathtt{pExp}^{\pi}_{\mathcal{F}_{RSI}^{\beta,\mu}}(m, \gamma)$, while ensuring that the evolved algorithm (9) never leaves the class $\widehat{\mathtt{pExp}}^{\pi}_{\mathcal{F}_{RSI}^{\beta,\mu}}(\gamma)$, irrespectively of how "badly" the enhancement term $\mathbf{v} \in \ell_{exp}(m, \gamma)$ may be chosen.

A few comments regarding the generality of the class of functions $\mathcal{F}_{RSI}^{\beta,\mu}$ are in order. First, $\mathcal{F}_{RSI}$ encompasses certain nonconvex functions, as highlighted in [15]. Second, it holds that $\mathcal{F}_{SC}^{\beta,\mu} \subset \mathcal{F}_{cPL}^{\beta,\mu} \subset \mathcal{F}_{RSI}^{\beta,\mu}$, where $\mathcal{F}_{SC}^{\beta,\mu}$ is the set of β-smooth and strongly convex functions complying with

$$F(y) \geq F(x) + \nabla F(x)^\top (y - x) + \frac{\mu}{2}|y - x|^2 \,, \qquad (10)$$

for some $\mu > 0$, and $\mathcal{F}_{cPL}^{\beta,\mu}$ is the set of all the β-smooth and convex functions that comply with the Polyak–Łojasiewicz (PL) inequality

$$F(x) - \min_{x \in \mathbb{R}^d} F(x) \leq \frac{1}{2\mu} |\nabla F(x)|^2 \,, \qquad (11)$$

for some $\mu > 0$.

Remark 1: It is well known that $\mathcal{F}_{RSI}^{\beta,\mu} \subseteq \mathcal{F}_{PL}^{\beta, \frac{\mu^2}{4\beta}}$, where $\mathcal{F}_{PL}^{\beta,\mu}$ is the set of all possibly nonconvex functions satisfying (11), see [16]. For functions in $\mathcal{F}_{PL}^{\beta,\mu}$, the gradient descent rule $\pi(F, x) = -\frac{1}{\beta}\nabla F(x)$ achieves linear convergence in the function value as per

$$F(x_t) - F^\star \leq \left(1 - \frac{\mu}{\beta}\right)^t (F(x_0) - F^\star) \,.$$

However, π induces a monotonically linearly convergent sequence of iterates only if the restricted secant inequality (8) also holds, see [14].

Corollary 1 ensures a complete parametrization of linearly convergent regular algorithms with the same rate γ as gradient descent for all functions in $\mathcal{F}_{RSI}^{\beta,\mu}$. For the special case $F \in \mathcal{F}_{SC}^{\beta,\mu}$, one typically wants to evolve ad-hoc algorithms tailored to $\mathcal{F}_{SC}^{\beta,\mu}$ such as Nesterov's accelerated gradient (NAG) [1], the Heavy-Ball method [17], [18], or optimal-rate algorithms such as those characterized in [2], [19].

Motivated as such, we show compatibility of the proposed framework with the evolution of accelerated algorithms for objectives $F \in \mathcal{F}_{SC}^{\beta,\mu}$.

Corollary 2: Consider the NAG algorithm

$$\pi(F, \xi_t) = \begin{bmatrix} 1 + \alpha & -\alpha \\ 1 & 0 \end{bmatrix} \xi_t + \begin{bmatrix} -\eta \\ 0 \end{bmatrix} \nabla F\left(\begin{bmatrix} 1 + \alpha & -\alpha \end{bmatrix} \xi_t\right) \,, \qquad (12)$$

where $\xi_t = \begin{bmatrix} x_t^\top & x_{t-1}^\top \end{bmatrix}^\top$ and $\alpha \geq 0$ is the momentum coefficient. Let $\eta = \frac{1}{\beta}$ and $\alpha = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}$, where $\kappa = \frac{\beta}{\mu} \geq 1$ is the condition number. Choose any target rate degradation factor $\tau \in (1, \frac{1}{\gamma})$, where $\gamma = \sqrt{1 - \frac{1}{\sqrt{\kappa}}}$. Then, for any $N \in \mathbb{N}$ such that $p(N) < \tau^N$ and $\mathbf{v}$ constructed as per (6) using any $\mathbf{w} \in \ell_{exp}(m, \tau\gamma)$, the evolved algorithm $\boldsymbol{\nu}(F, \xi_{t:0})$ defined by $\xi_{t+1} = \pi(F, \xi_t) + v_t$, is such that $\boldsymbol{\nu} \in \widehat{\mathtt{pExp}}^{\pi}_{\mathcal{F}_{SC}^{\beta,\mu}}(\tau\gamma)$.

Proof: It is well known that the NAG algorithm (12) applied to the class $\mathcal{F}_{SC}^{\beta,\mu}$ with the parameters α and η as above is such that $\pi \in \mathtt{pExp}^{\pi}_{\mathcal{F}}(0, \gamma)$, see [1], [2]. Since $\sqrt[N]{p(N)} < \tau < \frac{1}{\gamma}$, we have that $p(N)\gamma^N < 1$ and Theorem 1 applies.

While Corollary 2 focuses on the case where NAG is used as the base algorithm π in (5), we remark that the results extend analogously to any base algorithm $\pi \in \mathtt{pExp}^{\pi}_{\mathcal{F}_{SC}^{\beta,\mu}}(m, \gamma)$ such as those with optimal convergence rates designed using integral quadratic constraints (IQCs) as per [2], [19]. As also discussed after Theorem 1, we note that enhancing accelerated algorithms, which are not monotonic in general, involves a trade-off between keeping the worst-case degradation rate τ as small as possible and the frequency at which we can apply a learned update. Last, we remark that one can always impose a target $\tau \in (1, \frac{1}{\gamma})$. Indeed, a large enough $N \in \mathbb{N}$ such that $p(N) < \tau^N$ always exists since the exponential term dominates over the polynomial one.

The case of nonsmooth optimization: We now turn our attention to the case (1) where the objective $F(x) = f(x) + g(x)$ is nonsmooth. Our first result focuses on the class $\mathcal{F}_{cPL}^{\infty,\mu}$ of potentially nonsmooth proper, lower semi-continuous, convex functions that comply with the following inequality

$$F(x) - \min_{x \in \mathbb{R}^d} F(x) \leq \frac{1}{2\mu} \operatorname{dist}(0, \partial F(x))^2 \,, \qquad (13)$$

where $\partial F(x)$ is the convex subdifferential of F at x, defined as $\partial F(x) = \left\{ s \in \mathbb{R}^d : F(y) \geq F(x) + s^\top (y - x) \,, \ \forall y \in \mathbb{R}^d \right\}$. In particular, note that (13) corresponds to (11) when F is differentiable.

Corollary 3: Consider the class of functions $F \in \mathcal{F}_{cPL}^{\infty,\mu}$. Let π be the proximal point algorithm performing the iterations

$$x_{t+1} = \operatorname{prox}_F^c(x_t) = \min_{x \in \mathbb{R}^d} \ F(x) + \frac{1}{2c}|x - x_t|^2 \,, \qquad (14)$$

where $c > 0$. Let $\gamma = \min\left\{ \frac{1}{\sqrt{1 + c\mu}}, \frac{1}{\sqrt{1 + \frac{c^2}{\beta\mu}}} \right\} \in (0, 1)$. Then, any regular algorithm $\boldsymbol{\sigma} \in \mathtt{pExp}^{\pi}_{\mathcal{F}_{cPL}^{\infty,\mu}}(m, \gamma)$ can be written as

$$x_{t+1} = \nu_t(F, x_{t:0}) = \operatorname{prox}_F^c(x_t) + v_t(F, x_{t:0}) \,, \qquad (15)$$

with $\mathbf{v} \in \ell_{exp}(m, \gamma)$. Viceversa, for any $\mathbf{v} \in \ell_{exp}(m, \gamma)$, the algorithm (15) is such that $\boldsymbol{\nu} \in \widehat{\mathtt{pExp}}^{\pi}_{\mathcal{F}_{RSI}^{\infty,\mu}}(\gamma)$.

Proof: Similarly to Corollary 1, the result follows by combining our Theorem 2 with the linear convergence result of the proximal point method (14) when applied to functions $F \in \mathcal{F}_{cPL}^{\infty,\mu}$ from [14, Theorem 4.2] and the definitions of error bound and quadratic growth from [16].

The result of Corollary 3 holds for any objective $F \in \mathcal{F}_{cPL}^{\infty,\mu}$. In particular, $\mathcal{F}_{cPL}^{\infty,\mu}$ encompasses the class of optimization problems (1), where $f \in \mathcal{F}_{SC}^{\beta,\mu}$ and $g \in \mathcal{F}_{C}^{\infty}$, that is, $g(x)$ is nonsmooth and convex, for which ad-hoc algorithms have been developed to exploit the structure underlying these composite problems. Our next result focuses on the case where $g(x)$ represents the indicator function of a set of convex linear constraints (2b) to address constrained optimization problems of the form (2) with $f_0 \in \mathcal{F}_{SC}^{\beta,\mu}$.

Corollary 4: Consider the constrained optimization problem (2) with $f_0 \in \mathcal{F}_{SC}^{\beta,\mu}$ and $f_i(x) = A_i x - b_i$ for all $i \in [1, M]$ and define the feasible set $\mathcal{X} = \{x \in \mathbb{R}^d : f_i(x) \leq 0,\ \forall i \in [1, M]\}$. Let $g(x) = \mathbb{I}_{\mathcal{X}}(x)$ and define $\mathcal{F}_{\text{comp}}$ as the set of all such functions $F(x) = f_0(x) + g(x)$. Let π be the proximal gradient descent method performing the iterations

$$\begin{aligned} x_{t+1} &= \min_{x \in \mathbb{R}^d}\ g(x) + \frac{1}{2}|x - (x_t - \eta \nabla f(x_t)|^2 \qquad (16)\\ &= \text{prox}_g(x_t - \eta \nabla f(x_t)) = \text{proj}_{\mathcal{X}}(x_t - \eta \nabla f(x_t))\,, \end{aligned}$$

where $\eta \in (0, \frac{1}{\beta}]$. Consider any regular algorithm $\chi_{t+1} = \sigma_t(F, \chi_{t:0})$ with feasible iterates $\chi_t \in \mathcal{X}$ and such that $\boldsymbol{\sigma} \in \texttt{pExp}^{\pi}_{\mathcal{F}_{\text{comp}}}(m, \gamma)$, with $\gamma = 1 - \eta\mu$. Then, there exists $\mathbf{v} \in \ell_{exp}(m, \gamma)$ such that at all times $A_i v_t \leq 0$ for all $i \in [1, M]$, and the evolved algorithm

$$x_{t+1} = \nu_t(F, x_{t:0}) = \text{proj}_{\mathcal{X}}(x_t - \eta \nabla f(x_t)) + v_t(F, x_{t:0})\,, \qquad (17)$$

is equivalent to $\boldsymbol{\sigma}$. Viceversa, for any $\mathbf{v} \in \ell_{exp}(m, \gamma)$ such that at all times $A_i v_t \leq 0$ for all $i \in [1, M]$, the algorithm (17) is such that the iterates $x_t \in \mathcal{X}$ and $\boldsymbol{\nu} \in \widehat{\texttt{pExp}}^{\pi}_{\mathcal{F}_{\text{comp}}}(\gamma)$.

Proof: The base algorithm π defined in (16) is such that $\pi \in \texttt{Exp}^{\pi}_{\mathcal{F}_{\text{comp}}}(1 - \eta\mu)$ as shown in [20, Theorem 11.5]. Now, consider any regular algorithm $\chi_{t+1} = \sigma_t(F, \chi_{t:0})$ with feasible iterates $\chi_t \in \mathcal{X}$ and such that $\boldsymbol{\sigma} \in \texttt{pExp}^{\pi}_{\mathcal{F}_{\text{comp}}}(m, \gamma)$, with $\gamma = 1 - \eta\mu$. Its iterates are equivalent to those of (17) by choosing

$$v_t = -\pi(F, \chi_t) + \sigma_t(F, \chi_{t:0})\,, \quad x_0 = \chi_0\,. \qquad (18)$$

Next, we verify that $A_i v_t \leq 0$ for all $i \in [1, M]$. We have

$$\begin{aligned} A_i v_t &= -A_i \pi(F, \chi_t) + A_i \sigma_t(F, \chi_{t:0})\\ &= -A_i \,\text{proj}_{\mathcal{X}}\left(\chi_t - \eta \nabla f(\chi_t)\right) + A_i \chi_{t+1}\,, \end{aligned}$$

where both χ_{t+1} and $\widetilde{\chi}_{t+1} := \text{proj}_{\mathcal{X}}\left(\chi_t - \eta \nabla f(\chi_t)\right)$ lie in $\mathcal{X}$ by definition, and hence $A_i \chi_{t+1} \leq b$ and $A_i \widetilde{\chi}_{t+1} \leq b$. It follows that $A_i v_t \leq 0$. Next, we verify that $\pi(F, x)$ is Lipschitz in x. Since the projection onto an affine subspace is 1-Lipschitz, it holds that

$$\begin{aligned} |\pi(F, x) - \pi(F, y)| &\leq |x - y + \eta \nabla f(y) - \eta \nabla f(x)|\\ &\leq (1 + \eta\beta)|x - y|\,. \end{aligned}$$

Last, analogously to the proof of Theorem 2, it holds that $\mathbf{v} \in \ell_{exp}(m, 1 - \eta\mu)$ because $\boldsymbol{\sigma}$ is regular. Viceversa, if $\mathbf{v} \in \ell_{exp}(m, \gamma)$ is such that at all times $A_i v_t \leq 0$ for all $i \in [1, M]$, the iterates of (17) are feasible because $A_i(\pi(F, x_t) + v_t) \leq b$, and $\boldsymbol{\nu} \in \widehat{\texttt{pExp}}^{\pi}_{\mathcal{F}_{\text{comp}}}(1 - \eta\mu)$ by Theorem 1.

Leveraging the composite structure of (2), Corollary 4 addresses the requirement of ensuring feasibility of all iterates of (17) in optimization problems with polytopic constraints. In fact, while Corollary 3 guarantees convergence rates of the evolved algorithm $\boldsymbol{\nu}$, feasibility of iterates x_t of (15) maybe be lost for arbitrary choices of $\mathbf{v} \in \ell_{exp}(m, \gamma)$.

Last, we turn our attention to evolving the alternating direction method of multipliers (ADMM) algorithm [21], which is well-suited for multi-agent optimization settings where minimizing a global cost function $F(x) = f(x) + g(x)$ can be parallelized by iteratively and separately optimizing for $f(x)$ and $g(x)$ according to the iterations:

$$x^{[1]}_{t+1} = \arg\min_x f(x) + \frac{\rho}{2}|x - x^{[2]}_t + u_t|^2\,, \qquad (19a)$$

$$x^{[2]}_{t+1} = \arg\min_x g(x) + \frac{\rho}{2}|x^{[1]}_{t+1} - x + u_t|^2\,, \qquad (19b)$$

$$u_{t+1} = u_t + x^{[1]}_{t+1} - x^{[2]}_{t+1}\,, \qquad (19c)$$

where (19a) and (19b) represent the updates of two different agents, (19c) is an aggregation step, and $\rho > 0$ is a hyperparameter whose tuning is critical for fast convergence. Since $x^{[1]}_t$ does not enter the updates on the right-hand side of (19a)-(19c), one can define $\xi_t = (x^{[2]}_t, u_t)$ and express (19a)-(19c) equivalently as:

$$\xi_{t+1} = \pi(\xi_t) = \begin{bmatrix} \arg\min_x g(x) + \frac{\rho}{2}|y(x^{[2]}_t, u_t) - x + u_t|^2 \\ u_t + y(x^{[2]}_t, u_t) - z(x^{[2]}_t, u_t)\,, \end{bmatrix} \qquad (20)$$

where we let $y(x^{[2]}_t, u_t)$ denote the right-hand-side of (19a) and $z(x^{[2]}_t, u_t)$ denote the right-hand-side of (19b) to highlight the dependency on ξ_t only.

Corollary 5: Consider the optimization problem (1), where $f \in \mathcal{F}_{SC}^{\beta,\mu}$ and $g \in \mathcal{F}_{C}^{\infty}$. Let π be the ADMM algorithm (20) with $\rho = \mu\beta$ and consider the evolved algorithm $\xi_{t+1} = \pi(\xi_t) + v_t(\xi_{t:0}) = \nu_t(\xi_{t:0})$ corresponding to the iterations

$$x^{[1]}_{t+1} = \arg\min_x f(x) + \frac{\rho}{2}|x - x^{[2]}_t + u_t|^2\,, \qquad (21)$$

$$x^{[2]}_{t+1} = \arg\min_x g(x) + \frac{\rho}{2}|x^{[1]}_{t+1} - x + u_t|^2 + \qquad (22)$$

$$+ v^{[2]}_t(x^{[1]}_{t+1:0}, x^{[2]}_{t:0}, u_{t:0})\,, \qquad (23)$$

$$u_{t+1} = u_t + x^{[1]}_{t+1} - x^{[2]}_{t+1} + v^{u}_t(x^{[1]}_{t+1:0}, x^{[2]}_{t+1:0}, u_{t:0})\,, \qquad (24)$$

where $v_t = (v^{[2]}_t, v^{u}_t)$. Let $\gamma = 1 - \frac{1}{2\sqrt{\kappa}}$, where $\kappa = \frac{\beta}{\mu}$ is the condition number of f. Then, for any $\mathbf{v} \in \ell_{exp}(m, \gamma)$, the evolved ADMM algorithm $\boldsymbol{\nu}$ is such that $\boldsymbol{\nu} \in \widehat{\texttt{pExp}}^{\pi}_{\mathcal{F}}(\gamma)$.

Proof: It has been proven in [22] that $\pi \in \texttt{pExp}^{\pi}_{\mathcal{F}}(0, \gamma)$. Further notice that $\pi \in \texttt{Exp}^{\pi}_{\mathcal{F}}(0, \gamma)$ because κ_B of [22] is equal to 1 and because the iterations (19a)-(19c) correspond to the choice $\alpha = 1$ in [22]. The result of Theorem 1 thus applies with $N = 1$.

As opposed to Corollary 2 for the case of smooth strongly convex optimization using NAG as a base algorithm, the accelerated convergence rate of ADMM can be exactly preserved.

C. How to evolve an algorithm with neural networks

In order to evolve legacy algorithms according to theorems above, we consider objective functions drawn from a specific distribution $\mathbb{D}_{\mathcal{F}}$ representing problems encountered in a specific application, tailored to which we aim to evolve a linearly convergent base algorithm $\pi \in \widehat{\texttt{pExp}}^{\pi}_{\mathcal{F}}(\gamma)$. The algorithm design problem can be expressed as

$$\min_{v_0, v_1, \ldots} \quad \mathbb{E}_{F \sim \mathbb{D}_{\mathcal{F}}} \left[\texttt{AlgoPerf}(F, \boldsymbol{\xi}) \right] \tag{25a}$$

$$\text{subject to} \quad \xi_{t+1} = \xi_t + \pi(F, \xi_t) + v_t(\xi_{t:0}), \tag{25b}$$

$$\boldsymbol{\xi} \in \ell_{exp}(\overline{m}, \overline{\gamma}), \quad \forall \xi_0 \in \mathbb{R}^d \,, \tag{25c}$$

where $\overline{m} \geq m$ and $\overline{\gamma} \in [\gamma, 1)$ represent a target linear convergence rate, and $\texttt{AlgoPerf}(\cdot)$ measures the performance of the evolved algorithm (25b) initialized at ξ_0 in optimizing the function F. We refer to [7], [9], [23] for commonly used algorithm performance metrics.

In order to search over update functions $v_t(\xi_{t:0})$, and similar to the technique introduced in [11], it is convenient to decompose exponentially decaying evolution terms $v_t(F, \xi_{t:0})$ as per

$$v_t(F, \xi_{t:0}) = M_t(F, \xi_0) D_t(F, \xi_{t:0}) \,, \tag{26}$$

where $\mathbf{M}(F, \xi_0) \in \ell_{exp}(m, \gamma)$ must be an exponentially decaying magnitude term for any ξ_0, and $|D_t(F, \xi_{t:0})| \leq 1$ is an arbitrarily designed direction term. One can, for instance employ a finite-dimensional parametrization of v_t in (26) as per

$$v_t = \texttt{LRU}_t(\xi_0, \theta) \tanh(\texttt{NN}(\xi_t, \phi)) \,, \tag{27}$$

where the $\texttt{LRU}_t(\theta)(v_{t:0})$ terms are generated by the linear recurrent unit (LRU) [24] defined as;

$$\begin{aligned} \zeta_{t+1} &= \Lambda \, \xi_t + \Gamma(\Lambda) \, B \, w_t, \\ \texttt{LRU}_t((w_{t:0}) &= \texttt{NN}(\mathbb{R}e(C \, \zeta_t) + D \, w_t, \psi) + F w_t \,, \end{aligned} \tag{28}$$

where $\mathbb{R}e$ denotes the real part operator and $\theta = (\Lambda, C, D, F, \psi)$, $w_0 = \xi_0$ and $w_t = 0$ for all $t \in [1, \infty)$, and crucially, Λ is a stable matrix. As a result, the magnitude term $\texttt{LRU}_t(\xi_0, \theta)$ will decay exponentially to a rate we can design, and the direction term $\tanh(\texttt{NN}(\xi_t, \phi))$ picks the direction of the updates without affecting the magnitude. One can then improve the empirical performance of a legacy algorithm by evaluating it over several runs and backpropagating using standard libraries such as Pytorch. The final version of this manuscript will incude numerical results.

APPENDIX

A. Proof of Theorem 1

We first prove the result by assuming that $\pi \in \texttt{Exp}^{\pi}_{\mathcal{F}}(\gamma)$. This is instrumental towards establishing the general result. Let $\delta_t = \text{dist}(\xi_t, \texttt{Fix}_\pi)$ for compactness. By the algorithm definition (5) and the triangle inequality, we have that for every $F \in \mathcal{F}$

$$\begin{aligned} \delta_t &= \text{dist}(\pi(F, \xi_{t-1}) + v_{t-1}, \texttt{Fix}_\pi) \\ &= \inf_{c \in \texttt{Fix}_\pi} \text{dist}(\pi(F, \xi_{t-1}) + v_{t-1}, c) \\ &\leq \inf_{c \in \texttt{Fix}_\pi} \text{dist}(\pi(F, \xi_{t-1}), c) + |v_{t-1}| \\ &= \text{dist}(\pi(F, \xi_{t-1}), \texttt{Fix}_\pi) + |v_{t-1}| \,. \end{aligned}$$

Assuming that $\pi \in \texttt{Exp}^{\pi}_{\mathcal{F}}(\gamma)$, we have that (4) holds with $p(t) = 1$. It follows that $\delta_t \leq \gamma \delta_{t-1} + |v_{t-1}|$. Iterating this inequality, we deduce that

$$\begin{aligned} \delta_t &\leq \gamma^t \delta_0 + \sum_{k=0}^{t-1} \gamma^k |v_{t-1-k}| \\ &\leq \gamma^t \delta_0 + \sum_{k=0}^{t-1} \gamma^k p(t-1-k) \gamma^{t-1-k} \\ &\leq \gamma^t \left(\delta_0 + \frac{1}{\gamma} \sum_{k=0}^{t-1} p(k) \right) \,, \end{aligned}$$

where we used the fact that $\mathbf{v} \in \ell_{exp}(m, \gamma)$. Let $q(t) = \sum_{k=0}^{t} p(k)$ and note that the right-hand side of the above can be written as $\gamma^t r(t)$ where $r(t) = \delta_0 + \frac{1}{\gamma} q(t-1)$. We study $q(t)$. By linearity of summation, $q(t)$ can be equivalently rewritten as

$$\sum_{k=0}^{t} \sum_{j=0}^{m} a_j k^j = a_m \sum_{k=0}^{t} k^m + \cdots + a_1 \sum_{k=0}^{t} k + a_0 \sum_{k=0}^{t} 1 \,,$$

where $a_j \in \mathbb{R}$ with $j \in \{0, \ldots, m\}$ is the j-th coefficient of the polynomial $p(\cdot)$. Faulhaber's formula implies that $q(t)$ is a polynomial of degree $m+1$ in the variable t with coefficient $b_{m+1} = \frac{a_m}{m+1}$. Furthermore, $q(t)$ is positive and monotonically non-decreasing by construction, that is, $q(t) \in \mathcal{P}_{m+1}(t)$. Note that $q(t) \in \mathcal{P}_{m+1}(t)$ implies $r(t) \in \mathcal{P}_{m+1}(t)$. Hence, we conclude that $\delta_t \leq r(t) \gamma^t$ for all $t \in \mathbb{N}$, which proves the result for the case $\pi \in \texttt{Exp}^{\pi}_{\mathcal{F}}(\gamma)$.

We now turn our attention to the general case where π is any linearly convergent algorithm in $\texttt{pExp}^{\pi}_{\mathcal{F}}(m, \gamma)$. For any $w_t \in \ell_{exp}(m, \gamma)$, consider the recursion

$$\zeta_{k+1} = \pi^N(F, \zeta_k) + w_t \,, \tag{29}$$

where $\pi^N(\cdot)$ denotes the repeated application of π defined as $\pi^N(F, \zeta_k) = \pi(F, \pi^{N-1}(F, \zeta_k))$ with $\pi^1(F, \zeta_k) = \pi(F, \zeta_k)$.

We first observe that, if $\zeta_0 = \xi_0$ and $\mathbf{v}$ is constructed as per (6), then (29) is equivalent to (5) in the sense that $\zeta_k = \xi_{Nk}$ for every $k \in \mathbb{N}$. By construction, $\pi^N \in \texttt{Exp}^{\pi}_{\mathcal{F}}(\rho)$ and therefore complies with (4) with $p(t) = 1$ and $\gamma = \rho$. Hence, as proven above, it holds that

$$\text{dist}(\boldsymbol{\zeta}, \texttt{Fix}_{\pi^N}) \in \ell_{exp}(m+1, \rho) \,.$$

We now argue that $\texttt{Fix}_{\pi^N} = \texttt{Fix}_\pi$. Clearly, $\texttt{Fix}_{\pi^N} \supseteq \texttt{Fix}_\pi$ since $\xi^\star = \pi(F, \xi^\star)$ for every $\xi^\star \in \texttt{Fix}_\pi$ and thus $\xi^\star = \pi^N(F, \xi^\star)$. To show that $\texttt{Fix}_{\pi^N} \subseteq \texttt{Fix}_\pi$, assume there exists $\zeta^\star \in \texttt{Fix}_{\pi^N}$ such that $\zeta^\star \notin \texttt{Fix}_\pi$. Since $\pi \in \texttt{pExp}^{\pi}_{\mathcal{F}}(m, \gamma)$, we have that $\lim_{t \to \infty} \text{dist}(\pi^t(F, \zeta^\star), \texttt{Fix}_\pi) = 0$. At the same time, $\text{dist}(\pi^{\tau N}(F, \zeta^\star), \texttt{Fix}_\pi) > 0$ for any $\tau \in \mathbb{N}$ because $\pi^{\tau N}(F, \zeta^\star) = \zeta^\star \notin \texttt{Fix}_\pi$. This is a contradiction, and thus $\texttt{Fix}_{\pi^N} = \texttt{Fix}_\pi$. We conclude that

$$\text{dist}(\boldsymbol{\zeta}, \texttt{Fix}_\pi) \in \ell_{exp}(m+1, \rho) \,,$$

and therefore there exists a polynomial $q(k) \in \mathcal{P}_{m+1}(k)$ such that $\text{dist}(\zeta_k, \texttt{Fix}_\pi) \leq q(k) \rho^k$ for all $k \in \mathbb{N}$.

Next, we note that, for any $s \in \{1, \ldots, N-1\}$

$$\begin{aligned} \text{dist}(\xi_{Nk+s}, \texttt{Fix}_\pi) &= \text{dist}(\pi^s(F, \xi_{Nk}), \texttt{Fix}_\pi) \\ &\leq p(s) \gamma^s \, \text{dist}(\xi_{Nk}, \texttt{Fix}_\pi) \leq p(s) \gamma^s q(k) \rho^k \,, \end{aligned}$$

where we used the fact that $\operatorname{dist}(\xi_{Nk}, \texttt{Fix}_\pi) = \operatorname{dist}(\zeta_k, \texttt{Fix}_\pi)$ for any $k \in \mathbb{N}$. Letting $t = Nk + s$, and using the fact that $p(\cdot), q(\cdot) \in \mathcal{P}_{m+1}$, we obtain

$$\begin{aligned}
\operatorname{dist}(\xi_t, \texttt{Fix}_\pi) &\leq p(N-1)\gamma q\left(\left\lfloor \frac{t-s}{N} \right\rfloor\right) \rho^{\left\lfloor \frac{t-s}{N} \right\rfloor} \\
&\leq p(N-1)\gamma q\left(\frac{t}{N}\right) \rho^{\left\lfloor \frac{t-N+1}{N} \right\rfloor} \\
&\leq \underbrace{\frac{p(N-1)\gamma}{\rho^{2-\frac{1}{N}}} q\left(\frac{t}{N}\right)}_{r(t) \in \mathcal{P}_{m+1}(t)} \left(\rho^{\frac{1}{N}}\right)^t .
\end{aligned}$$

Since $\rho = p(N)\gamma^N$, we have that $\rho^{\frac{1}{N}} = \sqrt[N]{p(N)}\gamma$. This concludes the proof.

B. Proof of Theorem 2

Let $v_t(F, \xi_{t:0}) = -\pi(F, \chi_t) + \sigma_t(F, \chi_{t:0})$. We first show by induction that $\xi_t = \chi_t$ at all times, starting from the base case $\xi_0 = \chi_0$, which holds by construction. Assume now that $\xi_{t:0} = \chi_{t:0}$. We aim to prove that $\xi_{t+1} = \chi_{t+1}$. This holds because

$$\begin{aligned}
\xi_{t+1} &= \pi(F, \xi_t) - \pi(F, \chi_t) + \sigma_t(F, \chi_{t:0}) \\
&= \sigma_t(F, \chi_{t:0}) = \chi_{t+1} .
\end{aligned}$$

It remains to show that the sequence $v_t(F, \xi_{t:0}) = -\pi(F, \chi_t) + \sigma_t(\chi_{t:0})$ belongs to $\ell_{exp}(m, \gamma)$. To prove this, we rewrite v_t as

$$v_t = -(\pi(F, \chi_t) - \chi_t) + \sigma_t(F, \chi_{t:0}) - \chi_t . \tag{30}$$

Since $\pi(F, \cdot)$ is Lipschitz continuous, letting χ_t^p be any element of $\arg\min_{\chi \in \texttt{Fix}_\pi} |\chi - \chi_t|^2$, there exists a constant $L_\pi \in \mathbb{R}_+$ such that

$$\begin{aligned}
|\pi(F, \chi_t) - \chi_t| &= |\pi(F, \chi_t) - \chi_t^p + \chi_t^p - \chi_t| \\
&= |\pi(F, \chi_t) - \pi(F, \chi_t^p) + \chi_t^p - \chi_t| \\
&\leq (L_\pi + 1)|\chi_t - \chi_t^p| \\
&= (L_\pi + 1) \operatorname{dist}(\chi_t, \texttt{Fix}_\pi) ,
\end{aligned}$$

and hence $-(\pi(F, \boldsymbol{\chi}) - \boldsymbol{\chi}) \in \ell_{exp}(m, \gamma)$. We further have that $\boldsymbol{\sigma}(F, \boldsymbol{\chi}) - \boldsymbol{\chi} \in \ell_{exp}(m, \gamma)$ by the regularity assumption on $\boldsymbol{\sigma}$ as per Definition 2. Since the sum of signals in $\ell_{exp}(m, \gamma)$ belongs to $\ell_{exp}(m, \gamma)$, we conclude the proof by inspection of (30).

REFERENCES

[1] Y. E. Nesterov, "A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$," in *Doklady Akademii Nauk*, vol. 269, no. 3. Russian Academy of Sciences, 1983, pp. 543–547.

[2] L. Lessard, B. Recht, and A. Packard, "Analysis and design of optimization algorithms via integral quadratic constraints," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 57–95, 2016.

[3] C. Scherer and C. Ebenbauer, "Convex synthesis of accelerated gradient algorithms," *SIAM Journal on Control and Optimization*, vol. 59, no. 6, pp. 4615–4645, 2021.

[4] B. Van Scoy and L. Lessard, "The fastest known first-order method for minimizing twice continuously differentiable smooth strongly convex functions," *arXiv preprint arXiv:2506.01168*, 2025.

[5] H. Mohammadi, M. Razaviyayn, and M. R. Jovanović, "Robustness of accelerated first-order algorithms for strongly convex optimization problems," *IEEE Transactions on Automatic Control*, vol. 66, no. 6, pp. 2480–2495, 2020.

[6] D. Mayne, "Nonlinear model predictive control: Challenges and opportunities," *Nonlinear model predictive control*, pp. 23–44, 2000.

[7] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, "Learning to learn by gradient descent by gradient descent," *Advances in neural information processing systems*, vol. 29, 2016.

[8] H. Heaton, X. Chen, Z. Wang, and W. Yin, "Safeguarded learned convex optimization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 6, 2023, pp. 7848–7855.

[9] R. Sambharya, G. Hall, B. Amos, and B. Stellato, "Learning to warm-start fixed-point optimization algorithms," *Journal of Machine Learning Research*, vol. 25, no. 166, pp. 1–46, 2024.

[10] J. Ichnowski, P. Jain, B. Stellato, G. Banjac, M. Luo, F. Borrelli, J. E. Gonzalez, I. Stoica, and K. Goldberg, "Accelerating quadratic optimization with reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 043–21 055, 2021.

[11] A. Martin and L. Furieri, "Learning to optimize with convergence guarantees using nonlinear system theory," *IEEE Control Systems Letters*, vol. 8, pp. 1355–1360, 2024.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[13] S. Banert, J. Rudzusika, O. Öktem, and J. Adler, "Accelerated forward-backward optimization using deep learning," *SIAM Journal on Optimization*, vol. 34, no. 2, pp. 1236–1263, 2024.

[14] F.-Y. Liao, L. Ding, and Y. Zheng, "Error bounds, pl condition, and quadratic growth for weakly convex functions, and linear convergences of proximal point methods," in *6th Annual Learning for Dynamics & Control Conference*. PMLR, 2024, pp. 993–1005.

[15] H. Zhang and W. Yin, "Gradient methods for convex minimization: better rates under weaker conditions," *arXiv preprint arXiv:1303.4645*, 2013.

[16] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2016, pp. 795–811.

[17] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *Ussr computational mathematics and mathematical physics*, vol. 4, no. 5, pp. 1–17, 1964.

[18] E. Ghadimi, H. R. Feyzmahdavian, and M. Johansson, "Global convergence of the heavy-ball method for convex optimization," in *2015 European control conference (ECC)*. IEEE, 2015, pp. 310–315.

[19] B. Van Scoy, R. A. Freeman, and K. M. Lynch, "The fastest known globally convergent first-order method for minimizing strongly convex functions," *IEEE Control Systems Letters*, vol. 2, no. 1, pp. 49–54, 2017.

[20] G. Garrigos and R. M. Gower, "Handbook of convergence theorems for (stochastic) gradient methods," *arXiv preprint arXiv:2301.11235*, 2023.

[21] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[22] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. Jordan, "A general analysis of the convergence of admm," in *International conference on machine learning*. PMLR, 2015, pp. 343–352.

[23] R. Sambharya and B. Stellato, "Data-driven performance guarantees for classical and learned optimizers," *arXiv preprint arXiv:2404.13831*, 2024.

[24] A. Orvieto, S. L. Smith, A. Gu, A. Fernando, C. Gulcehre, R. Pascanu, and S. De, "Resurrecting recurrent neural networks for long sequences," in *International Conference on Machine Learning*. PMLR, 2023, pp. 26 670–26 698.