Primal-Dual Damping algorithms for optimization

Xinzhe Zuo*, Stanley Osher*, Wuchen Li†
* Department of Mathematics, University of California, Los Angles
{zxz, sjo}@math.ucla.edu
† Department of Mathematics, University of South Carolina
wuchen@mailbox.sc.edu

Abstract

We propose an unconstrained optimization method based on the well-known primal-dual hybrid gradient (PDHG) algorithm. We first formulate the optimality condition of the unconstrained optimization problem as a saddle point problem. We then compute the minimizer by applying generalized primal-dual hybrid gradient algorithms. Theoretically, we demonstrate the continuous-time limit of the proposed algorithm forms a class of second-order differential equations, which contains and extends the heavy ball ODEs and Hessian-driven damping dynamics. Following the Lyapunov analysis of the ODE system, we prove the linear convergence of the algorithm for strongly convex functions. Experimentally, we showcase the advantage of algorithms on several convex and non-convex optimization problems by comparing the performance with other well-known algorithms, such as Nesterov's accelerated gradient methods. In particular, we demonstrate that our algorithm is efficient in training two-layer and convolution neural networks in supervised learning problems.

Keywords— Optimization; Primal-dual hybrid gradient algorithms; Primal-dual damping dynamics

1 Introduction

Optimization is one of the essential building blocks in many applications, including scientific computing and machine learning problems. One of the classical algorithms for unconstrained optimization problems is the gradient descent method, which updates the state variable in the negative gradient direction at each step [Boyd and Vandenberghe, 2004]. Nowadays, accelerated gradient descent methods have been widely studied. Typical examples include Nesterov's accelerated gradient method [Nesterov, 1983], Polyak's heavy ball method [Polyak, 1964], and Hessian-driven damping methods [Chen and Luo, 2019, Attouch et al., 2019, 2020, 2021].

On the other hand, some first-order methods are introduced to solve linear-constrained optimization problems. Typical examples include the primal-dual hybrid gradient (PDHG) method [Chambolle and Pock, 2011] and the alternating direction method of multipliers (ADMM) [Boyd et al., 2011, Gabay and Mercier, 1976]. They are designed to solve an inf-sup saddle point type problem, which updates the gradient descent direction for the minimization variable and applies the gradient ascent direction for the maximization variable. Both PDHG and ADMM are designed for solving optimization problems with affine constraints. Ouyang et al. [2015] proposed accelerated linearized ADMM, which incorporates a multi-step acceleration scheme into linearized ADMM. Recently, the PDHG method has been extended into solving nonlinear-constrained minimization problems [Valkonen, 2014].

In this paper, we study a general class of accelerated first-order methods for unconstrained optimization problems. We reformulate the original optimization problem into an inf-sup type saddle point problem, whose saddle point solves the optimality condition. We then apply a linearized preconditioned primal-dual hybrid gradient algorithm to compute the proposed saddle point problem.

The main description of the algorithm is as follows. Consider the following inf-sup problem for a $\mathcal{C}^2$ strongly convex function f over $\mathbb{R}^d$

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d}\sup_{\boldsymbol{p}\in\mathbb{R}^d} \quad \langle \nabla f(\boldsymbol{x}), \boldsymbol{p}\rangle - \frac{\varepsilon}{2}\|\boldsymbol{p}\|^2 \,, \tag{1.1}$$

where $\boldsymbol{p}$ is a constructed "dual variable", $\varepsilon > 0$ is a constant, $\langle\cdot,\cdot\rangle$ is an Euclidean inner product, and $\|\cdot\|$ is an Euclidean norm. We later prove that the solution to the saddle point problem 1.1 gives the global minimum of f. We propose a linearized preconditioned PDHG algorithm for solving the above inf-sup problem:

$$\boldsymbol{p}^{n+1} = \boldsymbol{p}^n + \sigma \boldsymbol{A}(\boldsymbol{x}^n)\nabla f(\boldsymbol{x}^n) - \sigma\varepsilon \boldsymbol{A}(\boldsymbol{x}^n)\boldsymbol{p}^{n+1}\,, \tag{1.2a}$$

$$\tilde{\boldsymbol{p}}^{n+1} = \boldsymbol{p}^{n+1} + \omega(\boldsymbol{p}^{n+1} - \boldsymbol{p}^n)\,, \tag{1.2b}$$

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - \tau \boldsymbol{C}(\boldsymbol{x}^n)\tilde{\boldsymbol{p}}^{n+1}\,, \tag{1.2c}$$

where $n = 1, 2, \cdots$ is the iteration step, τ, $\sigma > 0$ are stepsizes for the updates of $\boldsymbol{x}$, $\boldsymbol{p}$, respectively, and $\omega > 0$ is a parameter. In the above algorithm, $\boldsymbol{C}(\boldsymbol{x}^n) = \boldsymbol{B}(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^n)$, where $\boldsymbol{A}(\boldsymbol{x}^n) \in \mathbb{R}^{d\times d}$, and $\boldsymbol{B}(\boldsymbol{x}^n) \in \mathbb{R}^{d\times d}$ act as preconditioners on the updates of $\boldsymbol{p}^{n+1}$ and $\boldsymbol{x}^{n+1}$, respectively. This paper will only focus on the simple case where $\boldsymbol{A}(\boldsymbol{x}) = A\mathbb{I}$ for some constant $A > 0$. Although there is a second-order term $\nabla^2 f(\boldsymbol{x}^n)$ in the update of $\boldsymbol{x}^{n+1}$ (hidden in $\boldsymbol{C}(\boldsymbol{x}^n)$), our algorithm is still a first-order algorithm by choosing $\boldsymbol{B}(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^n) = \boldsymbol{C}(\boldsymbol{x}^n)$ for some $\boldsymbol{C}$ that is easy to compute. For example, we test that $\boldsymbol{C} = \mathbb{I}$ is a very good choice in most of our numerical examples. See empirical choices of parameters in our numerical sections.

Our method forms a class of ordinary differential equation systems in terms of $(\boldsymbol{x}, \boldsymbol{p})$ in the continuous limit τ, $\sigma \to 0$. We call it the primal-dual damping (PDD) dynamics. We show that the PDD dynamics form a class of second-order ODEs, which contains and extends the inertia Hessian-driven damping dynamics [Chen and Luo, 2019, Attouch et al., 2019]. Theoretically, we analyze the convergence property of PDD dynamics. If f is a quadratic function of $\boldsymbol{x}$, with constant $\boldsymbol{A}$, $\boldsymbol{B}$, the PDD dynamic satisfies a linear ODE system. Under suitable choices of parameters, we obtain a similar convergence acceleration in heavy ball ODE [Siegel, 2019]. Moreover, for general nonlinear function f, we have the following informal theorem characterizing the convergence speed of our algorithm:

Theorem 1.1 (Informal)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a $\mathcal{C}^4$ strongly convex function. Let $\boldsymbol{x}^*$ be the global minimum of f and $\boldsymbol{p}^* = 0$. Then, the iteration $(\boldsymbol{x}^n, \boldsymbol{p}^n)$ produced by 1.2 converges to the saddle point $(\boldsymbol{x}^*, \boldsymbol{p}^*)$ if τ, σ, are small enough. Moreover,*

$$\|\boldsymbol{p}^n\|^2 + \|\nabla f(\boldsymbol{x}^n)\|^2 \le (\|\boldsymbol{p}^0\|^2 + \|\nabla f(\boldsymbol{x}^0)\|^2)(1 - \frac{\mu^2}{M+\delta})^n,$$

where $\mu = \min_{\boldsymbol{x}} \lambda_{\min}(\nabla^2 f(\boldsymbol{x})\boldsymbol{C}(\boldsymbol{x}))$, $\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x})$, $\delta > 0$ depends on the initial condition, and $M > 0$ depends on $\boldsymbol{C}(\boldsymbol{x})^T\big(\nabla^3 f(\boldsymbol{x})\nabla f(\boldsymbol{x}) + (\nabla^2 f(\boldsymbol{x}))^2\big)\boldsymbol{C}(\boldsymbol{x})$, τ, σ, A, ε, and ω. The detailed version is given in Theorem 3.10.

Numerically, we test the algorithms in both convex and non-convex optimization problems. In convex optimization, we demonstrate the fast convergence results of the proposed algorithm with selected preconditioners, compared with the gradient descent method, Nesterov accelerated gradient method, and Heavy ball damping method. This justifies the convergence analysis. We also test our algorithm for several well-known non-convex optimization problems. Some examples, such as

the Rosenbrock and Ackley functions, demonstrate the potential advantage of our algorithms in converging to the global minimizer. In particular, we compare our algorithms with the stochastic gradient descent method, Adam, for training two-layer and convolutional neural network functions in supervised learning problems. This showcases the potential advantage of the proposed methods in terms of convergence speed and test accuracy.

PDHG has been widely used in linear-constrained optimization problems [Chambolle and Pock, 2011]. Recently, Valkonen [2014] applied the PDHG for nonlinearly constrained optimization problems. They proved the asymptotic convergence for the nonlinear coupling saddle point problems. It is different from our PDHG algorithm for computing unconstrained optimizations. And we show the linear convergence for a particular nonlinear coupling saddle point problem. Meanwhile, Nesterov accelerated gradient methods and Hessian damping algorithms can also be formulated in both discrete-time updates and continuous-time second-order ODEs. Wibisono et al. [2016] also introduced the idea of Bregman Lagrangian to study a family of accelerated methods in continuous time limit. It forms a nonlinear second-order ODE. Compared to them, the PDD algorithm induces a generalized second-order ODE system, which contains both heavy ball ODE [Siegel, 2019] and Hessian damping dynamics [Chen and Luo, 2019, Attouch et al., 2019, 2020, 2021]. For example, when $C = \mathbb{I}$, algorithm Eq. 1.2 can be viewed as the other time discretization of Hessian damping dynamics [Chen and Luo, 2019, Attouch et al., 2019]. It provides a different update in discrete time update. We only evaluate the gradient of f once, whereas Attouch's algorithm [Attouch et al., 2020] evaluates the gradient of f twice. In numerical experiments, we demonstrate that the proposed algorithm outperforms Nesterov accelerated methods and Hessian-driven damping methods in some non-convex optimization problems, including supervised learning problems for training neural network functions.

Our work is also related to preconditioning, an important technique in numerical linear algebra [Trefethen and Bau, 2022] and numerical PDEs [Rees, 2010, Park et al., 2021]. In general, preconditioning aims to reduce the condition number of some operators to improve convergence speed. One famous example would be preconditioning gradient descent by the inverse of the Hessian matrix, which gives rise to Newton's method. In recent years, preconditioning techniques have also been developed in training neural networks Osher et al. [2022], Kingma and Ba [2014]. Adam [Kingma and Ba, 2014] is arguably one of the most popular optimizers in training deep neural networks. It can also be viewed as a preconditioned algorithm using a diagonal preconditioner that approximates the diagonal of the Fisher information matrix [Pascanu and Bengio, 2013]. Shortly after Chambolle and Pock [2011] developed PDHG for constrained optimization, the same authors also studied preconditioned PDHG method [Pock and Chambolle, 2011], in which they developed a simple diagonal preconditioner that can guarantee convergence without the need to compute step sizes. Liu et al. [2021] proposed non-diagonal preconditioners for PDHG and showed close connections between preconditioned PDHG and ADMM. Park et al. [2021] studied the preconditioned Nesterov's accelerated gradient method and proved convergence in the induced norm. Jacobs et al. [2019] introduced a preconditioned norm in the primal update of the PDHG method and improved the step size restriction of the PDHG method.

Our paper is organized as follows. In Section 2 we provide some background and derivations of our algorithm. We also provide the ODE formulations for our primal-dual damping dynamics. In Section 3, we prove our main convergence results for the algorithm. In Section 4 we showcase the advantage of our algorithm through several convex and non-convex examples. In particular, we show that our algorithm can train neural networks and is competitive with commonly used optimizers, such as SGD with Nesterov's momentum and Adam. We conclude in Section 5 with more discussions and future directions.

2 Primal-dual damping algorithms for optimizations

In this section, we first review PDHG algorithms for constrained optimization problems. We then construct a saddle point problem for the unconstrained optimization problem and apply the precon-

ditioned PDHG algorithm to compute the proposed saddle point problem. We last derive an ODE system, which takes the limit of stepsizes in the PDHG algorithm. It forms a second-order ODE, which generalizes the Hessian-driven damping dynamics. We analyze the convergence properties of the ODE system for quadratic optimization problems.

2.1 Review PDHG for constrained optimization

In Chambolle and Pock [2011], the following saddle point problem was considered:

$$\min_{x\in X}\max_{y\in Y}\langle Kx, y\rangle + G(x) - F^*(y)\,, \tag{2.1}$$

where X and Y are two finite-dimensional real vector spaces equipped with inner product $\langle\cdot,\cdot\rangle$ and norm $\|\cdot\| = \langle\cdot,\cdot\rangle^{1/2}$. The map $K : X \to Y$ is a continuous linear operator. $G : X \to [0, +\infty]$ and $F^* : Y \to [0, +\infty]$ are proper, convex, lower semi-continuous (l.s.c.) functions. F^* is the convex conjugate of a convex l.s.c. function F. It is straightforward to verify that 2.1 is the primal-dual formulation of the nonlinear primal problem

$$\min_{x\in X} F(Kx) + G(x)\,.$$

Then the PDHG algorithm for saddle point problem 2.1 is given by

$$y^{n+1} = (I + \sigma\partial F^*)^{-1}(y^n + \sigma K\tilde{x}^n)\,, \tag{2.2a}$$
$$x^{n+1} = (I + \tau\partial G)^{-1}(x^n + \tau K^* y^{n+1})\,, \tag{2.2b}$$
$$\tilde{x}^{n+1} = x^{n+1} + \omega(x^{n+1} - x^n)\,, \tag{2.2c}$$

where $(I+\sigma\partial F)^{-1}$ is the resolvent operator, which is defined the same way as the proximal operator

$$\begin{aligned}(I + \tau\partial F)^{-1}(y) &= \arg\min_x \frac{\|x - y\|^2}{2\tau} + F(x)\\ &= \mathrm{prox}_{\tau F}(y)\end{aligned}$$

When $\omega = 1$, Chambolle and Pock [2011] proved convergence if $\tau\sigma\|K\|^2 < 1$, where $\|\cdot\|$ is the induced operator norm. It is worth noting that the convergence analysis requires that K is a linear operator.

2.2 Saddle point problem for unconstrained optimization

We consider the problem of minimizing a $\mathcal{C}^2$ strongly convex function $f : \mathbb{R}^d \to \mathbb{R}$ over $\mathbb{R}^d$. Instead of directly solving for $\nabla f(\boldsymbol{x}^*) = 0$, we consider the following saddle point problem:

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d}\sup_{\boldsymbol{p}\in\mathbb{R}^d} \quad \langle\nabla f(\boldsymbol{x}), \boldsymbol{p}\rangle\,, \tag{2.3}$$

due to the following proposition.

Proposition 2.1. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a $\mathcal{C}^2$ strongly convex function. Then the saddle point to 2.3 is the unique global minimum of f.*

Proof. Directly differentiating 2.3 and setting the derivatives to 0 yields

$$\begin{aligned}\nabla f(\boldsymbol{x}^*) &= 0\,,\\ \nabla^2 f(\boldsymbol{x}^*)\boldsymbol{p}^* &= 0\,.\end{aligned}$$

By the strong convexity of f, we obtain that $\boldsymbol{x}^*$ is the unique global minimum and $\boldsymbol{p}^* = 0$. □

Recall that $\boldsymbol{p}^* = 0$ by the optimality condition. Thus we make the following change to our saddle point formulation. We add a regularization term in 2.3:

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d} \sup_{\boldsymbol{p}\in\mathbb{R}^d} \quad \langle \nabla f(\boldsymbol{x}), \boldsymbol{p}\rangle - \frac{\varepsilon}{2}\|\boldsymbol{p}\|^2 \,, \tag{2.4}$$

where $\varepsilon > 0$ is a constant. This regularization term further drives $\boldsymbol{p}$ to 0. Similar to Proposition 2.1, we have the following proposition

Proposition 2.2. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a $\mathcal{C}^2$ strongly convex function. Then the saddle point to 2.4 is the unique global minimum of f.*

Proof. Directly differentiating 2.4 and setting derivatives to 0 yields

$$\begin{aligned} \nabla f(\boldsymbol{x}^*) &= \varepsilon \boldsymbol{p}^* \,, \\ \nabla^2 f(\boldsymbol{x}^*)\boldsymbol{p}^* &= 0 \,. \end{aligned}$$

Since f is strongly convex, we have $\nabla^2 f(\boldsymbol{x}^*) \succ 0$ and the second equation implies $\boldsymbol{p}^* = 0$. Then the first equation implies $\nabla f(\boldsymbol{x}^*) = 0$. Since f is strongly convex, we conclude that $\boldsymbol{x}^*$ is the unique global minimum. □

2.3 PDHG for unconstrained optimization

We apply the scheme given by 2.2 to the saddle point problem 2.4 (set $F = G = 0$ and identify $K\boldsymbol{x} = \nabla f(\boldsymbol{x})$ in 2.1). Thus,

$$\boldsymbol{p}^{n+1} = \arg\max_{\boldsymbol{p}} \quad \langle \nabla f(\boldsymbol{x}^n), \boldsymbol{p}\rangle - \frac{\varepsilon}{2}\|\boldsymbol{p}\|^2 - \frac{\|\boldsymbol{p} - \boldsymbol{p}^n\|^2_{\boldsymbol{A}(\boldsymbol{x}^n)^{-1}}}{2\sigma} \,, \tag{2.5a}$$

$$\widetilde{\boldsymbol{p}}^{n+1} = \omega(\boldsymbol{p}^{n+1} - \boldsymbol{p}^n) + \boldsymbol{p}^{n+1} \,, \tag{2.5b}$$

$$\boldsymbol{x}^{n+1} = \arg\min_{\boldsymbol{x}} \quad \langle \nabla f(\boldsymbol{x}), \widetilde{\boldsymbol{p}}^{n+1}\rangle + \frac{\|\boldsymbol{x} - \boldsymbol{x}^n\|^2_{\boldsymbol{B}(\boldsymbol{x}^n)^{-1}}}{2\tau} \,, \tag{2.5c}$$

where we have added symmetric positive definite matrices $\boldsymbol{A}(\boldsymbol{x}^n)$, $\boldsymbol{B}(\boldsymbol{x}^n) \in \mathbb{R}^{d\times d}$, as preconditioners for updates of $\boldsymbol{p}$, $\boldsymbol{x}$, respectively. We also denote the norm $\|\boldsymbol{h}\|^2_{\boldsymbol{A}^{-1}}$ as $\boldsymbol{h}^T\boldsymbol{A}^{-1}\boldsymbol{h}$, where $\boldsymbol{h} \in \mathbb{R}^d$.

As mentioned, the convergence analysis of PDHG relies on the assumption that K is a linear operator. So we can not apply the same convergence analysis to 2.5 since $\nabla f(\boldsymbol{x})$ is not necessarily linear in $\boldsymbol{x}$. By taking the optimality conditions of 2.5, we find that $\boldsymbol{p}^{n+1}$ and $\boldsymbol{x}^{n+1}$ solves

$$\boldsymbol{p}^{n+1} + \sigma\varepsilon\boldsymbol{A}(\boldsymbol{x}^n)\boldsymbol{p}^{n+1} - \boldsymbol{p}^n - \sigma\boldsymbol{A}(\boldsymbol{x}^n)\nabla f(\boldsymbol{x}^n) = 0 \,, \tag{2.6a}$$

$$\tau\boldsymbol{B}(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^{n+1})\big((1+\omega)\sigma\boldsymbol{A}(\boldsymbol{x}^n)\nabla f(\boldsymbol{x}^n) + \boldsymbol{p}^n\big) + (\boldsymbol{x}^{n+1} - \boldsymbol{x}^n) = 0 \,, \tag{2.6b}$$

where we substitute the update Eq. 2.5b into update Eq. 2.6b.

Note that the update for $\boldsymbol{x}^{n+1}$ in Eq. 2.6b is implicit, unless $\nabla^2 f(\boldsymbol{x})$ does not depend on $\boldsymbol{x}$. We also remark that the update for $\boldsymbol{x}^{n+1}$ in Eq. 2.6b will be explicit if we perform a gradient step instead of a proximal step in Eq. 2.5c. To be more precise, when $\boldsymbol{B} = \mathbb{I}$, the linearized version of Eq. 2.5c can be written as

$$\boldsymbol{x}^{n+1} = \mathrm{prox}_{\tau\langle\nabla f(\cdot), \widetilde{\boldsymbol{p}}^{n+1}\rangle}(\boldsymbol{x}^n) \,.$$

Taking a gradient step instead of proximal step yields

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - \tau\nabla^2 f(\boldsymbol{x}^n)\widetilde{\boldsymbol{p}}^{n+1} \tag{2.7}$$

For general choice of preconditioner $\boldsymbol{B}(\boldsymbol{x}^n)$, the linearized version of Eq. 2.5c satisfies

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - \tau\boldsymbol{B}(x^n)\nabla^2 f(\boldsymbol{x}^n)\widetilde{\boldsymbol{p}}^{n+1} = \boldsymbol{x}^n - \tau\boldsymbol{C}(x^n)\widetilde{\boldsymbol{p}}^{n+1}.$$

Here we always denote a matrix function $\boldsymbol{C}$, such that

$$\boldsymbol{C}(\boldsymbol{x}^n) := \boldsymbol{B}(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^n).$$

For simplicity of presentation, we only consider the simple case where $\boldsymbol{A}(\boldsymbol{x}^n) = A\mathbb{I}$ for some $A > 0$. We now summarize the linearized update Eq. 2.6 into the following algorithm.

Algorithm 1 Linearized Primal-Dual Damping Algorithm

Require: Initial guesses $\boldsymbol{x}^0 \in \mathbb{R}^d$, $\boldsymbol{p}^0 \in \mathbb{R}^d$; Stepsizes $\tau > 0$, $\sigma > 0$; Parameters $A > 0$, $\varepsilon > 0$, $\omega > 0$, $\boldsymbol{C} \succ 0$.
while $n = 1, 2, \cdots$, not converge **do**
$\boldsymbol{p}^{n+1} = \frac{1}{1+\sigma\varepsilon A}\boldsymbol{p}^n + \frac{\sigma A}{1+\sigma\varepsilon A}\nabla f(\boldsymbol{x}^n)$;
$\tilde{\boldsymbol{p}}^{n+1} = \boldsymbol{p}^{n+1} + \omega(\boldsymbol{p}^{n+1} - \boldsymbol{p}^n)$;
$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - \tau\boldsymbol{C}(\boldsymbol{x}^n)\tilde{\boldsymbol{p}}^{n+1}$;
end while

We note that Algorithm 1 and update Eq. 2.6 are different methods for solving saddle point problem Eq. 2.3. In this paper, we focus on the computation and analysis of Algorithm 1.

2.4 PDD dynamics

An approach for analyzing optimization algorithms is by first studying the continuous limit of the algorithm using ODEs [Su et al., 2015, Siegel, 2019, Attouch et al., 2019]. The advantage of doing so is that ODEs provide insights into the convergence property of the algorithm.

We first reformulate the proposed algorithm Eq. 2.6 into a first-order ODE system.

Proposition 2.3. *As $\tau, \sigma \to 0$ and $\sigma\omega \to \gamma$, both updates in 2.6 and Algorithm 1 can be formulated as a discrete-time update of the following ODE system.*

$$\dot{\boldsymbol{p}} = \boldsymbol{A}(x)\nabla f(\boldsymbol{x}) - \varepsilon\boldsymbol{A}(x)\boldsymbol{p}\,, \tag{2.8a}$$
$$\dot{\boldsymbol{x}} = -\boldsymbol{C}(\boldsymbol{x})(\boldsymbol{p} + \gamma(\boldsymbol{A}(x)\nabla f(\boldsymbol{x}) - \varepsilon\boldsymbol{A}(x)\boldsymbol{p}))\,, \tag{2.8b}$$

where $\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x})$ and the initial condition satisfies $\boldsymbol{x}(0) = \boldsymbol{x}^0$, $\boldsymbol{p}(0) = \boldsymbol{p}^0$. Suppose that ∇f is Lispchitz continuous and each index in matrix $\boldsymbol{A}$, $\boldsymbol{C}$ is continuous and bounded. Then, there exists a unique solution for the ODE system Eq. 2.8. A stationary state $(\boldsymbol{x}^*, \boldsymbol{p}^*)$ of ODE system Eq. 2.8 satisfies

$$\nabla f(\boldsymbol{x}^*) = 0, \quad \boldsymbol{p}^* = 0.$$

Proof. Rearranging Eq. 2.6a and Eq. 2.6b, we have

$$\frac{\boldsymbol{p}^{n+1} - \boldsymbol{p}^n}{\sigma} = \boldsymbol{A}(\boldsymbol{x}^n)\nabla f(\boldsymbol{x}^n) - \varepsilon\boldsymbol{A}(\boldsymbol{x}^n)\boldsymbol{p}^{n+1}\,,$$
$$\frac{\boldsymbol{x}^{n+1} - \boldsymbol{x}^n}{\tau} = -\boldsymbol{B}(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^{n+1})\big((1+\omega)\sigma\boldsymbol{A}(\boldsymbol{x}^n)\nabla f(\boldsymbol{x}^n) + \boldsymbol{p}^n\big)\,.$$

Taking the limit as $\tau, \sigma \to 0$ and $\sigma\omega \to \gamma$, we obtain

$$\dot{\boldsymbol{p}} = \boldsymbol{A}(x)\nabla f(\boldsymbol{x}) - \varepsilon\boldsymbol{A}(x)\boldsymbol{p}\,,$$
$$\dot{\boldsymbol{x}} = -\boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x})(\boldsymbol{p} + \gamma(\boldsymbol{A}(x)\nabla f(\boldsymbol{x}) - \varepsilon\boldsymbol{A}(x)\boldsymbol{p}))\,.$$

Similarly, the update in Algorithm 1 also converges to the ODE system Eq. 2.8. Clearly, a stationary state satisfies $\boldsymbol{p}^* = 0$, $\nabla f(\boldsymbol{x}^*) = 0$. □

Proposition 2.4 (Primal-dual damping second order ODE)**.** *The ODE system Eq. 2.8 satisfies the following second-order ODE*

$$\ddot{\boldsymbol{x}} + \left[\varepsilon \boldsymbol{A} + \gamma \boldsymbol{C}\boldsymbol{A}\nabla^2 f(\boldsymbol{x}) - \dot{\boldsymbol{C}}\boldsymbol{C}^{-1}\right]\dot{\boldsymbol{x}} + \boldsymbol{C}\boldsymbol{A}\nabla f(\boldsymbol{x}) = 0\,. \tag{2.9}$$

Here $\dot{\boldsymbol{C}} = \frac{d}{dt}\boldsymbol{C}(\boldsymbol{x}(t))$.

The proof follows by direct calculations and can be found in Appendix C. We note that the formulation given by Eq. 2.9 includes several important special cases in the literature. In a word, we view Eq. 2.4 as a preconditioned accelerated gradient flow.

Example 2.1. Let $\boldsymbol{C} = \boldsymbol{A} = \mathbb{I}$ and $\gamma \neq 0$. Then equation Eq. 2.4 satisfies

$$\ddot{\boldsymbol{x}} + \epsilon\dot{\boldsymbol{x}} + \gamma\nabla^2 f(\boldsymbol{x})\dot{\boldsymbol{x}} + \nabla f(\boldsymbol{x}) = 0\,, \tag{2.10}$$

which is an inertial system with Hessian-driven damping [Attouch et al., 2020].

Remark 2.5. In the case of $\boldsymbol{C} = \boldsymbol{A} = \mathbb{I}$, although the derived second order ODE Eq. 2.9 is the same as the one in Attouch et al. [2020] at a continuous time level, our algorithm 1 provides a different time discretization from the one in Attouch et al. [2020].

Example 2.2. Let $\boldsymbol{C} = \boldsymbol{A} = \mathbb{I}$, $\gamma(t) = 0$. Then equation Eq. 2.4 satisfies the heavy ball ODE [Siegel, 2019]

$$\ddot{\boldsymbol{x}} + \varepsilon\dot{\boldsymbol{x}} + \nabla f(\boldsymbol{x}) = 0\,. \tag{2.11}$$

Example 2.3. Let $\boldsymbol{C} = \boldsymbol{A} = \mathbb{I}$, $\gamma(t) = 0$, $\varepsilon(t) = \frac{3}{t}$. Then equation Eq. 2.4 satisfies the Nesterov ODE [Su et al., 2015]:

$$\ddot{\boldsymbol{x}} + \frac{3}{t}\dot{\boldsymbol{x}} + \nabla f(\boldsymbol{x}) = 0\,. \tag{2.12}$$

We next provide a convergence analysis of ODE Eq. 2.8 for quadratic optimization problems. We demonstrate the importance of preconditioners in characterizing the convergence speed of ODE Eq. 2.8.

Theorem 2.6. *Suppose* $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}$ *for some symmetric positive definite matrix* $\boldsymbol{Q} \in \mathbb{R}^{d\times d}$. *Assume* $\boldsymbol{A}$, $\boldsymbol{B}$ *are constant matrices. In this case, equation Eq. 2.8 satisfies the linear ODE system:*

$$\begin{pmatrix}\dot{\boldsymbol{x}}\\ \dot{\boldsymbol{p}}\end{pmatrix} = \begin{pmatrix}-\gamma\boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q} & -\boldsymbol{B}\boldsymbol{Q}(\mathbb{I} - \gamma\varepsilon\boldsymbol{A})\\ \boldsymbol{A}\boldsymbol{Q} & -\varepsilon\boldsymbol{A}\end{pmatrix}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{p}\end{pmatrix}.$$

Suppose that $\boldsymbol{A}$ *commutes with* $\boldsymbol{Q}$, *such that* $\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{A}$. *Suppose* $\boldsymbol{A}$ *and* $\boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q}$ *are simultaneously diagonalizable and have positive eigenvalues. Let* $\mu_1 \geq \ldots \geq \mu_n > 0$ *be the eigenvalues of* $\boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q}$ *and* a_i *the i-th eigenvalue of* $\boldsymbol{A}$ *(not necessarily in descending order) in the same basis. Then*

(a) The solution of ODE system 2.8 converges to $(\boldsymbol{x}^*, \boldsymbol{p}^*) = (0, 0)$*:*

$$\|(\boldsymbol{x}(t), \boldsymbol{p}(t))\| \leq \|(\boldsymbol{x}_0, \boldsymbol{p}_0)\|\exp(\alpha t)\,,$$

where

$$\alpha = \max_i \frac{1}{2}\big[-\gamma\mu_i - \varepsilon a_i + \Re\big(\sqrt{(\gamma\mu_i + \varepsilon)^2 - 4\mu_i}\big)\big]\,.$$

(b) When $\boldsymbol{A} = \mathbb{I}, \varepsilon = 0$, *the optimal convergence rate is achieved at* $\gamma^* = \frac{2\sqrt{\mu_1}}{\sqrt{\mu_n(2\mu_1 - \mu_n)}}$. *The corresponding rate is* $\alpha = \frac{-\sqrt{\mu_n}}{\sqrt{2-\frac{1}{\kappa}}}$, *where* $\kappa = \mu_1/\mu_n > 1$.

(c) Moreover, when $\gamma = \varepsilon = 0$, *the system will not converge for any initial data* $(\boldsymbol{x}_0, \boldsymbol{p}_0) \neq (0, 0)$.

(d) If $\boldsymbol{A} = \mathbb{I}$, $\gamma \leq \frac{1}{\sqrt{\mu_1}}$, $\varepsilon = 2\sqrt{\mu'} - \gamma\mu'$ *for some* $\mu' \leq \mu_n$, *then*

$$\alpha = -\sqrt{\mu'} - \frac{\gamma}{2}(\mu_n - \mu') \leq -\sqrt{\mu'}\,.$$

We defer the proof to Appendix B.

Remark 2.7. If ω is bounded, then we have $\gamma = \mathcal{O}(\sigma)$. Then, in the limit as $\sigma \to 0$, we also have that $\gamma \to 0$. By Theorem 2.6 (c), the ODE system 2.8 does not converge for any initial data.

Remark 2.8. If μ' is an estimate of the smallest eigenvalue μ_n, then the convergence speed for the solution of heavy ball ODE is $\exp(-\sqrt{\mu'}t)$. In Theorem 2.6 (d), if $\gamma = 0$ and $\mu' = \mu_n$, then $\alpha = -\sqrt{\mu_n}$ which is the same as the convergence rate of the heavy ball ODE [Siegel, 2019]. However, if $\gamma > 0$ and $\mu' < \mu_n$, then we have $\alpha = -\sqrt{\mu'} - \gamma(\mu_n - \mu') < -\sqrt{\mu'}$, which converges faster than the heavy ball ODE.

3 Lyapunov Analysis

In this section, we present the main theoretical result of this paper. We provide the convergence analysis for general objective functions in both continuous-time ODEs Eq. 2.8 and discrete-time Algorithm 1. From now on, we make the following two assumptions for the convergence analysis.

Assumption 3.1. There exists two constants $L \geq \mu > 0$ such that $\mu\mathbb{I} \preceq \boldsymbol{C}_0(\boldsymbol{x}) \preceq L\mathbb{I}$ for all $\boldsymbol{x}$, where $\boldsymbol{C}_0(\boldsymbol{x}) = \nabla^2 f(\boldsymbol{x})\boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x})$, and $\mu \leq 1$.

Assumption 3.2. There exists a constant $L' > 0$ such that

$$\boldsymbol{C}(\boldsymbol{x})^T\big(\nabla^3 f(\boldsymbol{x})\nabla f(\boldsymbol{x}) + (\nabla^2 f(\boldsymbol{x}))^2\big)\boldsymbol{C}(\boldsymbol{x}) \preceq L'\mathbb{I} \tag{3.1}$$

for all $\boldsymbol{x}$, where $\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x})$.

3.1 Continuous time Lyapunov analysis

In this subsection, we establish convergence results of the ODE system Eq. 2.8.

Theorem 3.3. *Consider the ODE system Eq. 2.8 with an initial condition* $(x(0), p(0)) \in \mathbb{R}^{2d}$. *Define the functional*

$$\mathcal{I}(\boldsymbol{x}, \boldsymbol{p}) = \frac{1}{2}(\|\boldsymbol{p}\|^2 + \|\nabla f(\boldsymbol{x})\|^2)\,. \tag{3.2}$$

Suppose Assumption 3.1 holds, we have

$$\mathcal{I}(\boldsymbol{x}(t), \boldsymbol{p}(t)) \leq \mathcal{I}(\boldsymbol{x}(0), \boldsymbol{p}(0))\exp(-2\lambda t)\,, \tag{3.3}$$

where

$$\begin{aligned}\lambda = \min\Big\{&\mu\gamma A - \frac{1}{2}|A - \mu(1-\varepsilon\gamma A)|, L\gamma A - \frac{1}{2}|A - L(1-\varepsilon\gamma A)|,\\ &\varepsilon A - \frac{1}{2}|A - \mu(1-\varepsilon\gamma A)|, \varepsilon A - \frac{1}{2}|A - L(1-\varepsilon\gamma A)|\Big\}\end{aligned}$$

In particular, when $\gamma = \frac{1}{\mu}, \varepsilon = 1, A = \frac{\mu+L}{2+(\mu+L)\varepsilon\gamma}$, *then* $\lambda = \frac{\mu}{2}$.

Proof. It is straightforward to compute the following

$$\begin{aligned}\frac{\mathrm{d}\mathcal{I}}{\mathrm{d}t} &= \langle \boldsymbol{p}, \dot{\boldsymbol{p}}\rangle + \langle \nabla f, \nabla^2 f\dot{\boldsymbol{x}}\rangle\\ &= -\nabla f^T\boldsymbol{C}_0\gamma\boldsymbol{A}\nabla f - \boldsymbol{p}^T\varepsilon\boldsymbol{A}\boldsymbol{p} + \nabla f^T\big(\boldsymbol{A} - \boldsymbol{C}_0(\mathbb{I} - \varepsilon\gamma\boldsymbol{A})\big)\boldsymbol{p}\end{aligned} \tag{3.4}$$

We shall find λ such that $\frac{\mathrm{d}\mathcal{I}}{\mathrm{d}t} + 2\lambda\mathcal{I} \leq 0$. Then we obtain the exponential convergence by Gronwall's inequality, i.e.,

$$\mathcal{I}(\boldsymbol{x}(t), \boldsymbol{p}(t)) \leq \mathcal{I}(\boldsymbol{x}(0), \boldsymbol{p}(0))\exp(-2\lambda t)\ .$$

We can compute

$$\frac{\mathrm{d}\mathcal{I}}{\mathrm{d}t} + 2\lambda\mathcal{I} = \nabla f^T\big(-\boldsymbol{C}_0\gamma\boldsymbol{A} + \lambda\mathbb{I}\big)\nabla f + \boldsymbol{p}^T\big(-\varepsilon\boldsymbol{A} + \lambda\mathbb{I}\big)\boldsymbol{p} + \nabla f^T\big(\boldsymbol{A} - \boldsymbol{C}_0(\mathbb{I} - \varepsilon\gamma\boldsymbol{A})\big)\boldsymbol{p}\,. \tag{3.5}$$

By Lemma A.1, we obtain the following sufficient conditions for $\frac{\mathrm{d}\mathcal{I}}{\mathrm{d}t} + 2\lambda\mathcal{I} \le 0$

$$-\varepsilon A + \lambda + \frac{1}{2}|\xi_i(1-\varepsilon\gamma A) - A| \le 0 \tag{3.6a}$$

$$\lambda - \xi_i\gamma A + \frac{1}{2}|\xi_i(1-\varepsilon\gamma A) - A| \le 0 \tag{3.6b}$$

where $\xi_i(\boldsymbol{x})$ is the eigenvalue of $\boldsymbol{C}_0(\boldsymbol{x})$. By our assumptions, we have $L \ge \xi_1(\boldsymbol{x}) \ge \ldots \ge \xi_n(\boldsymbol{x}) \ge \mu$. Eq. 3.6 give two upper bounds on λ. Define $g_1(\xi) = \varepsilon A + \frac{1}{2}|\xi(1-\varepsilon\gamma A) - A|$, and $g_2(\xi) = \xi\gamma A - \frac{1}{2}|\xi(1-\varepsilon\gamma A) - A|$ on the interval $[\mu, L]$. Then Eq. 3.6 implies that

$$\lambda \le g_j(\xi_i)\,, \tag{3.7}$$

for all $i = 1, \ldots, n$ and $j = 1, 2$. Since each $g_j(\xi)$ is a piece-wise linear in ξ, it is not hard to see that

$$\min_{\xi\in[\mu,L]} g_j(\xi) = \min\{g_j(\mu), g_j(L)\}\,,$$

for $j = 1, 2$. This proves the formula for λ. When $A = \frac{\mu+L}{2+(\mu+L)\varepsilon\gamma}$, we have $g_1(\mu) = g_1(L)$, and

$$\mu(1-\varepsilon\gamma A) - A = -L(1-\varepsilon\gamma A) + A\,.$$

Further, requiring $g_1(\mu) = g_2(\mu)$ yields $\varepsilon = \mu\gamma$. And we obtain

$$\begin{aligned}\lambda &= \mu\gamma A - \frac{1}{2}|A - \mu(1-\varepsilon\gamma\boldsymbol{A})| \\ &= \mu\gamma A - \frac{1}{2}(A - \mu(1-\varepsilon\gamma A)) \\ &= \frac{\mu}{2} + A(\gamma\mu - \frac{1}{2}\gamma^2\mu^2 - \frac{1}{2}) \\ &= \frac{\mu}{2} - \frac{A}{2}(\gamma\mu - 1)^2.\end{aligned} \tag{3.8}$$

We note that λ is maximized when taking $\gamma = \mu^{-1}$. We obtain $\lambda = \frac{\mu}{2}$.

□

3.2 Discrete time Lyapunov analysis

In this subsection, we study the convergence criterion for the discretized linearized PDHG flow given by Eq. 1.2 and Algorithm 1.

From now on, we assume that f is a $\mathcal{C}^4$ strongly convex function.

We can rewrite the iterations as

$$\boldsymbol{p}^{n+1} = \frac{1}{1+\sigma\varepsilon A}\boldsymbol{p}^n + \frac{\sigma A}{1+\sigma\varepsilon A}\nabla f(\boldsymbol{x}^n)\,, \tag{3.9a}$$

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - \tau\boldsymbol{B}(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^n)\left(\frac{1-\varepsilon\gamma A}{1+\sigma\varepsilon A}\boldsymbol{p}^n + \frac{\sigma A + \gamma A}{1+\sigma\varepsilon A}\nabla f(\boldsymbol{x}^n)\right)\,, \tag{3.9b}$$

where $\gamma = \sigma\omega$. We define the following notations which will be used later.

$$\boldsymbol{N}(\boldsymbol{x}^n) = \frac{1}{1+\sigma\varepsilon A}\begin{pmatrix}\boldsymbol{B}(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^n)(\sigma A + \gamma A) & \boldsymbol{B}(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^n)(1-\varepsilon\gamma A) \\ -\frac{\sigma}{\tau}A & \frac{\sigma}{\tau}\varepsilon A\end{pmatrix}\,. \tag{3.10}$$

And

$$\boldsymbol{H}(\boldsymbol{x}^n) = \mathrm{sym}\left(\begin{pmatrix}\nabla^2 f(\boldsymbol{x}^n) & 0\\ 0 & \mathbb{I}\end{pmatrix}\cdot \boldsymbol{N}(\boldsymbol{x}^n)\right).$$

Remark 3.4. The matrix $\boldsymbol{N}(\boldsymbol{x}^n)$ and $\boldsymbol{H}(\boldsymbol{x}^n)$ also depends on the τ, σ, A, ε and ω.

Define the Lyapunov functional in discrete time as

$$\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n) = \frac{1}{2}\|\nabla f(\boldsymbol{x}^n)\|^2 + \frac{1}{2}\|\boldsymbol{p}^n\|^2.$$

Theorem 3.5. *Suppose that there exists positive constants $\lambda, M_1 \in \mathbb{R}_+$, such that*

$$\boldsymbol{H}(\boldsymbol{x}) \succeq \lambda\mathbb{I},$$
$$\boldsymbol{N}(\boldsymbol{x})^T\nabla^2\mathcal{I}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{p}})\boldsymbol{N}(\boldsymbol{x}) \preceq M_1\mathbb{I},$$

for all $\boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathbb{R}^n$. If $\tau = a\frac{\lambda}{M}$ for some $a \in (0,2)$, then the functional $\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n)$ decreases geometrically, i.e.

$$\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n) \le \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0)\big(1 + (a^2 - 2a)\frac{\lambda^2}{2M_1}\big)^n.$$

Proof. It follows from our definition of $\boldsymbol{N}(\boldsymbol{x}^n)$ that

$$\begin{pmatrix}\boldsymbol{x}^{n+1} - \boldsymbol{x}^n\\ \boldsymbol{p}^{n+1} - \boldsymbol{p}^n\end{pmatrix} = -\tau\boldsymbol{N}(\boldsymbol{x}^n)\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix}, \tag{3.11}$$

By the mean-value theorem, we obtain

$$\begin{aligned}&\mathcal{I}(\boldsymbol{x}^{n+1}, \boldsymbol{p}^{n+1}) - \mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n)\\ =&\begin{pmatrix}\nabla_{\boldsymbol{x}}\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n)\\ \nabla_{\boldsymbol{p}}\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n)\end{pmatrix}^T\begin{pmatrix}\boldsymbol{x}^{n+1} - \boldsymbol{x}^n\\ \boldsymbol{p}^{n+1} - \boldsymbol{p}^n\end{pmatrix} + \frac{1}{2}\begin{pmatrix}\boldsymbol{x}^{n+1} - \boldsymbol{x}^n\\ \boldsymbol{p}^{n+1} - \boldsymbol{p}^n\end{pmatrix}^T\nabla^2\mathcal{I}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{p}})\begin{pmatrix}\boldsymbol{x}^{n+1} - \boldsymbol{x}^n\\ \boldsymbol{p}^{n+1} - \boldsymbol{p}^n\end{pmatrix}\end{aligned}$$

where $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{p}})$ is in between $(\boldsymbol{x}^{n+1}, \boldsymbol{p}^{n+1})$ and $(\boldsymbol{x}^n, \boldsymbol{p}^n)$. And

$$\begin{aligned}&\nabla_{\boldsymbol{x}}\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n) = \nabla^2 f(\boldsymbol{x}^n)\nabla f(\boldsymbol{x}^n),\\ &\nabla_{\boldsymbol{p}}\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n) = \boldsymbol{p}^n,\\ &\nabla^2\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n) = \begin{pmatrix}\nabla^3 f(\boldsymbol{x}^n)\nabla f(\boldsymbol{x}^n) + \nabla^2 f(\boldsymbol{x}^n)\nabla^2 f(\boldsymbol{x}^n) & 0\\ 0 & \mathbb{I}\end{pmatrix}.\end{aligned}$$

Then using Eq. 3.11 and definition of $\boldsymbol{H}(\boldsymbol{x}^n)$, we obtain

$$\begin{aligned}&\mathcal{I}(\boldsymbol{x}^{n+1}, \boldsymbol{p}^{n+1}) - \mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n)\\ =&-\tau\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix}^T\begin{pmatrix}\nabla^2 f(\boldsymbol{x}^n) & 0\\ 0 & \mathbb{I}\end{pmatrix}\cdot\boldsymbol{N}(\boldsymbol{x}^n)\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix}\\ &+\frac{\tau^2}{2}\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix}^T\boldsymbol{N}(\boldsymbol{x}^n)^T\nabla^2\mathcal{I}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{p}})\boldsymbol{N}(\boldsymbol{x}^n)\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix}\\ =&-\tau\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix}^T\boldsymbol{H}(\boldsymbol{x}^n)\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix}\\ &+\frac{\tau^2}{2}\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix}^T\boldsymbol{N}(\boldsymbol{x}^n)^T\nabla^2\mathcal{I}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{p}})\boldsymbol{N}(\boldsymbol{x}^n)\begin{pmatrix}\nabla f(\boldsymbol{x}^n)\\ \boldsymbol{p}^n\end{pmatrix},\end{aligned} \tag{3.12}$$

From Eq. 3.12 and our assumption on $\boldsymbol{N}(\boldsymbol{x})$ and $\boldsymbol{H}(\boldsymbol{x})$, we obtain

$$\begin{aligned}\mathcal{I}(\boldsymbol{x}^{n+1},\boldsymbol{p}^{n+1})-\mathcal{I}(\boldsymbol{x}^{n},\boldsymbol{p}^{n}) &\leq \big(-\tau\lambda+\frac{\tau^2 M_1}{2}\big)\mathcal{I}(\boldsymbol{x}^{n},\boldsymbol{p}^{n})\\ &= \frac{M_1}{2}\big((\tau-\frac{\lambda}{M_1})^2-\frac{\lambda^2}{M_1^2}\big)\mathcal{I}(\boldsymbol{x}^{n},\boldsymbol{p}^{n})\\ &= (a^2-2a)\frac{\lambda^2}{2M_1}\mathcal{I}(\boldsymbol{x}^{n},\boldsymbol{p}^{n})\,, \end{aligned} \tag{3.13}$$

where we used $\tau = a\frac{\lambda}{M_1}$. Hence,

$$\mathcal{I}(\boldsymbol{x}^{n+1},\boldsymbol{p}^{n+1}) \leq \mathcal{I}(\boldsymbol{x}^{n},\boldsymbol{p}^{n})\big(1+(a^2-2a)\frac{\lambda^2}{2M_1}\big) \leq \mathcal{I}(\boldsymbol{x}^{0},\boldsymbol{p}^{0})\big(1+(a^2-2a)\frac{\lambda^2}{2M_1}\big)^{n+1}.$$

When $0<a<2$, we have $a^2-2a<0$. Thus we obtain the desired convergence result. □

Theorem 3.6. *Let $f:\mathbb{R}^d\to\mathbb{R}$ be a $\mathcal{C}^4$ strongly convex function. Suppose $(\boldsymbol{x}^0,\boldsymbol{p}^0)$ satisfies*

$$\mathcal{I}(\boldsymbol{x}^0,\boldsymbol{p}^0)^{1/2} \leq \frac{\delta}{\tau D_0 \|\boldsymbol{N}(\boldsymbol{x})\|_2^3}\,, \tag{3.14}$$

for some $\delta>0$ and all $\boldsymbol{x}$. Here

$$D_0 = \sup_{\boldsymbol{x},\boldsymbol{p},\boldsymbol{x}',\boldsymbol{p}'} \frac{\begin{pmatrix}\boldsymbol{x}'\\ \boldsymbol{p}'\end{pmatrix}^T\left(\nabla^3\mathcal{I}(\boldsymbol{x},\boldsymbol{p})\begin{pmatrix}\boldsymbol{x}'\\ \boldsymbol{p}'\end{pmatrix}\right)\begin{pmatrix}\boldsymbol{x}'\\ \boldsymbol{p}'\end{pmatrix}}{\left\|\begin{pmatrix}\boldsymbol{x}'\\ \boldsymbol{p}'\end{pmatrix}\right\|_2^3}\,.$$

Suppose further that there exists positive constants $\lambda, M_2\in\mathbb{R}_+$ such that

$$\begin{aligned}\boldsymbol{H}(\boldsymbol{x}) &\succeq \lambda\mathbb{I}\,,\\ \boldsymbol{N}(\boldsymbol{x})^T\nabla^2\mathcal{I}(\boldsymbol{x},\boldsymbol{p})\boldsymbol{N}(\boldsymbol{x}) &\preceq M_2\mathbb{I}\end{aligned}$$

for all $\boldsymbol{x}\in\mathbb{R}^n$. If $\tau = a\frac{\lambda}{M_2+\delta}$ for some $a\in(0,2)$, then the functional $\mathcal{I}(\boldsymbol{x}^n,\boldsymbol{p}^n)$ decreases geometrically, i.e.

$$\mathcal{I}(\boldsymbol{x}^n,\boldsymbol{p}^n) \leq \mathcal{I}(\boldsymbol{x}^0,\boldsymbol{p}^0)\big(1+\frac{a^2-2a}{2}\frac{\lambda^2}{M_2+\delta}\big)^n\,.$$

Remark 3.7. Note that the constant M_2 in Theorem 3.6 can be better than the constant M_1 in Theorem 3.5 because $\boldsymbol{N}$ and $\nabla^2\mathcal{I}$ are evaluated at the same $\boldsymbol{x}$ in Theorem 3.6.

Proof. We will prove it by induction. Using the mean-value theorem, we have

$$\begin{aligned}&\mathcal{I}(\boldsymbol{x}^{n+1},\boldsymbol{p}^{n+1})-\mathcal{I}(\boldsymbol{x}^{n},\boldsymbol{p}^{n})\\ =&\begin{pmatrix}\nabla_{\boldsymbol{x}}\mathcal{I}(\boldsymbol{x}^n,\boldsymbol{p}^n)\\ \nabla_{\boldsymbol{p}}\mathcal{I}(\boldsymbol{x}^n,\boldsymbol{p}^n)\end{pmatrix}^T\begin{pmatrix}\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\\ \boldsymbol{p}^{n+1}-\boldsymbol{p}^n\end{pmatrix}+\frac{1}{2}\begin{pmatrix}\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\\ \boldsymbol{p}^{n+1}-\boldsymbol{p}^n\end{pmatrix}^T\nabla^2\mathcal{I}(\boldsymbol{x}^n,\boldsymbol{p}^n)\begin{pmatrix}\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\\ \boldsymbol{p}^{n+1}-\boldsymbol{p}^n\end{pmatrix}\\ &+\frac{1}{6}\begin{pmatrix}\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\\ \boldsymbol{p}^{n+1}-\boldsymbol{p}^n\end{pmatrix}^T\left(\nabla^3\mathcal{I}(\tilde{\boldsymbol{x}}^n,\tilde{\boldsymbol{p}}^n)\begin{pmatrix}\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\\ \boldsymbol{p}^{n+1}-\boldsymbol{p}^n\end{pmatrix}\right)\begin{pmatrix}\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\\ \boldsymbol{p}^{n+1}-\boldsymbol{p}^n\end{pmatrix}\,,\end{aligned} \tag{3.15}$$

where $(\tilde{\boldsymbol{x}}^n, \tilde{\boldsymbol{p}}^n)$ is in between $(\boldsymbol{x}^{n+1}, \boldsymbol{p}^{n+1})$ and $(\boldsymbol{x}^n, \boldsymbol{p}^n)$. By Eq. 3.15 and Eq. 3.11, we can bound

$$
\begin{aligned}
&\mathcal{I}(\boldsymbol{x}^1, \boldsymbol{p}^1) - \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \\
=& -\tau \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{H}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
&+ \frac{\tau^2}{2} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{N}(\boldsymbol{x}^0)^T \nabla^2 \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \boldsymbol{N}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
&- \frac{\tau^3}{6} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{N}(\boldsymbol{x}^0)^T \left(\nabla^3 \mathcal{I}(\tilde{\boldsymbol{x}}^0, \tilde{\boldsymbol{p}}^0) \boldsymbol{N}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \right) \boldsymbol{N}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
\leq& -\tau \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{H}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
&+ \frac{\tau^2}{2} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{N}(\boldsymbol{x}^0)^T \nabla^2 \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \boldsymbol{N}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
&+ \frac{\tau^3}{6} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \left(D_0 \|\boldsymbol{N}(\boldsymbol{x}^0)\|_2^3 \left\| \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \right\|_2 \right) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
=& -\tau \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{H}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
&+ \frac{\tau^2}{2} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{N}(\boldsymbol{x}^0)^T \nabla^2 \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \boldsymbol{N}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
&+ \frac{\tau^3}{6} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T D_0 \|\boldsymbol{N}(\boldsymbol{x}^0)\|_2^3 \mathcal{I}(\boldsymbol{x}_0, \boldsymbol{p}_0)^{1/2} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
\leq& -\tau \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{H}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
&+ \frac{\tau^2}{2} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \boldsymbol{N}(\boldsymbol{x}^0)^T \nabla^2 \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \boldsymbol{N}(\boldsymbol{x}^0) \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix} \\
&+ \frac{\tau^2 \delta}{6} \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}^T \begin{pmatrix} \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{p}^0 \end{pmatrix}, \qquad (3.16)
\end{aligned}
$$

where the last inequality is by our assumption on $(\boldsymbol{x}^0, \boldsymbol{p}^0)$. Using our assumptions on the lower bound of $\boldsymbol{H}$ and the upper bound of $\boldsymbol{N}^T \cdot \nabla^2 \mathcal{I} \cdot \boldsymbol{N}$, we obtain

$$
\begin{aligned}
\mathcal{I}(\boldsymbol{x}^1, \boldsymbol{p}^1) - \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) &\leq \big(-\tau\lambda + \frac{\tau^2 \delta}{6} + \frac{\tau^2 M_2}{2} \big) \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \\
&\leq \big(-\tau\lambda + \frac{\tau^2 (\delta + M_2)}{2} \big) \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \\
&= \frac{1}{2}(a^2 - 2a) \frac{\lambda^2}{M_2 + \delta} \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0), \qquad (3.17)
\end{aligned}
$$

where we used $\tau = a \frac{\lambda}{M_2 + \delta}$ for some $a \in (0, 2)$. Hence,

$$
\mathcal{I}(\boldsymbol{x}^1, \boldsymbol{p}^1) \leq \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \big(1 + \frac{a^2 - 2a}{2} \frac{\lambda^2}{M_2 + \delta} \big).
$$

This proves the base case. Now suppose it holds that

$$
\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n) \leq \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0) \big(1 + \frac{a^2 - 2a}{2} \frac{\lambda^2}{M_2 + \delta} \big)^n,
$$

for some $n \geq 1$. In particular, this implies that

$$\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n) < \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0)\,,$$

which yields

$$\tau D_0 \|\boldsymbol{N}(\boldsymbol{x})\|_2^3 \mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n)^{1/2} < \tau D_0 \|\boldsymbol{N}(\boldsymbol{x})\|_2^3 \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0)^{1/2} \leq \delta\,.$$

Then, repeating the derivation of Eq. 3.16 and Eq. 3.17 yields

$$\mathcal{I}(\boldsymbol{x}^{n+1}, \boldsymbol{p}^{n+1}) \leq \mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n)\big(1 + \frac{a^2 - 2a}{2} \frac{\lambda^2}{M_2 + \delta}\big)\,.$$

Combining with our induction hypothesis, we conclude that

$$\mathcal{I}(\boldsymbol{x}^{n+1}, \boldsymbol{p}^{n+1}) \leq \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0)\big(1 + \frac{a^2 - 2a}{2} \frac{\lambda^2}{M_2 + \delta}\big)^{n+1}\,.$$

The proof is complete by induction. □

Corollary 3.8. *Suppose Assumption 3.1 and Assumption 3.2 hold. When* $\sigma = \tau$, $\gamma = \frac{1-\sigma\mu}{\mu}, \varepsilon = 1, A = \frac{\mu+L}{2+(\mu+L)\varepsilon\gamma}$, *we have*

$$\boldsymbol{H}(\boldsymbol{x}) \succeq \frac{\mu}{4}\mathbb{I}.$$

Proof. By definition of $\boldsymbol{H}$, we can compute

$$(1 + \sigma\varepsilon A) \cdot \boldsymbol{H}(\boldsymbol{x}) = \begin{pmatrix} \boldsymbol{C}_0(\boldsymbol{x})(\sigma\boldsymbol{A} + \gamma\boldsymbol{A}) & \frac{1}{2}\boldsymbol{C}_0(\boldsymbol{x})(1 - \varepsilon\gamma A) - \frac{1}{2}\eta\boldsymbol{A} \\ \frac{1}{2}\boldsymbol{C}_0(\boldsymbol{x})(1 - \varepsilon\gamma A) - \frac{1}{2}\eta\boldsymbol{A} & \eta\varepsilon\boldsymbol{A} \end{pmatrix}\,,$$

where $\eta = \sigma/\tau = 1$, $\boldsymbol{C}_0(\boldsymbol{x}) = \nabla^2 f(\boldsymbol{x})\boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x})$. We want to find some constant $\lambda > 0$, such that

$$\begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{w} \end{pmatrix}^T \boldsymbol{H}(\boldsymbol{x}) \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{w} \end{pmatrix} \geq \lambda(\|\boldsymbol{z}\|^2 + \|\boldsymbol{w}\|^2)\,.$$

Observe that

$$\begin{aligned} &\begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{w} \end{pmatrix}^T \boldsymbol{H}(\boldsymbol{x}) \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{w} \end{pmatrix} - \lambda(\|\boldsymbol{z}\|^2 + \|\boldsymbol{w}\|^2) \\ &= \boldsymbol{z}^T\big(\boldsymbol{C}_0(\gamma + \sigma)A/(1 + \sigma\varepsilon A) - \lambda\mathbb{I}\big)\boldsymbol{z} + \boldsymbol{w}^T\big(\varepsilon A/(1 + \sigma\varepsilon A) - \lambda\mathbb{I}\big)\boldsymbol{w} \\ &\quad + \boldsymbol{z}^T\big(-A + \boldsymbol{C}_0(\mathbb{I} - \varepsilon\gamma A)\big)\boldsymbol{w}/(1 + \sigma\varepsilon A)\,, \end{aligned} \tag{3.18}$$

which is almost the same as Eq. 3.5. Thus, following a similar procedure in Theorem 3.3 with the provided parameters, we obtain that

$$\lambda \geq \frac{\mu}{2}\frac{1 + \frac{A\sigma}{2}}{1 + \sigma A} \geq \frac{\mu}{4}\,.$$

This implies

$$\boldsymbol{H}(\boldsymbol{x}) \succeq \frac{\mu}{4}\mathbb{I}\,. \tag{3.19}$$

□

Corollary 3.9. *Let* $f : \mathbb{R}^d \to \mathbb{R}$ *be a* $\mathcal{C}^4$ *strongly convex function. Suppose Assumption 3.1 and Assumption 3.2 hold. If* $\sigma = \tau$, $\gamma = \frac{1-\sigma\mu}{\mu}, \varepsilon = 1, A = \frac{\mu+L}{2+(\mu+L)\varepsilon\gamma}$, *we have*

(1)

$$\|\boldsymbol{N}(\boldsymbol{x})\|_2 \leq \frac{\max\{L, 1\} \cdot \big(A(\sigma + 2\gamma + 2) + 1\big)}{(1 + \sigma A)}\,.$$

(2)

$$\boldsymbol{N}(\boldsymbol{x})^T\nabla^2\mathcal{I}(\boldsymbol{x},\boldsymbol{p})\boldsymbol{N}(\boldsymbol{x}) \preceq \frac{(3+\sigma A+2A)^2}{(1+\sigma A)^2}\cdot\max\{L',1\}\cdot\mathbb{I}\,.$$

Proof. We can decompose

$$(1+\sigma A)\cdot\boldsymbol{N}(\boldsymbol{x}) = \begin{pmatrix}\boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x}) & 0\\ 0 & \mathbb{I}\end{pmatrix}\begin{pmatrix}(\sigma+\gamma)\boldsymbol{A} & (\mathbb{I}-\gamma\boldsymbol{A})\\ -\boldsymbol{A} & \boldsymbol{A}\end{pmatrix}.$$

Observe that

$$\begin{pmatrix}(\sigma+\gamma)\boldsymbol{A} & (\mathbb{I}-\gamma\boldsymbol{A})\\ -\boldsymbol{A} & \boldsymbol{A}\end{pmatrix} = \begin{pmatrix}(\sigma+\gamma)\mathbb{I} & -\gamma\mathbb{I}\\ -\mathbb{I} & \mathbb{I}\end{pmatrix}\cdot\begin{pmatrix}\boldsymbol{A} & 0\\ 0 & \boldsymbol{A}\end{pmatrix} + \begin{pmatrix}0 & \mathbb{I}\\ 0 & 0\end{pmatrix}.$$

Therefore,

$$\begin{aligned}\left\|\begin{pmatrix}(\sigma+\gamma)\boldsymbol{A} & (\mathbb{I}-\gamma\boldsymbol{A})\\ -\boldsymbol{A} & \boldsymbol{A}\end{pmatrix}\right\|_2 &\le A\left\|\begin{pmatrix}(\sigma+\gamma)\mathbb{I} & -\gamma\mathbb{I}\\ -\mathbb{I} & \mathbb{I}\end{pmatrix}\right\|_2 + \left\|\begin{pmatrix}0 & \mathbb{I}\\ 0 & 0\end{pmatrix}\right\|_2\\ &\le A\left\|\begin{pmatrix}(\sigma+\gamma)\mathbb{I} & -\gamma\mathbb{I}\\ -\mathbb{I} & \mathbb{I}\end{pmatrix}\right\|_2 + 1\\ &\le A(\sigma+2\gamma+2)+1\,. \end{aligned} \tag{3.20}$$

To get the last inequality, we consider $(\boldsymbol{z},\boldsymbol{w})$ such that $\|\boldsymbol{z}\|^2+\|\boldsymbol{w}\|^2=1$. Thus

$$\begin{aligned}\left\|\begin{pmatrix}(\sigma+\gamma)\mathbb{I} & -\gamma\mathbb{I}\\ -\mathbb{I} & \mathbb{I}\end{pmatrix}\begin{pmatrix}\boldsymbol{z}\\ \boldsymbol{w}\end{pmatrix}\right\|_2 &= \left\|\begin{pmatrix}(\sigma+\gamma)\boldsymbol{z}-\gamma\boldsymbol{w}\\ -\boldsymbol{z}+\boldsymbol{w}\end{pmatrix}\right\|\\ &\le \sigma\left\|\begin{pmatrix}\boldsymbol{z}\\ \boldsymbol{w}\end{pmatrix}\right\| + \gamma\|\boldsymbol{z}-\boldsymbol{w}\| + \|\boldsymbol{z}-\boldsymbol{w}\|\\ &\le \sigma+2\gamma+2\,.\end{aligned}$$

We now have

$$\begin{aligned}(1+\sigma A)\|\boldsymbol{N}(\boldsymbol{x})\|_2 &\le \left\|\begin{pmatrix}\boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x}) & 0\\ 0 & \mathbb{I}\end{pmatrix}\right\|_2\left\|\begin{pmatrix}(\sigma+\gamma)\boldsymbol{A} & (\mathbb{I}-\gamma\boldsymbol{A})\\ -\boldsymbol{A} & \boldsymbol{A}\end{pmatrix}\right\|_2\\ &\le \max\{L,1\}\cdot\big(A(\sigma+2\gamma+2)+1\big)\,.\end{aligned}$$

This proves part (1) of our Corollary. It follows that

$$\begin{aligned}&\boldsymbol{N}(\boldsymbol{x})^T\nabla^2\mathcal{I}(\boldsymbol{x},\boldsymbol{p})\boldsymbol{N}(\boldsymbol{x})\\ &= \frac{1}{(1+\sigma A)^2}\begin{pmatrix}(\sigma+\gamma)\boldsymbol{A} & -\boldsymbol{A}\\ (\mathbb{I}-\gamma\boldsymbol{A}) & \boldsymbol{A}\end{pmatrix}\cdot\begin{pmatrix}\boldsymbol{C}(\boldsymbol{x})^T\big(\nabla^3 f(\boldsymbol{x})\nabla f(\boldsymbol{x})+(\nabla^2 f(\boldsymbol{x}))^2\big)\boldsymbol{C}(\boldsymbol{x}) & 0\\ 0 & \mathbb{I}\end{pmatrix}\\ &\qquad\cdot\begin{pmatrix}(\sigma+\gamma)\boldsymbol{A} & (\mathbb{I}-\gamma\boldsymbol{A})\\ -\boldsymbol{A} & \boldsymbol{A}\end{pmatrix},\end{aligned} \tag{3.21}$$

where we recall $\boldsymbol{C}(\boldsymbol{x})=\boldsymbol{B}(\boldsymbol{x})\nabla^2 f(\boldsymbol{x})$.

By assumption, there exists $L'>0$, such that

$$\boldsymbol{C}(\boldsymbol{x})^T\big(\nabla^3 f(\boldsymbol{x})\nabla f(\boldsymbol{x})+(\nabla^2 f(\boldsymbol{x}))^2\big)\boldsymbol{C}(\boldsymbol{x}) \preceq L'\mathbb{I}\,,$$

for all $\boldsymbol{x}$. Then, combining Eq. 3.21 and Eq. 3.20, we obtain that

$$\begin{aligned}\|\boldsymbol{N}(\boldsymbol{x})^T\nabla^2\mathcal{I}(\boldsymbol{x},\boldsymbol{p})\boldsymbol{N}(\boldsymbol{x})\|_2 &\le \frac{\big(A(\sigma+2\gamma+2)+1\big)^2}{(1+\sigma A)^2}\cdot\max\{L',1\}\\ &\le \frac{(3+\sigma A+2A)^2}{(1+\sigma A)^2}\cdot\max\{L',1\}\,,\end{aligned} \tag{3.22}$$

where we have used $\gamma A<1$ to derive the last inequality. □

Theorem 3.10 (Restatement of Theorem 1.1)**.** *Suppose Assumption 3.1 and Assumption 3.2 hold. Let* $\sigma = \tau$, $\gamma = \frac{1-\sigma\mu}{\mu}, \varepsilon = 1, A = \frac{\mu+L}{2+(\mu+L)\varepsilon\gamma}$. *And suppose 3.14 holds for some* $\delta > 0$ *and all* $\boldsymbol{x}$. *If* $\tau = \frac{1}{4}\frac{\mu}{\delta+36\max\{L',1\}}$, *then*

$$\mathcal{I}(\boldsymbol{x}^n, \boldsymbol{p}^n) \leq \mathcal{I}(\boldsymbol{x}^0, \boldsymbol{p}^0)\big(1 - \frac{\mu^2/32}{\delta + 36\max\{L',1\}}\big)^n .$$

Proof. By Assumption 3.1 and Assumption 3.2, we have $\mu \leq L \leq L'$. Thus $\mu/L' \leq 1$ and $\sigma = \tau < 1/36$. Moreover,

$$\gamma = \frac{1}{\mu} - \sigma \geq 1 - \frac{1}{36} = \frac{35}{36} .$$

And

$$A = \frac{\mu + L}{2 + (\mu + L)\varepsilon\gamma} < \frac{1}{\gamma} \leq \frac{36}{35} .$$

Then it follows

$$\frac{3 + \sigma A + 2A}{1 + \sigma A} \leq 3 + \sigma A + 2A < 6 .$$

By Corollary 3.9, we have

$$\|\boldsymbol{N}(\boldsymbol{x})^T \nabla^2 \mathcal{I}(\boldsymbol{x}, \boldsymbol{p}) \boldsymbol{N}(\boldsymbol{x})\|_2 \leq 36\max\{L', 1\} .$$

Combining this with Theorem 3.6 and Corollary 3.8, we finish the proof.

□

Remark 3.11. The choice of parameters in Theorem 3.10 may not be optimal. The main purpose of Theorem 3.10 is to show the existence of geometric convergence in Algorithm 1.

4 Numerical experiment

We test our PDD algorithm using several convex and non-convex functions and compare the results with other commonly used optimizers, such as gradient descent, Nesterov's accelerated gradient (NAG), IGAHD (inertial gradient algorithm with Hessian damping) [Attouch et al., 2020], and IGAHD-SC (inertial gradient algorithm with Hessian damping for strongly convex functions) [Attouch et al., 2020].

4.1 Summary of algorithms

For reference, we write down the iterations of gradient descent, NAG, IGAHD-SC, and IGAHD for better comparison.

Gradient descent:

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - \tau_{\mathrm{gd}} \nabla f(\boldsymbol{x}^n) ,$$

where $\tau_{\mathrm{gd}} > 0$ is a stepsize.

NAG:

$$\begin{aligned} \boldsymbol{y}^{n+1} &= \boldsymbol{x}^n - \tau_{\mathrm{nag}} \nabla f(\boldsymbol{x}^n) , \\ \boldsymbol{x}^{n+1} &= \boldsymbol{y}^{n+1} + \beta_{\mathrm{nag}}(\boldsymbol{y}^n - \boldsymbol{y}^{n-1}) , \end{aligned}$$

where $\tau_{\mathrm{nag}} > 0$ is a stepsize, and $\beta_{\mathrm{nag}} > 0$ is a parameter.

IGAHD: Suppose ∇f is L_1-Lipschitz.

$$\begin{aligned} \boldsymbol{y}^n &= \boldsymbol{x}^n + \alpha_n(\boldsymbol{x}^n - \boldsymbol{x}^{n-1}) - \beta^{(1)}\sqrt{\tau_{\mathrm{att}}}(\nabla f(\boldsymbol{x}^n) - \nabla f(\boldsymbol{x}^{n-1})) - \frac{\beta^{(1)}\sqrt{\tau_{\mathrm{att}}}}{n}\nabla f(\boldsymbol{x}^{n-1}) , \\ \boldsymbol{x}^{n+1} &= \boldsymbol{y}^n - \tau_{\mathrm{att}} \nabla f(\boldsymbol{y}^n) . \end{aligned}$$

Here $\alpha_n = 1 - \frac{\alpha}{n}$ for some $\alpha \geq 3$. $\beta^{(1)}$ needs to satisfy

$$0 \leq \beta^{(1)} \leq 2\sqrt{\tau_{\text{att}}}\,.$$

And $\tau_{\text{att}} > 0$ is a stepsize, which needs to satisfy

$$\tau_{\text{att}} \leq \frac{1}{L_1}\,.$$

Remark 4.1. As mentioned earlier, in each iteration of IGAHD, $\nabla f(\cdot)$ is evaluated twice: at $\boldsymbol{x}^n$ and at $\boldsymbol{y}^n$. This differs from one gradient evaluation in gradient descent, NAG, and our method Eq. 1.2. Chen and Luo [2021] proposed a slightly different algorithm from IGAHD that only requires one gradient evaluation in each iteration.

IGHD-SC: Suppose f is m_1-strongly convex and ∇f is L_1-Lipschitz.

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n + \frac{1-\sqrt{m_1\tau_{\text{att}}}}{1+\sqrt{m_1\tau_{\text{att}}}}(\boldsymbol{x}^n - \boldsymbol{x}^{n-1}) - \frac{\beta^{(2)}\sqrt{\tau_{\text{att}}}}{1+\sqrt{m_1\tau_{\text{att}}}}(\nabla f(\boldsymbol{x}^n) - \nabla f(\boldsymbol{x}^{n-1})) - \frac{\tau_{\text{att}}}{1+\sqrt{m_1\tau_{\text{att}}}}\nabla f(\boldsymbol{x}^n)\,.$$

Here $\beta^{(2)}$ and L_1 need to satisfy

$$\beta^{(2)} \leq \frac{1}{\sqrt{m_1}}\,, \quad L_1 \leq \min\left\{\frac{\sqrt{m_1}}{8\beta^{(2)}}, \frac{\frac{\sqrt{m_1}}{2\tau_{\text{att}}} + \frac{m_1}{\sqrt{\tau_{\text{att}}}}}{2\beta^{(2)}m_1 + \frac{1}{\sqrt{\tau_{\text{att}}}} + \frac{\sqrt{m_1}}{2}}\right\}. \tag{4.1}$$

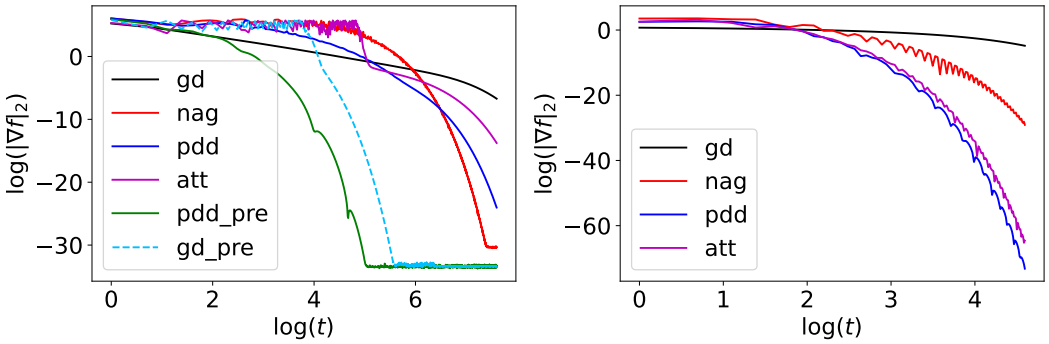
4.2 regularized log-sum-exp

Consider the regularized log-sum-exp function

$$f(\boldsymbol{x}) = \log\left(\sum_{i=1}^{n}\exp(\boldsymbol{q}_i^T\boldsymbol{x})\right) + \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}\,,$$

where $n = 100$, $\boldsymbol{Q} = \boldsymbol{Q}^T \succ 0$ and $\boldsymbol{q}_i^T$ is the ith row of $\boldsymbol{Q}$. $\boldsymbol{Q}$ is chosen to be diagonally dominant, i.e. $Q_{i,i} > \sum_{j\neq i}|Q_{i,j}|$. In this case, we may choose the diagonal preconditioner $\boldsymbol{C}(\boldsymbol{x}) = \big(\text{diag}(\boldsymbol{Q})\big)^{-1}$. We compare the performance of gradient descent, preconditioned gradient descent, PDD with $\boldsymbol{C}(\boldsymbol{x}) = \mathbb{I}$, PDD with diagonal preconditioner, NAG, and IGAHD-SC (inertial gradient algorithm with Hessian damping for strongly convex functions) by Attouch et al. [2020] methods for minimizing f. The stepsize of gradient descent is determined by $\tau_{\text{gd}} = \frac{2}{\lambda_1 * 3 + \lambda_n}$, where λ_1 and λ_n are the maximum and minimum eigenvalues of $\boldsymbol{Q}$, respectively. For a pure quadratic objective function, $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}$, the optimal stepsize of gradient descent is $\frac{2}{\lambda_1+\lambda_n}$. However, since our objective function also contains a log-sum-exp term, we slightly change the stepsize. Otherwise, gradient descent will not converge. Similarly, when deciding the parameters for NAG, we choose $\tau_{\text{nag}} = \frac{4}{30*\lambda_1+\lambda_n}$ and $\beta_{\text{nag}} = \frac{\sqrt{3\kappa'+1}-2}{\sqrt{3\kappa'+1}+2}$, where $\kappa' = 10\lambda_1/\lambda_n$, which is slightly smaller than the optimal parameters of NAG for a purely quadratic function to guarantee convergence. For PDD with $\boldsymbol{C}(\boldsymbol{x}) = \mathbb{I}$, we choose $\tau_{\text{pdd}} = \sigma_{\text{pdd}} = \frac{2}{\lambda_1+\lambda_n}$, $\varepsilon = 1$, $A = 10$, $\omega = 1$. For PDD with diagonal preconditioner $\boldsymbol{C}(\boldsymbol{x}) = \big(\text{diag}(\boldsymbol{Q})\big)^{-1}$, we choose $\tau_{\text{pdd}} = \sigma_{\text{pdd}} = 0.5$, $\varepsilon = 1$, $A = 1$, $\omega = 1$. We use the same $\boldsymbol{C}(\boldsymbol{x}) = \big(\text{diag}(\boldsymbol{Q})\big)^{-1}$ as a preconditioner for gradient descent. The stepsize for preconditioned gradient descent is chosen to be the same as $\tau_{\text{pdd}} = 0.5$. For IGAHD-SC ('att'), we need m_1 as the smallest eigenvalue of $\nabla^2 f(\boldsymbol{x})$. In this example, we may estimate m_1 as the smallest eigenvalue of $\boldsymbol{Q}$. And $\tau_{\text{att}} = 0.0016$ via grid search. $\beta^{(2)}$ in IGAHD-SC is found by solving (see Theorem 11 Eq. (26) of Attouch et al. [2020])

$$\frac{\sqrt{m_1}}{8\beta^{(2)}} = \frac{\frac{\sqrt{m_1}}{2\tau_{\text{att}}} + \frac{\sqrt{m_1}}{\sqrt{\tau_{\text{att}}}}}{2\beta^{(2)}m_1 + \frac{1}{\sqrt{\tau_{\text{att}}}} + \frac{\sqrt{m_1}}{2}}\,,$$



(a) Regularized log-sum-exp

(b) Qudratic minus cosine

Figure 1: Comparison of gradient descent, NAG, PDD, and IGAHD-SC (we use 'att' as a shorthand for this method) on minimizing (a) the regularized log-sum-exp function and (b) the quadratic minus cosine function. The y-axis represents the 2-norm of the gradient of the objective function on a logarithmic scale. The x-axis represents the number of iterations on a logarithmic scale.

which gives

$$\beta^{(2)} = \frac{\sqrt{\tau_{\text{att}}} + \tau_{\text{att}}\sqrt{m_1}/2}{4 + 8\sqrt{m_1}\sqrt{\tau_{\text{att}}} - 2m_1\tau_{\text{att}}}\,. \tag{4.2}$$

The initial condition is $\boldsymbol{x}^0 = \text{np.ones}(n) * 0.1$. The result is presented in Fig. 1a.

4.3 Quadratic minus cosine function

Consider the function

$$f(\boldsymbol{x}) = \|\boldsymbol{x}\|^2 - \cos(\boldsymbol{c}^T\boldsymbol{x})\,,$$

where $\boldsymbol{c}$ is a vector in $\mathbb{R}^{100}$ with $\|\boldsymbol{c}\|^2 = 1.9$. Then a direct calculation shows that $0.1\mathbb{I} \preceq \nabla^2 f(\boldsymbol{x}) \preceq 3.9\mathbb{I}$ for any $\boldsymbol{x}$. This allows us to choose the optimal stepsize for gradient descent and NAG. When minimizing f using gradient descent, we can choose $\tau_{\text{gd}} = \frac{2}{0.1+3.9} = 0.5$. Meanwhile, for NAG, we may choose $\tau_{\text{nag}} = \frac{4}{3*3.9+0.1}$, and $\beta = \frac{\sqrt{3\kappa+1}-2}{\sqrt{3\kappa+1}+2}$, where $\kappa = 3.9/0.1$. For PDD with $\boldsymbol{C}(\boldsymbol{x}) = \mathbb{I}$, we choose $\tau_{\text{pdd}} = \sigma_{\text{pdd}} = 0.5$, $\varepsilon = 1$, $A = 1$, $\omega = 1$. For IGAHD-SC ('att'), we choose $m_1 = 0.1$, $\tau_{\text{att}} = 0.55$ via grid search and $\beta^{(2)}$ is given by Eq. 4.2. The initial condition is $\boldsymbol{x}^0 = \text{np.ones}(n) * 5$. The result is presented in Fig. 1b.

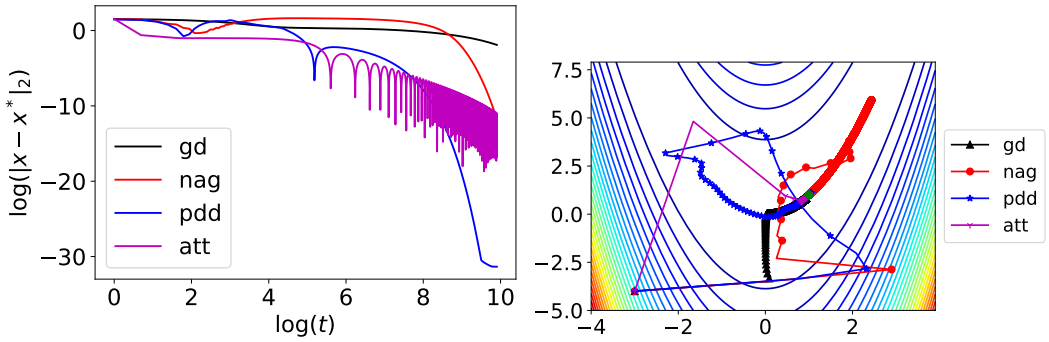
4.4 Rosenbrock function

4.4.1 2-dimension

The 2-dimensional Rosenbrock function is defined as

$$f(x, y) = (a - x)^2 + b(y - x^2)^2\,,$$

where a common choice of parameters is $a = 1$, $b = 100$. This is a non-convex function with a global minimum of $(x^*, y^*) = (a, a^2)$. The global minimum is inside a long, narrow, parabolic-shaped flat valley. To find the valley is trivial. To converge to the global minimum, however, is difficult. We compare the performance of gradient descent, NAG, PDD with $\boldsymbol{C}(\boldsymbol{x}) = \mathbb{I}$ and IGAHD (inertia gradient algorithm with Hessian damping) by Attouch et al. [2020] when minimizing the



(a) Convergence comparison (b) Optimization trajectories

Figure 2: Minimizing the Rosenbrock function with gradient descent, NAG, PDD with $\boldsymbol{C}(\boldsymbol{x}) = \mathbb{I}$ and IGAHD ('att'). The left panel shows the convergence speed of each method. The right panel shows the optimization trajectories of each method.

Rosenbrock function starting from $(-3, -4)$. The stepsize of gradient descent is $\tau_{\text{gd}} = 0.0002$. The stepsize of NAG is $\tau_{\text{nag}} = 0.0002$, $\beta_{\text{nag}} = 0.9$. The parameters of PDD are $\tau_{\text{pdd}} = \sigma_{\text{pdd}} = 0.005$, $\varepsilon = 1$, $\omega = 1$, $A = 5$. The stepsize of the PDD method is larger than τ_{gd} and τ_{nag} because gradient descent and NAG do not allow larger stepsizes for convergence. For IGAHD ('att'), we choose $\tau_{\text{att}} = 0.00045$, $\alpha = 3$, $\beta^{(1)} = \sqrt{\tau_{\text{att}}}/14$. The convergence result and the optimization trajectories are shown in Fig. 2.

4.4.2 N-dimension

The N-dimensional coupled Rosenbrock function is defined as

$$f(\boldsymbol{x}) = \sum_{i=1}^{N-1} \left((a - x_i)^2 + b(x_{i+1} - x_i^2)^2\right),$$

where we choose $a = 1$ and $b = 100$ as in the 2-dimensioal case and we set $N = 100$. The global minimum is at $\boldsymbol{x}^* = (1, 1, \ldots, 1)$. Using the same stepsizes as in the 2-dimensional case, we compare the performance of the three algorithms starting from $\boldsymbol{x}_0 = (0, \ldots, 0)$. The stepsize of gradient descent is $\tau_{\text{gd}} = 0.001$. The stepsize of NAG is $\tau_{\text{nag}} = 0.0008$, $\beta = 0.95$. The parameters of PDD are $\tau_{\text{pdd}} = \sigma_{\text{pdd}} = 0.01$, $\varepsilon = 0.5$, $\omega = 1$, $A = 5$. The stepsize of the PDD method is larger than τ_{gd} and τ_{nag} because gradient descent and NAG do not allow larger stepsizes for convergence. For IGAHD ('att'), we choose $\tau_{\text{att}} = 0.0002$, $\alpha = 3$, $\beta^{(1)} = 2 * \sqrt{\tau_{\text{att}}}$. The result is summarized in Fig. 3

4.5 Ackley function

We consider minimizing the two-dimensional Ackley function given by

$$f(x, y) = -20 \exp\big(-0.2\sqrt{0.5(x^2 + y^2)}\big) - \exp\big[0.5\big(\cos(2\pi x) + \cos(2\pi y)\big)\big] + \mathrm{e} + 20,$$

which has many local minima. The unique global minimum is located at $(x^*, y^*) = (0, 0)$. We compare the performance of gradient descent, NAG, PDD, and IGAHD ('att') for minimizing the two-dimensional Ackley function starting from $(x_0, y_0) = (2.5, 4)$. The stepsize of gradient descent is $\tau_{\text{gd}} = 0.002$. The stepsize of NAG is $\tau_{\text{nag}} = 0.002$, $\beta_{\text{nag}} = 0.9$. The parameters of PDD are $\tau_{\text{pdd}} = \sigma_{\text{pdd}} = 0.002$, $\varepsilon = 1$, $\omega = 1$, $A = 1$. For IGAHD ('att'), we choose $\tau_{\text{att}} = 0.01$, $\alpha = 3$, $\beta^{(1)} = 2 * \sqrt{\tau_{\text{att}}}$. The results are summarized in Fig. 4.

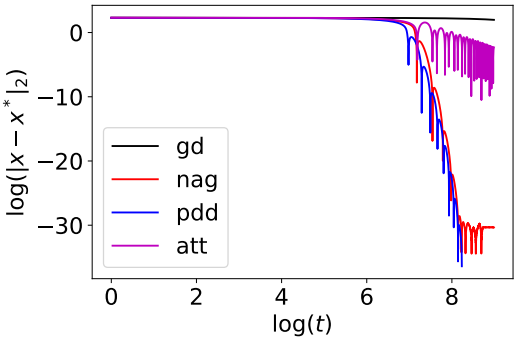
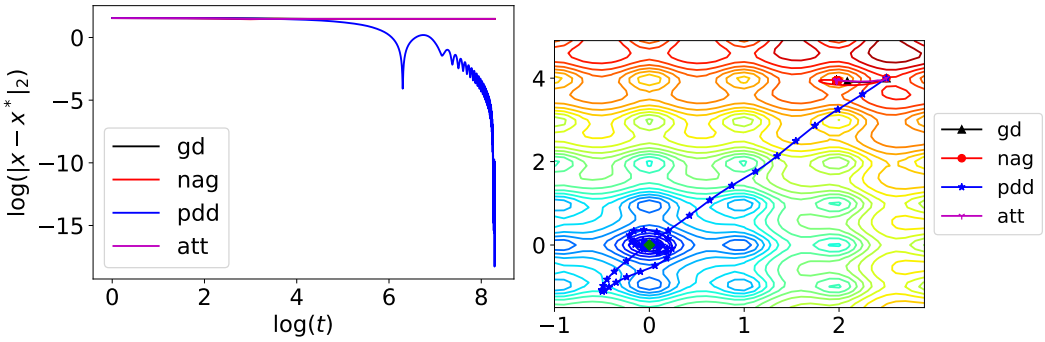


Figure 3: Comparison of gradient descent, NAG, PDD with $\boldsymbol{C}(\boldsymbol{x}) = \mathbb{I}$ and IGAHD ('att') on minimizing the 100-dimensional coupled Rosenbrock function. The y-axis represents the distance between the current iterate and the global minimum on a logarithmic scale. The x-axis represents the number of iterations on a logarithmic scale.



(a) Convergence comparison

(b) Optimization trajectories

Figure 4: Minimizing the Ackley function with gradient descent, NAG, PDD and IGAHD ('att'). The left panel shows the convergence speed of each method. The right panel shows the optimization trajectories of each method.

Algorithm	SGD	NAG	PDD	Adam	Att
train loss	2.223 ± 0.034	0.964 ± 0.244	**0.433 ± 0.270**	0.589 ± 0.282	0.591 ± 0.288
test acc	29.3 ± 8.3 %	71.2 ± 9.4 %	**85.4 ± 10.4** %	79.1 ± 11.3 %	80.8 ± 11.4 %

Table 1: Average training loss and test accuracy of different algorithms for MNIST handwritten digit recognition over 60 random seeds.

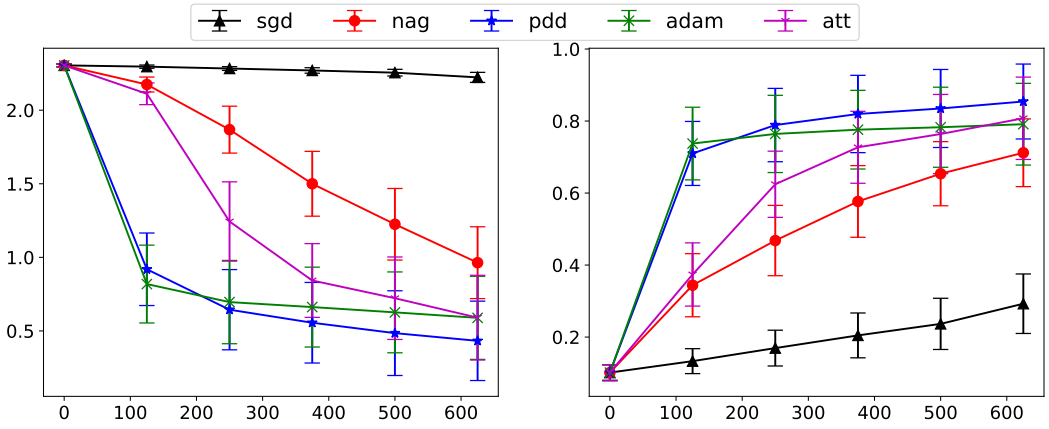


Figure 5: Training a two-layer neural network with the MNIST data set using gradient descent, NAG, PDD, Adam, and IGAHD ('att'). The left panel shows the convergence speed of training loss. The right panel shows the test accuracy of each method. The x-axis represents the number of iterations in terms of mini-batches.

Remark 4.2. We remark that our algorithm has no stochasticity. It will not always converge to the global minimum for non-convex functions in general. For example, it will not converge for the Griewank, Drop-Wave, and Rastrigin functions.

4.6 Neural Networks training

4.6.1 MNIST with Two-layer neural network

We consider the classification problem using the MNIST handwritten digit data set with a two-layer neural network. The neural network has an input layer of size $784 = 28 \times 28$, a hidden layer of size 32 followed by another hidden layer of size 32, and an output layer of size 10. We use ReLU activation function across the layers, and the loss is evaluated using the cross-entropy loss. We use a batch size of 200 for all the algorithms. The stepsize of gradient descent is $\tau_{\text{gd}} = 0.001$. The stepsize of NAG is $\tau_{\text{nag}} = 0.001$, momentum $= 0.9$. The parameters of PDD are $\tau_{\text{pdd}} = 0.001$, $\sigma_{\text{pdd}} = 5$, $\varepsilon = 0.005$, $\omega = 1$, $A = 1$. For IGAHD ('att'), we choose $\tau_{\text{att}} = 0.001$, $\alpha = 3$, $\beta^{(1)} = 0.01$. For Adam, we choose $\tau_{\text{adam}} = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$.

4.6.2 CIFAR10 with CNN

We train a convolutional neural network using the CIFAR10 datasets with SGD, Nesterov, PDD, Adam, and IGAHD ('Att'). The architecture of the network is described as follows. The network consists of two convolutional layers. The first convolutional layer has 32 output channels, and the filter size is 3×3. The second convolutional layer has 64 output channels, and the filter size is

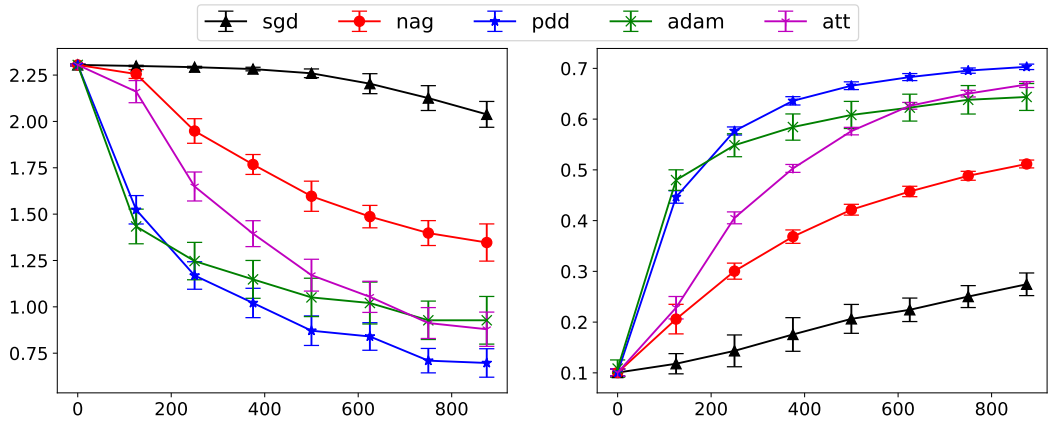


Figure 6: Training loss (left panel) and test accuracy (right panel) of a convolutional neural network on the CIFAR10 data set. The x-axis represents the number of iterations in terms of mini-batches.

Algorithm	SGD	NAG	PDD	Adam	Att
train loss	2.038 ± 0.070	1.347 ± 0.100	**0.697 ± 0.077**	0.927 ± 0.128	0.879 ± 0.092
test acc	27.5 ± 2.2 %	51.2 ± 0.7 %	**70.3 ± 0.5** %	64.4 ± 2.7 %	66.8 ± 0.6 %

Table 2: Average training loss and test accuracy of different algorithms for CIFAR10 data set over 60 random seeds.

4×4. Each convolutional layer is followed by a ReLU activation and then a 2×2 max-pooling layer. Lastly, we have 3 fully connected layers of size $(64 \cdot 4 \cdot 4, 120)$, $(120, 84)$, and $(84, 10)$. The loss is evaluated using the cross-entropy loss. The stepsize of gradient descent is $\tau_{\text{gd}} = 0.01$. The stepsize of NAG is $\tau_{\text{nag}} = 0.005$, momentum $= 0.9$. The parameters of PDD are $\tau_{\text{pdd}} = 0.005$, $\sigma_{\text{pdd}} = 5$, $\varepsilon = 0.005$, $\omega = 1$, $A = 1$. For IGAHD ('att'), we choose $\tau_{\text{att}} = 0.005$, $\alpha = 3$, $\beta^{(1)} = 0.01$. For Adam, we choose $\tau_{\text{adam}} = 0.005$, $\beta_1 = 0.9$, $\beta_2 = 0.999$.

5 Discussion

This paper presents primal-dual hybrid gradient algorithms for solving unconstrained optimization problems. We reformulate the optimality condition of the optimization problem as a saddle-point problem and then compute the proposed saddle-point problem by a preconditioned PDHG method. We present the geometric convergence analysis for the strongly convex objective functions. In numerical experiments, we demonstrate that the proposed method works efficiently in non-convex optimization problems, at least in some examples, such as Rosenbrock and Ackley functions. In particular, it could efficiently train two-layer and convolution neural networks in supervised learning problems.

So far, our convergence study is limited to strongly convex objective functions, not convex ones. Meanwhile, the choice of preconditioners and stepsizes are independent of time. We also have not discussed the optimal choices of parameters or general proximal operators in the updates of algorithms. These generalized choices of functions, parameters, and their convergence properties have been intensively studied in Nesterov accelerated gradient methods and Attouch's Hessian-driven damping methods. In future work, we shall explore the convergence property of PDHG

methods for convex functions with time-dependent parameters. We also investigate the convergence of similar algorithms in scientific computing problems of implicit time updates of partial differential equations [Li et al., 2022, 2023, Liu et al., 2023].

Acknowledgement: X. Zuo and S. Osher's work was partly supported by AFOSR MURI FP 9550-18-1-502 and ONR grants: N00014-20-1-2093 and N00014-20-1-2787. W. Li's work was supported by AFOSR MURI FP 9550-18-1-502, AFOSR YIP award 2023, and NSF RTG: 2038080.

References

H. Attouch, Z. Chbani, and H. Riahi. Fast proximal methods via time scaling of damped inertial dynamics. *SIAM Journal on Optimization*, 29(3):2227–2256, 2019.

H. Attouch, Z. Chbani, J. Fadili, and H. Riahi. First-order optimization algorithms via inertial systems with Hessian driven damping. *Mathematical Programming*, pages 1–43, 2020.

H. Attouch, Z. Chbani, J. Fadili, and H. Riahi. Convergence of iterates for first-order optimization algorithms with inertia and Hessian driven damping. *Optimization*, pages 1–40, 2021.

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

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*, 3(1):1–122, 2011.

A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of mathematical imaging and vision*, 40:120–145, 2011.

L. Chen and H. Luo. First order optimization methods based on Hessian-driven Nesterov accelerated gradient flow. *arXiv preprint arXiv:1912.09276*, 2019.

L. Chen and H. Luo. A unified convergence analysis of first order convex optimization methods via strong Lyapunov functions. *arXiv preprint arXiv:2108.00132*, 2021.

D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1): 17–40, 1976. ISSN 0898-1221. doi: https://doi.org/10.1016/0898-1221(76)90003-1. URL `https://www.sciencedirect.com/science/article/pii/0898122176900031`.

M. Jacobs, F. Léger, W. Li, and S. Osher. Solving large-scale optimization problems with a convergence rate independent of grid size. *SIAM Journal on Numerical Analysis*, 57(3):1100–1123, 2019.

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

W. Li, S. Liu, and S. Osher. Controlling conservation laws II: Compressible Navier–Stokes equations. *Journal of Computational Physics*, 463:111264, 2022.

W. Li, S. Liu, and S. Osher. Controlling conservation laws I: Entropy–entropy flux. *Journal of Computational Physics*, 480:112019, 2023.

S. Liu, S. Osher, W. Li, and C.-W. Shu. A primal-dual approach for solving conservation laws with implicit in time approximations. *Journal of Computational Physics*, 472:111654, 2023.

Y. Liu, Y. Xu, and W. Yin. Acceleration of primal–dual methods by preconditioning and simple subproblem procedures. *Journal of Scientific Computing*, 86(2):21, 2021.

Y. E. Nesterov. A method of solving a convex programming problem with convergence rate $\mathcal{O}(\frac{1}{k^2})$. In *Doklady Akademii Nauk*, volume 269, pages 543–547. Russian Academy of Sciences, 1983.

S. Osher, B. Wang, P. Yin, X. Luo, F. Barekat, M. Pham, and A. Lin. Laplacian smoothing gradient descent. *Research in the Mathematical Sciences*, 9(3):55, 2022.

Y. Ouyang, Y. Chen, G. Lan, and E. Pasiliao Jr. An accelerated linearized alternating direction method of multipliers. *SIAM Journal on Imaging Sciences*, 8(1):644–681, 2015.

J.-H. Park, A. J. Salgado, and S. M. Wise. Preconditioned accelerated gradient descent methods for locally lipschitz smooth objectives with applications to the solution of nonlinear PDEs. *Journal of Scientific Computing*, 89(1):17, 2021.

R. Pascanu and Y. Bengio. Revisiting natural gradient for deep networks. *arXiv preprint arXiv:1301.3584*, 2013.

T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *2011 International Conference on Computer Vision*, pages 1762–1769, 2011. doi: 10.1109/ICCV.2011.6126441.

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

T. Rees. *Preconditioning iterative methods for PDE constrained optimization*. PhD thesis, University of Oxford Oxford, UK, 2010.

J. W. Siegel. Accelerated first-order methods: Differential equations and Lyapunov functions. *arXiv preprint arXiv:1903.05671*, 2019.

W. Su, S. Boyd, and E. J. Candes. A differential equation for modeling Nesterov's accelerated gradient method: theory and insights. *arXiv preprint arXiv:1503.01243*, 2015.

L. N. Trefethen and D. Bau. *Numerical linear algebra*, volume 181. SIAM, 2022.

T. Valkonen. A primal–dual hybrid gradient method for nonlinear operators with applications to MRI. *Inverse Problems*, 30(5):055012, 2014.

A. Wibisono, A. C. Wilson, and M. I. Jordan. A variational perspective on accelerated methods in optimization. *Proceedings of the National Academy of Sciences*, 113(47):E7351–E7358, 2016.

Appendix A Matrix lemma

Lemma A.1. *Let $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{R}^n$ be real symmetric matrices that are simultaneously diagonalizable. Then for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, if*

$$\lambda_{A,i} + \frac{|\lambda_{C,i}|}{2} \leq 0$$
$$\lambda_{B,i} + \frac{|\lambda_{C,i}|}{2} \leq 0$$

for all i, where $\lambda_{A,i}, \lambda_{B,i}, \lambda_{C,i}$ are the ith eigenvalues of $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ respectively in the same basis. Then

$$\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{B}\boldsymbol{y} + \boldsymbol{x}^T \boldsymbol{C}\boldsymbol{y} \leq 0,$$

for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$.

Proof. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. By our assumption, there exists $\boldsymbol{Q}$ unitary such that $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ are simultaneously diagonalizable by $\boldsymbol{Q}$. Set $\tilde{\boldsymbol{x}} = \boldsymbol{Q}\boldsymbol{x}$ and $\tilde{\boldsymbol{y}} = \boldsymbol{Q}\boldsymbol{y}$. Then we can compute

$$\begin{aligned}
\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{B}\boldsymbol{y} + \boldsymbol{x}^T \boldsymbol{C}\boldsymbol{y} &= \sum_{i=1}^n \tilde{x}_i^2 \lambda_{A,i} + \tilde{y}_i^2 \lambda_{B,i} + \tilde{x}_i \lambda_{C,i} \tilde{y}_i \\
&\leq \sum_{i=1}^n \tilde{x}_i^2 \big(\lambda_{A,i} + \frac{|\lambda_{C,i}|}{2}\big) + \tilde{y}_i^2 \big(\lambda_{B,i} + \frac{|\lambda_{C,i}|}{2}\big) \\
&\leq 0\,,
\end{aligned}$$

where the first inequality follows from $\alpha xy \leq (x^2 + y^2)|\alpha|/2$ for any $\alpha, x, y \in \mathbb{R}$. □

Appendix B Proof of Theorem 2.6

B.1 Part (a)

We have the following system of ODE:

$$\begin{pmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{p}} \end{pmatrix} = \begin{pmatrix} -\gamma \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q} & -\boldsymbol{B}\boldsymbol{Q}(\mathbb{I} - \gamma\varepsilon \boldsymbol{A}) \\ \boldsymbol{A}\boldsymbol{Q} & -\varepsilon \boldsymbol{A} \end{pmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{p} \end{pmatrix}. \tag{B.1}$$

Let us compute the eigenvalues of the above system. Let α be an eigenvalue, then α satisfies

$$\begin{aligned}
\det \begin{pmatrix} -\gamma \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q} - \alpha\mathbb{I} & -\boldsymbol{B}\boldsymbol{Q}(\mathbb{I} - \gamma\varepsilon \boldsymbol{A}) \\ \boldsymbol{A}\boldsymbol{Q} & -\varepsilon \boldsymbol{A} - \alpha\mathbb{I} \end{pmatrix} &= 0 \\
\det\big((-\gamma \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q} - \alpha\mathbb{I})(-\varepsilon \boldsymbol{A} - \alpha\mathbb{I}) + \boldsymbol{B}\boldsymbol{Q}(\mathbb{I} - \gamma\varepsilon \boldsymbol{A})\boldsymbol{A}\boldsymbol{Q}\big) &= 0 \\
\det\big(\alpha^2\mathbb{I} + \alpha(\varepsilon \boldsymbol{A} + \gamma \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q}) + \gamma\varepsilon \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q}\boldsymbol{A} + \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q} - \gamma\varepsilon \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{A}\boldsymbol{Q}\big) &= 0 \\
\det\big(\alpha^2\mathbb{I} + \alpha(\varepsilon \boldsymbol{A} + \gamma \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q}) + \boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q}\big) &= 0\,.
\end{aligned}$$

The late equality is because $\boldsymbol{A}$ commutes with $\boldsymbol{Q}$. We assume that $\boldsymbol{A}$ and $\boldsymbol{B}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q}$ are simultaneously diagonalizable. Thus,

$$\begin{aligned}
0 &= \alpha^2 + \alpha(\varepsilon a_i + \gamma\mu_i) + \mu_i\,, \\
\alpha &= \frac{-\varepsilon a_i - \gamma\mu_i \pm \sqrt{(\varepsilon + \gamma\mu_i)^2 - 4\mu_i}}{2}\,.
\end{aligned}$$

If $\gamma > 0$ and $\varepsilon \geq 0$, then the real part of the eigenvalues are negative, and the system will converge. The convergence rate depends on the largest real part of the eigenvalues, which is

$$\max_i \frac{1}{2}\big[-\gamma\mu_i - \varepsilon a_i + \Re\big(\sqrt{(\gamma\mu_i + \varepsilon)^2 - 4\mu_i}\big)\big]\,.$$

B.2 Part (c)

When $\gamma = \varepsilon = 0$, we see that α is purely imaginary. Thus solutions to Eq. B.1 will be oscillatory and will not converge.

B.3 Part (b)

Let us define

$$g(\gamma) = \max_i \Big\{ \frac{\mu_i\big(-\gamma + \Re\big(\sqrt{\gamma^2 - 4/\mu_i}\big)\big)}{2} \Big\}.$$

Essentially, we would like to find $\gamma^* = \arg\min_\gamma g(\gamma)$. We then define

$$\begin{aligned}\gamma(\mu) &:= \arg\min_\gamma \frac{\mu\big(-\gamma + \Re\big(\sqrt{\gamma^2 - 4/\mu}\big)\big)}{2} \\ &= \frac{2}{\sqrt{\mu}}\,.\end{aligned}$$

Observe that if $\gamma \geq 2/\sqrt{\mu_n}$, then $\gamma^2 - 4/\mu_i \geq 0$ for all i. Thus

$$g(\gamma) = \max_i \Big\{ \frac{\mu_i\big(-\gamma + \sqrt{\gamma^2 - 4/\mu_i}\big)}{2} \Big\}.$$

For $\mu \in [\mu_n, \mu_1]$ and $\gamma \geq 2/\sqrt{\mu_n}$, one can check that the function $\mu\big(-\gamma + \sqrt{\gamma^2 - 4/\mu}\big)$ is increasing in μ by computing the partial derivative with respect to μ. Then we get

$$g(\gamma) = \frac{\mu_1\big(-\gamma + \sqrt{\gamma^2 - 4/\mu_1}\big)}{2} \geq g(2/\sqrt{\mu_n}) = \sqrt{\mu_1}(\sqrt{\kappa - 1} - \sqrt{\kappa}) \approx -\sqrt{\mu_n}/2\,,$$

where $\kappa = \mu_1/\mu_n > 1$. The last approximation is valid for $\mu_1/\mu_n \gg 1$. This shows that $\gamma^* \leq 2/\sqrt{\mu_n}$. Similarly, if $\gamma \leq 2/\sqrt{\mu_1}$, then $\gamma^2 - 4/\mu_i \leq 0$ for all i. Thus

$$\begin{aligned} g(\gamma) &= \max_i \Big\{ \frac{-\mu_i \gamma}{2} \Big\} \\ &= \frac{-\mu_n \gamma}{2} \\ &\geq -\frac{\mu_n}{\sqrt{\mu_1}} = g(2/\sqrt{\mu_1})\,. \end{aligned}$$

This shows that $\gamma^* \geq 2/\sqrt{\mu_1}$. Combining with our previous observation, we get $\gamma^* \in [2/\sqrt{\mu_1}, 2/\sqrt{\mu_n}]$. Now let us fix some $\gamma' \in [2/\sqrt{\mu_1}, 2/\sqrt{\mu_n}]$. Let $j = \inf\{i : 1 \leq i \leq n, \gamma'^2 - 4/\mu_i \leq 0\}$. By our assumption on γ', we know that $1 < j < n$. Now for $1 \leq i \leq j-1$, we have

$$\frac{\mu_i\big(-\gamma' + \Re\big(\sqrt{\gamma'^2 - 4/\mu_i}\big)\big)}{2} = \frac{\mu_i\big(-\gamma' + \sqrt{\gamma'^2 - 4/\mu_i}\big)}{2} \leq \frac{\mu_1\big(-\gamma' + \sqrt{\gamma'^2 - 4/\mu_1}\big)}{2}\,.$$

And for $j \leq k \leq n$, we have

$$\frac{\mu_k\big(-\gamma' + \Re\big(\sqrt{\gamma'^2 - 4/\mu_k}\big)\big)}{2} = \frac{-\mu_k\gamma'}{2} \leq \frac{-\mu_n\gamma'}{2}\,.$$

It is thus clear that for $\gamma' \in [2/\sqrt{\mu_1}, 2/\sqrt{\mu_n}]$,

$$g(\gamma') = \max\Big\{ \frac{\mu_1\big(-\gamma' + \sqrt{\gamma'^2 - 4/\mu_1}\big)}{2}, \frac{-\mu_n\gamma'}{2} \Big\}.$$

It is straightforward to calculate that for $\gamma \in [\frac{2}{\sqrt{\mu_1}}, \frac{2\sqrt{\mu_1}}{\sqrt{\mu_n(2\mu_1-\mu_n)}}]$, we have

$$\frac{-\mu_n\gamma}{2} \geq \frac{\mu_1\big(-\gamma+\sqrt{\gamma^2-4/\mu_1}\big)}{2}\,.$$

So

$$g(\gamma) = \frac{-\mu_n\gamma}{2} \geq g(\frac{2\sqrt{\mu_1}}{\sqrt{\mu_n(2\mu_1-\mu_n)}}) = \frac{-\sqrt{\mu_n}}{\sqrt{2-\frac{1}{\kappa}}}\,.$$

And for $\gamma \in [\frac{2\sqrt{\mu_1}}{\sqrt{\mu_n(2\mu_1-\mu_n)}}, 2/\sqrt{\mu_n}]$ we have

$$\frac{-\mu_n\gamma}{2} \leq \frac{\mu_1\big(-\gamma+\sqrt{\gamma^2-4/\mu_1}\big)}{2}\,.$$

This implies

$$g(\gamma) = \frac{\mu_1\big(-\gamma+\sqrt{\gamma^2-4/\mu_1}\big)}{2} \geq g(\frac{2\sqrt{\mu_1}}{\sqrt{\mu_n(2\mu_1-\mu_n)}}) = \frac{-\sqrt{\mu_n}}{\sqrt{2-\frac{1}{\kappa}}}\,.$$

This shows $\gamma^* = \frac{2\sqrt{\mu_1}}{\sqrt{\mu_n(2\mu_1-\mu_n)}}$.

B.4 Part (d)

Define $\Delta_\gamma(\mu,\varepsilon) = (\gamma\mu+\varepsilon)^2 - 4\mu$. Also define $g_\gamma(\mu) = 2\sqrt{\mu} - \gamma\mu$. Then for $\mu \geq 0$, we have $\Delta_\gamma(\mu,\varepsilon) \leq 0$ if and only if $\varepsilon \leq g_\gamma(\mu)$. Note that $g'_\gamma(\mu) = \frac{1}{\sqrt{\mu}} - \gamma \geq 0$ for $\mu \leq \mu_1$ if $\gamma \leq \frac{1}{\sqrt{\mu_1}}$. Then $\Delta_\gamma(\mu,\varepsilon) \leq 0$ for all $\mu \leq \mu_1$ if $\gamma \leq \frac{1}{\sqrt{\mu_1}}$ and $\varepsilon \leq g_\gamma(\mu_n)$. In particular, $\Delta_\gamma(\mu,\varepsilon) \leq 0$ for all $\mu \leq \mu_1$ if $\varepsilon = g_\gamma(\mu')$ for some $\mu' \leq \mu_n$. We have

$$\begin{aligned}
\alpha &= \max_i \frac{1}{2}\big[-\gamma\mu_i - \varepsilon + \Re\big(\sqrt{(\gamma\mu_i+\varepsilon)^2-4\mu_i}\big)\big] \\
&= \max_i \frac{1}{2}\big[-\gamma\mu_i - \varepsilon\big] \\
&= \max_i \frac{1}{2}\big[-\gamma\mu_i - 2\sqrt{\mu'} + \gamma\mu'\big] \\
&= -\sqrt{\mu'} - \frac{\gamma(\mu_n-\mu')}{2}.
\end{aligned}$$

Appendix C Proof of Proposition 2.4

We directly compute

$$\begin{aligned}
\ddot{\boldsymbol{x}} &= -\boldsymbol{C}\big((\mathbb{I}-\gamma\varepsilon\boldsymbol{A})\dot{\boldsymbol{p}} + \gamma\boldsymbol{A}\nabla^2 f(\boldsymbol{x})\dot{\boldsymbol{x}}\big) - \dot{\boldsymbol{C}}\big((\mathbb{I}-\gamma\varepsilon\boldsymbol{A})\boldsymbol{p} + \gamma\boldsymbol{A}\nabla f(\boldsymbol{x})\big) \\
&= -\boldsymbol{C}\big((\mathbb{I}-\gamma\varepsilon\boldsymbol{A})(\boldsymbol{A}\nabla f(\boldsymbol{x}) - \varepsilon\boldsymbol{A}\boldsymbol{p}) + \gamma\boldsymbol{A}\nabla^2 f(\boldsymbol{x})\dot{\boldsymbol{x}}\big) \\
&\quad - \dot{\boldsymbol{C}}\big((\mathbb{I}-\gamma\varepsilon\boldsymbol{A})\boldsymbol{p} + \gamma\boldsymbol{A}\nabla f(\boldsymbol{x})\big) \\
&= -\boldsymbol{C}\big[(\mathbb{I}-\gamma\varepsilon\boldsymbol{A})\boldsymbol{A}\nabla f(\boldsymbol{x}) + \varepsilon\boldsymbol{A}(\boldsymbol{C}^{-1}\dot{\boldsymbol{x}} + \gamma\boldsymbol{A}\nabla f(\boldsymbol{x})) + \gamma\boldsymbol{A}\nabla^2 f(\boldsymbol{x})\dot{\boldsymbol{x}}\big] + \dot{\boldsymbol{C}}\boldsymbol{C}^{-1}\dot{\boldsymbol{x}} \\
&= -\boldsymbol{C}\big[\boldsymbol{A}\nabla f(\boldsymbol{x}) + \varepsilon\boldsymbol{A}\boldsymbol{C}^{-1}\dot{\boldsymbol{x}} + \gamma\boldsymbol{A}\nabla^2 f(\boldsymbol{x})\dot{\boldsymbol{x}}\big] + \dot{\boldsymbol{C}}\boldsymbol{C}^{-1}\dot{\boldsymbol{x}}\,.
\end{aligned}$$