Fast Unconstrained Optimization via Hessian Averaging and Adaptive Gradient Sampling Methods

Thomas O'Leary-Roseberry and Raghu Bollapragada

August 27, 2024

Abstract

We consider minimizing finite-sum and expectation objective functions via Hessian-averaging based subsampled Newton methods. These methods allow for gradient inexactness and have fixed per-iteration Hessian approximation costs. The recent work (Na et al. 2023) demonstrated that Hessian averaging can be utilized to achieve fast $\mathcal{O}\left(\sqrt{\frac{\log k}{k}}\right)$ local superlinear convergence for strongly convex functions in high probability, while maintaining fixed per-iteration Hessian costs. These methods, however, require gradient exactness and strong convexity, which poses challenges for their practical implementation. To address this concern we consider Hessian-averaged methods that allow gradient inexactness via norm condition based adaptive-sampling strategies. For the finite-sum problem we utilize deterministic sampling techniques which lead to global linear and sublinear convergence rates for strongly convex and nonconvex functions respectively. In this setting we are able to derive an improved deterministic local superlinear convergence rate of $\mathcal{O}\left(\frac{1}{k}\right)$. For the expectation problem we utilize stochastic sampling techniques, and derive global linear and sublinear rates for strongly convex and nonconvex functions, as well as a $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ local superlinear convergence rate, all in expectation. We present novel analysis techniques that differ from the previous probabilistic results. Additionally, we propose scalable and efficient variations of these methods via diagonal approximations and derive the novel diagonally-averaged Newton (Dan) method for large-scale problems. Our numerical results demonstrate that the Hessian averaging not only helps with convergence, but can lead to state-of-the-art performance on difficult problems such as CIFAR100 classification with ResNets.

1 Introduction

We consider finite-sum optimization problems of the form

$$\min_{w\in\mathbb{R}^d} f(w) := \frac{1}{N}\sum_{i=1}^{N} F_i(w), \tag{1.1}$$

where the objective function $f : \mathbb{R}^d \to \mathbb{R}$ and the component functions $F_i : \mathbb{R}^d \to \mathbb{R}$ (for $i \in \{1, 2, \ldots, N\}$) are twice continuously differentiable functions. Problems of this form are ubiquitous in modern computing applications including machine learning [19, 37] and scientific computing [48, 58]. In addition, finite-sum problems commonly arise in stochastic optimization settings when sample average approximations (SAA) of the optimization problems of the form

$$\min_{w\in\mathbb{R}^d} f(w) := \mathbb{E}_{\zeta}[F(w,\zeta)], \tag{1.2}$$

are considered. Here ζ is a random variable with associated probability space $(\Omega, \mathcal{F}, P)$, where Ω is a sample space, $\mathcal{F}$ is an event space, and P is a probability distribution. The function $F : \mathbb{R}^d \times \Omega \to \mathbb{R}$ is a twice continuously differentiable function, and $\mathbb{E}_\zeta[\cdot]$ is the expectation taken with respect to the distribution of ζ. For any given set of random realizations $\{\zeta_1, \zeta_2, \cdots, \zeta_N\}$ generated from the distribution P, an SAA problem of the form (1.1) is constructed by defining $F_i(\cdot) := F(\cdot, \zeta_i)$. In supervised machine learning, the random variable $\zeta := (x, y)$ represents input-output data pairs and the function f is a composition of a prediction function and a smooth loss function [19, 37]. The resulting finite-sum problem is referred to as *empirical risk* and the expectation problem is referred to as the *expected risk* minimization problem [82]. In this paper, we consider algorithms for solving problem (1.1) and also suitably adapt them to problem (1.2).

Several classes of methods have been proposed to solve (1.1) and (1.2) (c.f. [19, 65]). In this paper, we focus on subsampled Newton methods that employ gradient and Hessian approximations of the objective function in standard Newton-type methods. In these methods, at any given iteration $k \in \mathbb{N}$, the search direction p_k is computed as the solution of the linear system of equations

$$\nabla^2 F_{S_k}(w_k) p_k = -\nabla F_{X_k}(w_k),$$

where

$$\nabla F_{X_k}(w_k) = \frac{1}{|X_k|} \sum_{i \in X_k} \nabla F_i(w_k), \quad \nabla^2 F_{S_k}(w_k) = \frac{1}{|S_k|} \sum_{i \in S_k} \nabla^2 F_i(w_k). \tag{1.3}$$

Here the sets X_k, S_k are subsets of the index set $\{1, 2, \cdots\}$ and ζ_i's are drawn independently from each other at random from the distribution P. In the case of the finite-sum problem (1.1), the sets X_k, S_k are subsets of the index set $\{1, 2, \cdots, N\}$. The choice of the subsets X_k, S_k, referred to as sample sets, result in different algorithms. Several recent works have analyzed the theoretical and empirical properties for different choices of sampling sets X_k and S_k [1, 13, 21, 22, 25, 34, 35, 73, 75]. Although the computationally efficient choice is to choose Hessian sample sizes ($|S_k|$) to be a fixed small constant, the Hessian approximations in these works require large $|S_k|$ to achieve fast local linear convergence, and increasingly large $|S_k|$ to achieve local superlinear convergence [13, 75]. However, such large Hessian sample sizes lead to high per-iteration computational cost making them unsuitable for large-scale optimization settings.

Recently, Na et. al. [64], have proposed and analyzed stochastic Hessian-averaged Newton methods that overcome the high per-iteration Hessian sampling costs of previous methods by instead employing a weighted average of the subsampled Hessians computed at past iterations. This approach increases the accuracy of the Hessian approximations by reducing its variance, albeit at the cost of introducing bias due to utilization of past iterates. Na et. al. established fast superlinear convergence results in high probability with exact gradients and fixed small Hessian sample sizes ($|S_k| = |S_0|$ for all k). However, the requirements therein of exact gradient computation at each iteration is not practical in large-scale optimization problems. In this work, we consider the gradients to be inexact and also propose deterministic Hessian-averaging methods where the Hessian samples (with fixed sample size) are chosen in a cyclic order without replacement that lead to faster deterministic superlinear rate of convergence for the finite-sum problem (1.1).

While inexact gradients can be utilized in Newton-type methods, such methods lead to slower rate of convergence which result in a significant increase in the overall number of linear system solves in these methods. Adaptive gradient sampling methods overcome this limitation where the sample sizes employed in the gradient estimation are gradually increased to increase the accuracy in gradient estimation and retain the fast convergence properties of their deterministic counterparts [8, 12, 16, 22, 26, 70, 71]. Although these methods employ increasingly accurate gradient approximations as the iterations increase ($|X_k|$ increases), they achieve similar overall gradient computational complexity as in the stochastic gradient methods (see [22, Table 4.1], Table 2). These methods typically employ tests to control sample sizes that automatically adapt to problem settings [12, 16, 22, 25]. In this work, we extend adaptive

gradient sampling tests to the Hessian-averaging setting, where the gradient sample sizes are chosen either deterministically or randomly at each iteration. Our goal is a framework for fully inexact (stochastic) Hessian-averaged subsampled Newton methods that limit per-iteration Hessian computational costs, maintain rigorous convergence theory (i.e., global convergence with fast (superlinear) local convergence), and at the same time lead to practical algorithms.

While Hessian-based methods have sound theoretical properties, they may not be viable for modern large-scale optimization problems such as those arising in machine learning due to the storage cost of $\mathcal{O}(d^2)$ and ostensible matrix inversion cost of $\mathcal{O}(d^3)$ associated with Hessian matrix. To overcome this limitation, we propose practical diagonally approximated Newton (Dan) algorithms based on the Hessian-averaged Newton methods, similar to the one employed in Adahessian [90], requiring only $\mathcal{O}(1)$ per-iteration Hessian-vector products and $\mathcal{O}(d)$ storage, making it appropriate for modern memory-constrained settings. We consider variations of this algorithm based on different weightings, and sampling schemes. Additionally, we illustrate the performance benefits of these methods on numerical examples ranging from stochastic quadratic minimization to large-scale deep learning classification, such as CIFAR100 with ResNets.

1.1 Related Work

We provide a concise summary of second-order methods that employ Hessian approximations in Newton-type methods and adaptive gradient sampling methods for solving (1.1) and (1.2). We note that this is not an extensive review of methods but rather closely related works to the methods considered in this work.

Second-order methods. Several second-order methods that employ Hessian approximations have been developed in the literature. Subsampled Newton methods are one of the main class in these methods [1, 13, 21, 22, 23, 29, 33, 34, 52, 60, 73, 83, 88, 89]. Roosta et al. have analyzed the theoretical properties of these methods and provided convergence results in probability for the finite-sum problems [75]. Bollapragada et al. have established convergence results in expectation for both finite-sum and expectation problems and employed conjugate gradient method for solving the Newton system of equations inexactly [13]. They assumed that the individual component functions to be strongly convex to establish the results in expectation. Both these works have established that Hessian samples have to be increased to achieve superlinear rate of convergence. The empirical performance of subsampled Newton methods has been established on different problems [6, 28, 84, 88]. Erdogdu and Montanari et al. have developed and analyzed subsampled Newton methods with truncated eigenvalue decomposition [34]. Agarwal et al. have analyzed Newton methods where the Newton system of equations are solved inexactly using a stochastic gradient method at each iteration[1]. Subsampled Newton methods have also been adapted to nonconvex settings using trust region and cubic regularization methods [86, 87].

Newton-sketch methods are alternate methods for subsampled Newton methods applied to finite-sum problems [6, 38, 39, 51, 73]. These methods require access to the square root of the Hessian, which is possible when generalized linear models are considered in machine learning. Typically, sketching strategies such as randomized Hadamard transformations provide better Hessian approximations compared to subsampling strategies; however they are typically more computationally expensive due to the high per-iteration cost associated with constructing the linear system of equations [6]. Dereziński et al. proposed the Newton-Less method that employs sparse sketch matrices to reduce the computational cost of forming the approximate Hessians [30]. Sketching techniques have also been adapted to the distributed optimization settings [3].

Subsampled Newton and Newton sketch methods require increasingly accurate Hessian approximations to achieve superlinear rate of convergence. Na et al. have proposed Hessian-averaging methods that overcome this limitation [64]. Jiang et al. have improved the global rate of convergence of [64]

while maintaining similar superlinear rate of convergence [46]. In these works, true gradients are employed in the step computations, thus limiting their deployment in practice. In this work, we consider inexact gradients and also consider deterministic Hessian-averaging methods in addition to stochastic Hessian-averaging methods that could further improve the superlinear rate of convergence.

Adaptive gradient sampling methods. Stochastic gradient methods are well-known and widely used method in machine learning. This method however suffers from slow sublinear convergence for strongly convex function due to variance in the stochastic gradient estimation. Adaptive gradient sampling methods overcome this limitation by gradually increasing the accuracy in the gradient estimation by controlling the gradient samples $|X_k|$ to ensure similar convergence guarantees as their deterministic counterparts [14, 22, 25, 26, 35, 70, 71]. Several adaptive rules have been developed to choose the gradient accuracy at each iteration within the algorithm and "norm test" is a popular condition proposed for the unconstrained settings [12, 22, 25, 26]. These adaptive methods achieve both optimal theoretical convergence results and first-order complexity results to achieve an ϵ-accurate solution for the expectation problem. These methods have also been adapted to other problem settings, including derivative-free optimization [14, 15, 17, 79] and stochastic constrained optimization [5, 8, 14, 85]. In this work, we will adapt these methods to the Hessian-averaging based subsampled Newton methods.

Other related methods. There are several other classes of methods for solving (1.1) and (1.2). Variants of first-order methods including diagonal scaling and momentum attain good empirical performance on challenging machine learning tasks [32, 44, 47, 56]. Stochastic quasi-Newton methods that construct quadratic models of the objective function using only stochastic gradient information are a popular class of methods [9, 10, 16, 63, 78, 92]. The limited memory variants of these methods are competitive to first-order methods on several machine learning classification problems [16]. Additionally, Kronecker-factored approximate curvature (KFAC) methods have been demonstrated as powerful algorithms in stochastic optimization [2, 61].

1.2 Contributions

The main contributions of our work are as follows.

1. We develop an adaptive Hessian-averaging algorithmic framework where we incorporate deterministic and stochastic adaptive generalizations of the "norm condition" to choose the gradient accuracy at each iteration for solving (1.1) and (1.2) respectively. We choose Hessian samples either in a deterministic cyclic fashion without replacement for the finite-sum problem (1.1) or randomly from the distribution P for the expectation problem (1.2). Furthermore, we modify the Hessian-averaging scheme whenever it is not a positive-definite matrix to ensure that the Newton steps are employed at each iteration instead of skipping them (see [64, Algorithm 1]) which ensures a convergence rate for every iteration from the start of the iteration as opposed to having a warm-up phase [46]. Furthermore, such modifications automatically become inactive after the iterates enter locally strongly convex regime (see Lemma 3.7).

2. For the finite-sum problem (1.1), we establish *deterministic* global linear (Theorem 3.2) and sublinear (Theorem 3.3) convergence results by employing appropriately chosen gradient accuracy conditions for strongly convex and nonconvex functions respectively. When the iterates enter a locally strongly convex regime, we further establish *deterministic* local superlinear convergence for the case where the Hessian samples are chosen in a cyclic manner without replacement as opposed to randomly choosing samples at each iteration (Theorem 3.10). This choice of Hessian samples leads to an improved superlinear rate of $\mathcal{O}\left(\frac{1}{k}\right)$ as opposed to existing results in the literature

(see Table 1). Moreover this choice produces stronger deterministic results compared to results in probability or expectation.

3. We establish theoretical convergence results for the stochastic settings (1.2) where the inaccurate gradient approximations are chosen such that the stochastic gradient accuracy conditions are satisfied and the Hessian samples are chosen randomly from the distribution P at each iteration. We established similar global linear (Theorem 4.2) and sublinear (Theorem 4.3) convergence results for strongly convex and nonconvex functions, respectively, as in the case of finite-sum problem. Furthermore, using an additional assumption related to the boundedness of the moment of iterates and local strong convexity of subsampled Hessians, we establish local superlinear convergence where the rate is $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ that matches existing results in [46, 64], albeit in expectation as opposed to in probability (see Table 1). We note that in order to prove the $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ expectation result and the $\mathcal{O}\left(\frac{1}{k}\right)$ deterministic result, we utilize different proof techniques than those employed in [46, 64]. While the probabilistic results therein rely on Freedman's inequality for matrix martingales [81], our approach decomposes the Hessian error and places the statistical sampling error at the optimum, allowing us to concentrate the sampling error via direct analysis. We overview the contributions of our convergence results relative to existing methods in Table 1.

4. We establish total computational complexity for Hessian (Corollary 4.9) and gradient (Corollary 4.10) computations necessary to achieve an ϵ-accurate solution. To allow for appropriate comparisons with existing results in the literature, we only considered globally strongly convex functions. Although, the gradient samples are increasing at each iteration, the overall complexity in terms of total gradient samples match with the well-known stochastic gradient method for the expectation problem in terms of the dependence on ϵ. Furthermore, the dependence on the condition number of the problem is improved due to the Hessian-averaging techniques.These results are summarized in Table 2.

5. To tackle large-scale problems, we motivate the use of Hessian-matrix products, which can be efficiently implemented via vectorization on GPUs, leading to algorithms with $\mathcal{O}(1)$ Hessian-vector products per iteration and $\mathcal{O}(d)$ memory requirements. In particular we utilize a diagonal approximation, which can be efficiently computed via randomized estimation [31, 62], leading to the novel diagonally-averaged Newton (Dan) method and its variants.

6. We demonstrate the numerical benefits of Hessian averaging on a range of numerical experiments ranging from stochastic quadratics, logistic regression, image classification (CIFAR10 and CIFAR100) as well as neural operator training. We demonstrate that Hessian averaging overcomes the inherent instability of fully-subsampled Newton methods. In order to target large-scale optimization problems, we propose efficient implementation of Dan using randomized Hutchinson diagonal estimation. In our experiments, we demonstrate that Dan was competitive with Adam and Adahessian (often achieving superior performance); notably Dan does not employ gradient momentum.

result	objective	gradient	Hessian	global	local (superlinear)	result type
[75]	strong convex finite-sum	subsampled	subsampled	linear	asymptotic	$\mathbb{P}$
[13]	strong convex expectation	subsampled	subsampled	linear	asymptotic	$\mathbb{E}$
[46, 64]	strong convex finite-sum	exact	path-averaged	linear	$\mathcal{O}\left(\sqrt{\frac{\log k}{k}}\right)$	$\mathbb{P}$
Theorem 3.10	nonconvex	adaptive	path-averaged	sublinear	$\mathcal{O}\left(\frac{1}{k}\right)$	deterministic
	strong convex finite-sum			linear		
Theorem 4.8	nonconvex	adaptive	path-averaged	sublinear	$\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$	$\mathbb{E}$
	strong convex expectation			linear		

Table 1: Comparison of convergence results. Here $\mathbb{P}$ denotes that the result is probabilistic, while $\mathbb{E}$ denotes that the result is proven in expectation.

1.3 Organization

The paper is organized into six sections. In the remainder of this section, we define notation that is employed throughout the paper. In Section 2, we first describe the Hessian averaging methods and then discuss adaptive gradient-accuracy conditions and the corresponding sample size requirements to satisfy these conditions. In Section 3, we establish theoretical results for the finite-sum minimization problem (1.1). We establish global convergence, superlinear local convergence, followed by global to local transition iteration complexity results. In Section 4, we repeat this analysis for the expectation based sampling methods whose target is the expectation minimization problem (1.2). We also provide iteration and gradient evaluation complexity results. In Section 5, we discuss practical algorithms that are designed to handle large-scale problems. In Section 6, we illustrate the performance of the proposed algorithms on various problems. Finally, concluding remarks are presented in Section 7.

1.4 Notation

We use the following notation throughout the paper. We work with the natural numbers $\mathbb{N} = \{1, 2, \cdots\}$, positive integers $\mathbb{Z}^+ = \mathbb{N} \cup \{0\}$, and the reals $\mathbb{R} = (-\infty, +\infty)$. We consider real-valued ($\mathbb{R}$) spaces $\mathbb{R}^d, \mathbb{R}^{d\times r}$ for vectors and matrices, respectively, for dimensions $d, r \in \mathbb{N}$. We use the Euclidean ℓ^2 norm, $\|\cdot\| = \|\cdot\|_2$ for both vectors and matrices, unless otherwise specified. Assuming $A \in \mathbb{R}^{d\times d}$ is symmetric positive definite, the A weighted norm is defined as $\|w\|_A = \sqrt{w^T A w}$. Given a symmetric but potentially indefinite matrix A, we denote by $\lambda_{\min}(A), \lambda_{\max}(A)$ the smallest and largest eigenvalues of A, respectively and $|A|$ denotes the matrix obtained by replacing the negative eigenvalues with their magnitudes. We denote by $\mathbb{E}_k[\cdot]$ the conditional expectation conditioned on the fact that the algorithm reach iterate w_k. When using more specific conditional expectations, definitions will be specified in the text. Given a sequence $\{e_k \in \mathbb{R}\}_{k=1}^{\infty}$ with a limit e^*, we characterize its rate of convergence as follows. We say that e_k

has Q-convergence with order q to $e^* \in \mathbb{R}$ if there exists a constant C such that

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k^q} = C.$$

For all cases with $q > 1$, and the case that $q = 1$ and $C = 0$, we say that e_k convergences Q-superlinearly to e^*. We say that e_k has R-convergence with order q to e^* if there exists a sequence r_k, such that $|e_k - e^*| < r_k$ for all k and r_k has Q-convergence with order q to 0. R-superlinear convergence is attained when r_k has Q-superlinear convergence to 0. We note by $\tilde{\mathcal{O}}(\cdot)$ the limiting behavior of a function, disregarding logarithmic factors.

2 Hessian Averaging with Adaptive Gradient Sampling

The generic iterate update form of Hessian-averaged Newton method for solving problems of the form (1.1) or (1.2) is given as,

$$w_{k+1} = w_k - \alpha_k p_k, \quad p_k = \widetilde{H}_k^{-1} g_k, \tag{2.1}$$

where $\alpha_k > 0$ is the step size parameter, and $g_k \in \mathbb{R}^d, \widetilde{H}_k \in \mathbb{R}^{d\times d}$, are the gradient and Hessian approximations of $\nabla f(w_k)$ and $\nabla^2 f(w_k)$ respectively. We now discuss the algorithmic components of Hessian averaging to compute $\widetilde{H}_k$ and adaptive gradient sampling to compute g_k for each w_k, with $k \in \mathbb{Z}^+$.

Hessian Averaging. In typical subsampled Newton methods, Hessians are approximated via subsampling where the sample sets are chosen either in a deterministic cyclic fashion or randomly drawn from P as given in (1.3). Though these estimates are unbiased, they typically have large variance and require either large or increasing sample sizes $|S_k|$ to achieve fast local linear or superlinear rates of convergence respectively [13, 75]. Additionally, subsampled Newton iterates that are based on aggressively subsampled Hessian approximations may become unstable, due to the injection of statistical sampling errors in the iterate update. Sample size selection procedures that overcome these issues are not computationally viable in large-scale settings. Hessian-averaging approaches overcome this computational hurdle and reduce the variance in the estimation by employing a path-averaged Hessian with coefficients $\gamma_i \in [0, 1]$, $\sum_{i=0}^k \gamma_i = 1$, defined as follows:

$$\text{Path-averaged Hessian:} \qquad \widehat{H}_k = \sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w_i). \tag{2.2}$$

Here S_i are the independent sample sets drawn either deterministically without replacement in a cyclic fashion or at random at each iteration i. The Hessian approximation error corresponding to this path-averaged estimate can be decomposed into two terms as follows:

$$\widehat{H}_k - \nabla^2 f(w_k) = \underbrace{\sum_{i=0}^{k} \gamma_i \left(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w_k)\right)}_{\text{Hessian memory error}} + \underbrace{\sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w_k) - \nabla^2 f(w_k)}_{\text{sampling error}}. \tag{2.3}$$

Choosing $\gamma_i = 0$ for all $i = 0, \cdots, k-1$ and $\gamma_k = 1$ correspond to the well-known subsampled Newton methods [13, 75] and results only in the *sampling error* which could be significantly large when sample size $|S_k|$ is small. On the other hand, choosing $\gamma_i = \frac{1}{k+1}$ for all $i = 0, \cdots, k$, reduces the *sampling error* but introduces *Hessian memory error* due to utilization of past information. However, as the

iterates converge, both these errors decrease leading to accurate Hessian approximations which is the main motivation for this approach.

In nonconvex settings, the path-averaged subsampled Hessians $\widehat{H}_k$ (2.2) may not be positive-definite and so the computation of search direction $\widehat{H}_k^{-1} g_k$ may not be well-defined. To overcome this limitation, we modify the path-averaged subsampled Hessians to ensure it is spectrally lower bounded below. That is, for any given $\tilde{\mu} > 0$, we define the Hessian approximation as

$$\widetilde{H}_k = \begin{cases} |\widehat{H}_k| & \text{if } \lambda_{\min}(|\widehat{H}_k|) \geqslant \tilde{\mu} \\ |\widehat{H}_k| + \left(\tilde{\mu} - \lambda_{\min}(|\widehat{H}_k|)\right) I & \text{otherwise,} \end{cases} \tag{2.4}$$

where $\lambda_{\min}(A)$ is the smallest eigenvalue of symmetric matrix $A \in \mathbb{R}^{d\times d}$, and $|A|$ denotes the matrix obtained by replacing the negative eigenvalues with their magnitudes. That is, for any symmetric matrix $A = U\Lambda U^T$, $U \in \mathbb{R}^{d\times d}$ is the orthogonal matrix, and $\Lambda \in \mathbb{R}^{d\times d}$ is the diagonal matrix with eigenvalues, $|A| = U|\Lambda|U^T$. This modification ensures that

$$\widetilde{H}_k \succeq \tilde{\mu} I. \tag{2.5}$$

and in the case where $\widehat{H}_k \succeq \tilde{\mu} I$, there is no modification to the path-averaged Hessian ($\widetilde{H}_k = \widehat{H}_k$).

Adaptive Gradient Sampling. The performance of the algorithms based on the iterate update form given in (2.1) also depends on the accuracy of the gradient approximations. In [64], the authors consider exact gradients, which is not practical in large-scale finite-sum problems (1.1) or expectation problems (1.2). Subsampled gradients overcome this limitation, however the sampling errors that they introduce lead to slow convergence. Furthermore, due to the additional costs due to the Hessian evaluations, it is imperative to reduce the number of overall iterations to achieve computational efficiency. Adaptive sampling approaches gradually increase the accuracy in the gradient estimation via increasing sample sizes used in the gradient approximation to achieve fast convergence. These methods ensure that the error in the gradient approximation at each iteration is relatively small compared to the gradient itself. In this work, we combine these approaches with Hessian averaging approaches leading to generalized versions of the norm condition [12, 22, 25] adapted to Newton-type methods. Specifically, we consider the following deterministic and stochastic conditions on the accuracy of the gradient approximations for the finite-sum (1.1) and expectation (1.2) problems.

Condition 2.1. *(Gradient sampling conditions). At each iteration $k \in \mathbb{Z}^+$ and a given symmetric positive-definite matrix $A_k \in \mathbb{R}^{d\times d}$, we utilize the following conditions.*

1. *Deterministic norm condition: For any given $\theta_k \in [0, 1]$ and $\iota_k > 0$, the error in the gradient approximation satisfies the following deterministic norm condition.*

$$\|g_k - \nabla f(w_k)\|_{A_k}^2 \leqslant \theta_k^2 \|\nabla f(w_k)\|_{A_k}^2 + \iota_k. \tag{2.6a}$$

2. *Stochastic norm condition: For any given $\theta_k > 0$ and $\iota_k > 0$, the expected error in the gradient approximation satisfies the following generalized norm condition.*

$$\mathbb{E}[\|g_k - \nabla f(w_k)\|_{A_k}^2 | w_k, A_k] \leqslant \theta_k^2 \|\nabla f(w_k)\|_{A_k}^2 + \iota_k, \tag{2.6b}$$

where $\mathbb{E}[\cdot|w_k, A_k]$ denote the conditional expectation conditioned on A_k and that the algorithm reach iterate w_k.

Remark 2.1. We note that these gradient sampling conditions reduce to the well-known "norm condition" for the choice of $A_k = I$ and $\iota_k = 0$. In this paper, we consider this choice along with $A_k = \widetilde{H}_k^{-1}$ that leads to better theoretical convergence results (see Theorems 3.2 and 4.3). In addition, the sequence ι_k further relaxes the norm condition and by suitably driving this sequence to zero, we establish the convergence and rate of convergence results.

These gradient sampling conditions are satisfied by choosing the gradient sample sizes $|X_k|$ appropriately. Specifically, under the following assumption, we can establish bounds on $|X_k|$ that satisfy these conditions.

Assumption 2.1. *(Gradient approximations). For all $w \in \mathbb{R}^d$, the individual component gradients are bounded relative to the gradient of the objective function $f(w)$. That is,*

1. *For the finite-sum problem: There exists constants $\beta_{1,g}, \beta_{2,g} \geqslant 0$ such that*

$$\|\nabla F_i(w)\|^2 \leqslant \beta_{1,g}\|\nabla f(w)\|^2 + \beta_{2,g} \quad \forall i \in \{1, 2, \cdots, N\}. \tag{2.7a}$$

2. *For the expectation problem: There exists constants $\sigma_{1,g}, \sigma_{2,g} \geqslant 0$ such that*

$$\mathbb{E}_\zeta[\|\nabla F(w, \zeta) - \nabla f(w)\|^2 | w] \leqslant \sigma_{1,g}^2\|\nabla f(w)\|^2 + \sigma_{2,g}^2. \tag{2.7b}$$

Remark 2.2. We note that (2.7a) is a weaker assumption compared to the assumption where the individual gradient components are absolutely bounded and (2.7b) is a standard assumption in stochastic optimization literature [19].

The following lemma establishes the bounds on gradient sample sizes $|X_k|$ at each iteration k.

Lemma 2.1. *Suppose Assumption 2.1 holds. For any $k \in \mathbb{Z}^+$ and $\lambda_{\min}(A_k), \lambda_{\max}(A_k) \in (0, \infty)$ denote the smallest and largest eigenvalues of A_k respectively, we have that*

1. *If* (2.7a) *holds and*

$$|X_k| \geqslant N\left(1 - \sqrt{\frac{\theta_k^2\|\nabla f(w_k)\|_{A_k}^2 + \iota_k}{4\lambda_{\max}(A_k)(\beta_{1,g}\|\nabla f(w_k)\|^2 + \beta_{2,g})}}\right), \tag{2.8a}$$

   then deterministic norm condition (2.6a) *is satisfied.*

2. *If* (2.7b) *holds, $\mathbb{E}[g_k|w_k] = \nabla f(w_k)$, and*

$$|X_k| \geqslant \frac{\lambda_{\max}(A_k)(\sigma_{1,g}^2\|\nabla f(w_k)\|^2 + \sigma_{2,g}^2)}{\theta_k^2\|\nabla f(w_k)\|_{A_k}^2 + \iota_k}, \tag{2.8b}$$

   then stochastic norm condition (2.6b) *is satisfied.*

Proof. Deterministic norm condition. Consider

$$\begin{aligned}
\|g_k - \nabla f(w_k)\|_{A_k}^2 &\leqslant \lambda_{\max}(A_k)\|g_k - \nabla f(w_k)\|^2 \\
&\leqslant 4\lambda_{\max}(A_k)\left(\frac{N - |X_k|}{N}\right)^2\left(\beta_{1,g}\|\nabla f(w_k)\|^2 + \beta_{2,g}\right) \\
&\leqslant \theta_k^2\|\nabla f(w_k)\|_{A_k}^2 + \iota_k.
\end{aligned}$$

where the first inequality is due to $w^T A w \leqslant \lambda_{\max}(A_k)\|w\|^2$, the second inequality is due to (2.7a) and the analysis provided in [35, Section 3.1], and the third inequality is due to the bound on $|X_k|$ given in (2.8a). We provide the derivation of the second inequality in Appendix 9.1 for completeness.

Stochastic norm condition. Following a similar approach as in the deterministic norm condition, and using $\mathbb{E}[g_k|w_k] = \nabla f(w_k)$, we have

$$\begin{aligned}
\mathbb{E}[\|g_k - \nabla f(w_k)\|_{A_k}^2 | w_k, A_k] &\leqslant \lambda_{\max}(A_k)\mathbb{E}[\|g_k - \nabla f(w_k)\|^2 | w_k] \\
&= \lambda_{\max}(A_k)\frac{\mathbb{E}_\zeta[\|\nabla F(w_k, \zeta) - \nabla f(w_k)\|^2 | w_k]}{|X_k|} \\
&\leqslant \lambda_{\max}(A_k)\frac{\sigma_{1,g}^2\|\nabla f(w_k)\|^2 + \sigma_{2,g}^2}{|X_k|} \\
&\leqslant \theta_k^2\|\nabla f(w_k)\|_{A_k}^2 + \iota_k.
\end{aligned}$$

□

In Section 5.4.3, we provide practical strategies for choosing the gradient sample sizes $|X_k|$ at each iterate w_k, $k \in \mathbb{Z}^+$ instead of employing these pessimistic theoretical bounds that require accessing unknown problem specific constants such as $\sigma_{1,g}$ and $\sigma_{2,g}$.

In what follows, we split our convergence theory into two parts: first the deterministic sampling convergence theory where we prove a novel $\mathcal{O}\left(\frac{1}{k}\right)$ superlinear local convergence rate, followed by a section where we extend this analysis to the stochastic setting, deriving bounds *in expectation*, where our results match the *probabilistic* $\tilde{\mathcal{O}}\left(\frac{1}{\sqrt{k}}\right)$ superlinear local convergence rate. In both cases we simultaneously assume gradient and Hessian inexactness, and maintain a fixed per-iteration Hessian computational cost.

3 Deterministic Sampling Analysis

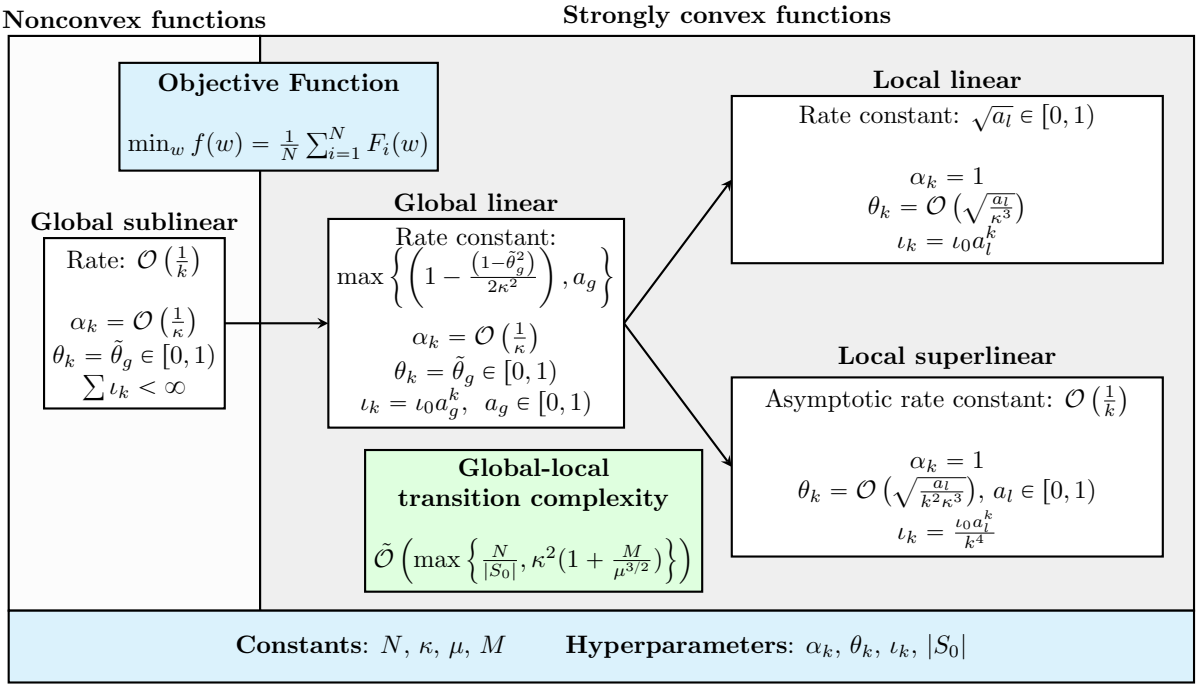

Figure 1: Overview of the results presented in this section. We characterize the main results for global and local convergence results, and their relationship to the problem constants and algorithmic hyperparameters. Here, N denotes the number of data, μ is the Hessian spectral lower bound, $\kappa = \frac{L}{\mu}$ is a condition-number like constant, and M is the Hessian Lipschitz constant.

We begin our analysis by focusing on the deterministic sampling-based algorithms for the solution of the finite-sum minimization problem (1.1), where the subsampled gradients satisfy the deterministic norm condition (2.6a), and the subsampled Hessians are deterministically sampled without replacement in a cyclic fashion. We establish theoretical global linear and sublinear convergence results for strongly convex and nonconvex functions respectively, and local superlinear convergence results when the iterates enter a strongly convex neighborhood of the optimal solution. A schematic of this analysis summarizing the convergence results is given in Figure 1. We begin with an assumption about the subsampled functions.

Assumption 3.1. *(Spectral upper bounds of subsampled Hessians). The subsampled functions are twice continuously differentiable with the eigenvalues of the subsampled Hessians bounded above where the bound depend on the sample size $|S|$. That is, for all $|S| \in \mathbb{N}$, there exists constants $0 < L_{|S|} < \infty$ such that*

$$\nabla^2 F_S(w) \preceq L_{|S|} I, \qquad \forall w \in \mathbb{R}^d. \tag{3.1}$$

Furthermore there exists constant L such that $L_{|S|} \leqslant L < \infty$ for all $|S| \in \mathbb{N}$. As a consequence, we have that $\nabla^2 f(w) \preceq LI$ for all $w \in \mathbb{R}^d$.

Before establishing theoretical convergence results, we state key inequalities due to the Assumption 3.1 that are used throughout the analysis. From Assumption 3.1 and Taylor's theorem [65, 80], it follows that

$$f(w) \leqslant f(v) + \nabla f(v)^T (w - v) + \frac{L}{2}\|w - v\|^2, \quad \forall w, v \in \mathbb{R}^d. \tag{3.2}$$

Furthermore, we also have that the path-averaged Hessians have upper bounded eigenvalues. That is, due to (2.4) and Assumption 3.1, we have

$$\widehat{H}_k \preceq \widehat{L}_k I,\ \widehat{L}_k \leqslant \sum_{i=0}^{k} \gamma_i L_{|S_i|} \leqslant L, \tag{3.3}$$

$$\widetilde{H}_k \preceq \tilde{L} I,\ \tilde{L} \leqslant \sum_{i=0}^{k} \gamma_i L_{|S_i|} + \tilde{\mu} - \lambda_{\min}(|\widehat{H}_k|) \leqslant L + \tilde{\mu}. \tag{3.4}$$

We begin our analysis by providing a technical lemma that establishes an upper bound on the difference between the objective function values at successive iterations.

Lemma 3.1. *Suppose Assumption 3.1 holds. For any w_0, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by (2.1) with the Hessian approximation given in (2.4). If the step size α_k at each iteration k is chosen such that $\alpha_k \leqslant \frac{\tilde{\mu}}{L}$. Then, for all $k \in \mathbb{Z}^+$, it follows that,*

$$f(w_{k+1}) \leqslant f(w_k) - \frac{\alpha_k}{2} \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{\alpha_k}{2} \delta_k^T \widetilde{H}_k^{-1} \delta_k, \tag{3.5}$$

where $\delta_k = g_k - \nabla f(w_k)$.

Proof. Using (3.2) and the definition of δ_k, we have

$$\begin{aligned}
f(w_{k+1}) &\leqslant f(w_k) - \alpha_k \nabla f(w_k)^T \widetilde{H}_k^{-1} g_k + \frac{L\alpha_k^2}{2} \|\widetilde{H}_k^{-1} g_k\|^2 \\
&= f(w_k) - \alpha_k \nabla f(w_k)^T \widetilde{H}_k^{-1} (\nabla f(w_k) + \delta_k) + \frac{L\alpha_k^2}{2} \|\widetilde{H}_k^{-1} (\nabla f(w_k) + \delta_k)\|^2 \qquad (3.6) \\
&= f(w_k) - \alpha_k \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) - \alpha_k \nabla f(w_k)^T \widetilde{H}_k^{-1} \delta_k \\
&\quad + \frac{L\alpha_k^2}{2} \left[\|\widetilde{H}_k^{-1} \nabla f(w_k)\|^2 + \|\widetilde{H}_k^{-1} \delta_k\|^2 + 2 \nabla f(w_k)^T \widetilde{H}_k^{-2} \delta_k \right] \\
&= f(w_k) - \alpha_k \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{L\alpha_k^2}{2} \left[\|\widetilde{H}_k^{-1} \nabla f(w_k)\|^2 + \|\widetilde{H}_k^{-1} \delta_k\|^2 \right] \\
&\quad - \alpha_k (\widetilde{H}_k^{-1/2} \nabla f(w_k))^T \left[I - L\alpha_k \widetilde{H}_k^{-1} \right] (\widetilde{H}_k^{-1/2} \delta_k). \qquad (3.7)
\end{aligned}$$

Substituting $I - L\alpha_k \widetilde{H}_k^{-1} \succeq 0$ due to $\alpha_k \leqslant \frac{\tilde{\mu}}{L}$ and using the fact that $-w^T A v \leqslant \frac{1}{2} w^T A w + \frac{1}{2} v^T A v$ for any $w, v \in \mathbb{R}^d$ and $0 \preceq A \in \mathbb{R}^{d \times d}$ in (3.7) yields

$$\begin{aligned}
f(w_{k+1}) &\leqslant f(w_k) - \alpha_k \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{L\alpha_k^2}{2} \left[\|\widetilde{H}_k^{-1} \nabla f(w_k)\|^2 + \|\widetilde{H}_k^{-1} \delta_k\|^2 \right] \\
&\quad + \frac{\alpha_k}{2} (\widetilde{H}_k^{-1/2} \nabla f(w_k))^T [I - L\alpha_k \widetilde{H}_k^{-1}] (\widetilde{H}_k^{-1/2} \nabla f(w_k)) \\
&\quad + \frac{\alpha_k}{2} (\widetilde{H}_k^{-1/2} \delta_k))^T [I - L\alpha_k \widetilde{H}_k^{-1}] (\widetilde{H}_k^{-1/2} \delta_k) \\
&= f(w_k) - \frac{\alpha_k}{2} \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{\alpha_k}{2} \delta_k^T \widetilde{H}_k^{-1} \delta_k.
\end{aligned}$$

□

3.1 Global Convergence

In this section we derive global convergence rates. We first start with strongly convex functions, where we make the following assumption about the objective function.

Assumption 3.2. *(Global strong convexity). The eigenvalues of the Hessians are all positive and are bounded away from zero. That is, there exists constant $\mu > 0$ such that*

$$\nabla^2 f(w) \geq \mu I. \tag{3.8}$$

From Assumption 3.2, we have

$$\|\nabla f(w)\|^2 \geqslant 2\mu(f(w) - f(w^*)), \qquad \forall w \in \mathbb{R}^d, \tag{3.9}$$

where w^* is the unique optimal solution of (1.1) or (1.2) (see [19] for the proof). We are now ready to provide the global linear convergence result for strongly convex functions.

Theorem 3.2. *(Global linear convergence, deterministic sampling). Suppose Assumptions 3.1 and 3.2 hold. For any $w_0 \in \mathbb{R}^d$, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by (2.1) where the Hessian approximation is given in (2.4) and the gradient approximations g_k satisfies the Condition 2.1 with $\iota_{k+1} = \iota_k a_g$ for some $\iota_0 > 0$ and $a_g \in [0, 1)$. Then, if g_k satisfies deterministic norm condition (2.6a) with $A_k = \widetilde{H}_k^{-1}$ and $\theta_k = \tilde{\theta}_g \in [0, 1)$, and the step size is chosen such that $\alpha_k = \alpha \leqslant \frac{\tilde{u}}{L}$,*

$$f(w_k) - f(w^*) \leqslant \tilde{C}_1 \tilde{\rho}_1^k, \tag{3.10}$$

$$\tilde{C}_1 := \max\left\{ f(w_0) - f(w^*), \frac{\tilde{L}\iota_0}{\mu\left(1 - \tilde{\theta}_g^2\right)} \right\}, \text{ and } \tilde{\rho}_1 := \max\left\{ 1 - \frac{\alpha\mu\left(1 - \tilde{\theta}_g^2\right)}{2\tilde{L}}, a_g \right\}.$$

Proof. From (2.6a) and (3.5), we have

$$\begin{aligned} f(w_{k+1}) &\leqslant f(w_k) - \frac{\alpha_k}{2}\nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{\alpha_k}{2}\delta_k^T \widetilde{H}_k^{-1}\delta_k \\ &\leqslant f(w_k) - \frac{\alpha_k}{2}\nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{\alpha_k\theta_k^2}{2}\nabla f(w_k)^T \widetilde{H}_k^{-1}\nabla f(w_k) + \frac{\alpha_k\iota_k}{2} \\ &\leqslant f(w_k) - \frac{\alpha_k(1-\theta_k^2)}{2\tilde{L}}\|\nabla f(w_k)\|^2 + \frac{\alpha_k\iota_k}{2}, \end{aligned} \tag{3.11}$$

where the last inequality is due to (3.4). Subtracting $f(w^*)$ from both sides of (3.11) and using (3.9), it follows that

$$f(w_{k+1}) - f(w^*) \leqslant \left(1 - \alpha_k \frac{\mu(1-\theta_k^2)}{\tilde{L}}\right)(f(w_k) - f(w^*)) + \frac{\alpha_k\iota_k}{2}. \tag{3.12}$$

We will use induction to show the rest of the proof. Substitute $\alpha_k = \alpha$ and $\theta_k = \tilde{\theta}_g$ in (3.12). We note that (3.10) trivially holds for $k = 0$. Now, suppose that (3.10) holds for some k. Considering (3.12), we get,

$$\begin{aligned} f(w_{k+1}) - f(w^*) &\leqslant \left(1 - \alpha\frac{\mu(1-\tilde{\theta}_g^2)}{\tilde{L}}\right)\tilde{C}_1\tilde{\rho}_1^k + \frac{\alpha\iota_0 a_g^k}{2} \\ &= \tilde{C}_1\tilde{\rho}_1^k\left(1 - \alpha\frac{\mu(1-\tilde{\theta}_g^2)}{\tilde{L}} + \frac{\alpha\iota_0}{2\tilde{C}_1}\left(\frac{a_g}{\tilde{\rho}_1}\right)^k\right) \\ &\leqslant \tilde{C}_1\tilde{\rho}_1^k\left(1 - \alpha\frac{\mu(1-\tilde{\theta}_g^2)}{\tilde{L}} + \frac{\alpha\iota_0}{2\tilde{C}_1}\right) \\ &\leqslant \tilde{C}_1\tilde{\rho}_1^k\left(1 - \alpha\frac{\mu(1-\tilde{\theta}_g^2)}{2\tilde{L}}\right) = \tilde{C}_1\tilde{\rho}_1^{k+1}, \end{aligned}$$

where the inequalities are due to the definitions of $\tilde{C}_1$ and $\tilde{\rho}_1$.

□

Remark 3.1. We make the following remarks about the Theorem 3.2

- For $\tilde{\theta}_g = 0$ and $a_g = 0$ ($\iota_k = 0$), Theorem 3.2 recovers the classical global linear convergence result for Newton's method. We note that the rate constant $\left(1 - \frac{\alpha\mu}{2\tilde{L}}\right)$ is worse than that of steepest descent method, which is an artifact of global convergence analysis of Newton's method. As is typical with Newton methods, the global convergence bounds are more pessimistic than similar bounds for first-order methods due to taking into account worst-case spectral bounds of the Hessian. We note that in practice we do not expect significantly worse convergence rates for second-order methods, indeed our numerical results demonstrate that Hessian-averaged Newton methods are able to take larger steps than first-order methods.
- We do not consider other cases of deterministic norm condition in this setting of $A_k = I$ as it enforces stringent restrictions on the choice of $\tilde{\theta}_g$.

Next we consider general nonconvex functions where we provide the following sublinear convergence results.

Theorem 3.3. *(Global sublinear convergence, deterministic sampling). Suppose Assumption 3.1 holds and the objective function f is bounded below by f_{min}. For any w_0, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by (2.1) where the gradient approximations g_k satisfy the Condition 2.1with $\sum_{i=0}^{\infty} \iota_k = \tilde{\iota} < \infty$. Then, if g_k satisfies deterministic norm condition (2.6a) with $A_k = \widetilde{H}_k^{-1}$ and $\theta_k = \tilde{\theta}_g \in [0, 1)$, and the step size is chosen such that $\alpha_k = \alpha \leqslant \frac{\tilde{\mu}}{\tilde{L}}$, then $\lim_{k\to\infty} \|\nabla f(w_k)\|^2 = 0$ and for any positive integer T,*

$$\min_{0\leqslant k\leqslant T-1} \|\nabla f(w_k)\|^2 \leqslant \frac{\tilde{L}}{(1-\tilde{\theta}_g^2)T}\left(\frac{2(f(w_0) - f_{min})}{\alpha} + \tilde{\iota}\right). \tag{3.13}$$

Proof. Substituting $\alpha_k = \alpha$ and $\theta_k = \theta_g$ in (3.11), rearranging the terms, and summing up the inequalities from $k = 0$ to $T - 1$ yields

$$\sum_{k=0}^{T-1} \|\nabla f(w_k)\|^2 \leqslant \frac{\tilde{L}}{(1-\tilde{\theta}_g^2)}\left(\frac{2(f(w_0) - f(w_T))}{\alpha} + \sum_{k=0}^{T-1} \iota_k\right) \leqslant \frac{\tilde{L}}{(1-\tilde{\theta}_g^2)}\left(\frac{2(f(w_0) - f_{\min})}{\alpha} + \tilde{\iota}\right),$$

where the last inequality is due to $f(w_T) \geqslant f_{\min}$ and $\sum_{k=0}^{\infty} \iota_k = \tilde{\iota}$. Taking the limits on T yields $\lim_{k\to\infty} \|\nabla f(w_k)\|^2 = 0$. In addition, we have

$$\min_{0\leqslant k\leqslant T-1} \|\nabla f(w_k)\|^2 \leqslant \frac{1}{T}\sum_{k=0}^{T-1} \|\nabla f(w_k)\|^2 \leqslant \frac{\tilde{L}}{(1-\tilde{\theta}_g^2)T}\left(\frac{2(f(w_0) - f_{\min})}{\alpha} + \tilde{\iota}\right).$$

□

3.2 Local Convergence

We now provide local superlinear rates of convergence results for the iterates generated by (2.1) when unit step size is eventually employed. We make the following standard assumption in local analysis of Newton-type methods that the Hessians are Lipschitz continuous. That is,

Assumption 3.3. *(Lipschitz continuous Hessians). For any $|S| \in \mathbb{N}$, there exists a constant $0 < M_{|S|} < \infty$ such that*

$$\|\nabla^2 F_S(w) - \nabla^2 F_S(v)\| \leqslant M_{|S|}\|w - v\| \qquad \forall w, v \in \mathbb{R}^d. \tag{3.14}$$

Furthermore, there exists constant M such that $M_{|S|} < M < \infty$ for all $|S| \in \mathbb{N}$. As a consequence, we have that $\|\nabla^2 f(w) - \nabla^2 f(v)\| \leqslant M\|w - v\|$ for all $w, v \in \mathbb{R}^d$.

We start by presenting a fundamental lemma that establishes a generalized linear-quadratic bound on the iterate distance to optimality when unit step size is employed at that iteration ($\alpha_k = 1$).

Lemma 3.4. *Suppose Assumptions 3.1 and 3.3 hold. For any $w_0 \in \mathbb{R}^d$, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by (2.1) where the Hessian approximation is given in (2.4) and the gradient approximations g_k satisfies the Condition 2.1 with $\lambda_{\min}(A_k) \geqslant \lambda_A$ and $\frac{\lambda_{\max}(A_k)}{\lambda_{\min}(A_k)} \leqslant \kappa_A$ for some positive constants $\lambda_A, \kappa_A < \infty$. If at any iteration $k \in \mathbb{N}$, unit step size is chosen ($\alpha_k = 1$). Then, if g_k satisfies deterministic norm condition (2.6a)*

$$\begin{aligned}\|w_{k+1} - w^*\| \leqslant \frac{M}{2\tilde{\mu}}\|w_k - w^*\|^2 + \frac{1}{\tilde{\mu}}\|(\widetilde{H}_k - \nabla^2 f(w_k))(w_k - w^*)\| \\ + \frac{L\sqrt{\kappa_A}}{\tilde{\mu}}\theta_k\|w_k - w^*\| + \frac{\sqrt{\iota_k}}{\tilde{\mu}\sqrt{\lambda_A}},\end{aligned} \tag{3.15}$$

where w^* is an optimal solution.

Proof. We proceed by decomposing the iterate update (2.1) into three terms: Newton update term, Hessian error term, and gradient error term as follows.

$$\begin{aligned}&\|w_{k+1} - w^*\| \\ &\quad\leqslant \|\widetilde{H}_k^{-1}\|\|\widetilde{H}_k(w_k - w^*) - g_k\| \\ &\quad\leqslant \frac{1}{\tilde{\mu}}\left(\underbrace{\|\nabla^2 f(w_k)(w_k - w^*) - \nabla f(w_k)\|}_{\text{Newton update}} + \underbrace{\|(\widetilde{H}_k - \nabla^2 f(w_k))(w_k - w^*)\|}_{\text{Hessian error}} + \underbrace{\|g_k - \nabla f(w_k)\|}_{\text{gradient error}}\right).\end{aligned} \tag{3.16}$$

The Newton update term is standard in Newton-type methods and has been analyzed in many prior works and we provide it here for the sake of completeness. Using Assumption 3.3, it follows that

$$\begin{aligned}\|\nabla^2 f(w_k)(w_k - w^*) - \nabla f(w_k)\| &= \left\|\nabla^2 f(w_k)(w_k - w^*) - \int_{t=0}^1 \nabla^2 f(w_k + t(w^* - w_k))(w_k - w^*)dt\right\| \\ &\leqslant \|w_k - w^*\| \int_{t=0}^1 \|\nabla^2 f(w_k) - \nabla^2 f(w_k + t(w^* - w_k))\|dt \\ &\leqslant M\|w_k - w^*\|^2 \int_{t=0}^1 tdt = \frac{M}{2}\|w_k - w^*\|^2.\end{aligned} \tag{3.17}$$

Next, we analyze the gradient error term. If g_k satisfies deterministic norm condition (2.6a), then we have that

$$\begin{aligned}\sqrt{\lambda_{\min}(A_k)}\|g_k - \nabla f(w_k)\| &\leqslant \|g_k - \nabla f(w_k)\|_{A_k} \\ &\leqslant \sqrt{\theta_k^2\|\nabla f(w_k)\|_{A_k}^2 + \iota_k} \\ &\leqslant \sqrt{\lambda_{\max}(A_k)}\theta_k\|\nabla f(w_k)\| + \sqrt{\iota_k}.\end{aligned} \tag{3.18}$$

Rearranging the terms in the above inequality and using Assumption 3.1, we get

$$\|g_k - \nabla f(w_k)\| \leqslant L\sqrt{\frac{\lambda_{\max}(A_k)}{\lambda_{\min}(A_k)}}\theta_k\|w_k - w^*\| + \sqrt{\frac{\iota_k}{\lambda_{\min}(A_k)}}. \tag{3.19}$$

Combining (3.16), (3.17), and (3.19) yields (3.15).

□

Lemma 3.4 establishes the dependence of the iterate distance to optimality on the Hessian approximation error. In the following lemmas, we derive bounds for individual terms, in the service of establishing the local rate of convergence. We first decompose the error into different error terms as stated in the following lemma.

Lemma 3.5. *Suppose Assumption 3.3 holds. For any iteration $k \in \mathbb{Z}^+$, the error in the Hessian approximation is upper bounded as*

$$\begin{aligned}
&\|(\widetilde{H}_k - \nabla^2 f(w_k))(w_k - w^*)\| \\
&\quad \leqslant \|(\widetilde{H}_k - \widehat{H}_k)(w_k - w^*)\| + 3M\|w_k - w^*\|^2 + \sum_{i=0}^{k} \gamma_i \|(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*))(w_k - w^*)\| \\
&\qquad + \left\| \left(\sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \right) (w_k - w^*) \right\|. \qquad (3.20)
\end{aligned}$$

Proof. Let $e_k = w_k - w^*$. The Hessian approximation error can be decomposed into two terms.

$$(\widetilde{H}_k - \nabla^2 f(w_k))e_k = \underbrace{(\widetilde{H}_k - \widehat{H}_k)e_k}_{\text{nonconvex error}} + (\widehat{H}_k - \nabla^2 f(w_k))e_k, \qquad (3.21)$$

where the first term is arising due to the nonconvexity. Now, using the decomposition of the second term given in (2.3), we get

$$\begin{aligned}
&\|(\widehat{H}_k - \nabla^2 f(w_k))e_k\| \\
&\quad \leqslant \left\| \left(\sum_{i=0}^{k} \gamma_i \left(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w_k) \right) e_k \right) \right\| + \left\| \left(\sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w_k) - \nabla^2 f(w_k) \right) e_k \right\|. \qquad (3.22)
\end{aligned}$$

Considering the first term in (3.22), using $\sum_{i=0}^{k} \gamma_i = 1$ and Assumption 3.3, we get

$$\begin{aligned}
&\left\| \left(\sum_{i=0}^{k} \gamma_i \left(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w_k) \right) \right) e_k \right\| \\
&\quad \leqslant \sum_{i=0}^{k} \gamma_i \|(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*))e_k\| + \|(\nabla^2 F_{S_i}(w^*) - \nabla^2 F_{S_i}(w_k))e_k\| \\
&\quad \leqslant \sum_{i=0}^{k} \gamma_i \|(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*))e_k\| + M\|w_k - w^*\|^2. \qquad (3.23)
\end{aligned}$$

Considering the second term in (3.22), we get

$$\begin{aligned}
&\left\| \left(\sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w_k) - \nabla^2 f(w_k) \right) e_k \right\| \\
&\quad \leqslant \left\| \sum_{i=0}^{k} \gamma_i (\nabla^2 F_{S_i}(w_k) - \nabla^2 F_{S_i}(w^*))e_k \right\| + \left\| \left(\sum_{i=1}^{k} \gamma_i \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \right) e_k \right\| \\
&\qquad + \|(\nabla^2 f(w_k) - \nabla^2 f(w^*))e_k\| \\
&\quad \leqslant 2M\|w_k - w^*\|^2 + \left\| \left(\sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \right) e_k \right\|. \qquad (3.24)
\end{aligned}$$

Combining (3.21), (3.22), (3.23), and (3.24) yields (3.20). □

We will analyze each term in the upper bound on the Hessian approximation error established in Lemma 3.5. We achieve this by utilizing a key assumption in the local analysis of Newton methods where we assume that the iterates generated by (2.1) eventually enter a locally strongly convex regime. That is, we make the following assumption about the iterates generated by (2.1) with α_k specified in Section 3.1 and g_k satisfying deterministic norm condition (2.6a).

Assumption 3.4. *(Local strong convexity). For any w_0, there exists $\nu > 0$ such that for all $k \in \{j \in \mathbb{N} \mid \|\nabla f(w_k)\| \leqslant \nu\}$, we have that $\nabla^2 f(w_k) \succeq \mu I$, where $\{w_k : k \in \mathbb{N}\}$ are iterates generated by (2.1). Moreover, we also assume that the following well-known inequalities associated with strong convexity also hold with respect to a local solution w^* ($\|\nabla f(w^*)\| = 0$).*

$$\|\nabla f(w_k)\|^2 \geqslant 2\mu(f(w_k) - f(w^*)) \geqslant \mu^2 \|w_k - w^*\|^2. \tag{3.25}$$

Remark 3.2. We note that similar assumptions have been made in the constrained setting [7, Assumption 5.1]. This assumption is required not for all iterates, but only for those obtained after running the algorithm for a sufficiently large number of iterations, such that the iterates enter a locally strongly convex regime. This assumption is trivially satisfied when the functions are globally strongly convex (see Assumption 3.2). Due to the global convergence results established in Section 3.1, this assumption also implies that the iterates are indeed converging to a second-order stationary point ($\nabla f(w^*) = 0$ and $\nabla^2 f(w^*) \succ 0$). Such assumptions are commonly employed in local analysis of Newton-type methods, albeit in the form of proximity to a second-order stationary point w^*. That is, $\|w_k - w^*\| \leqslant \nu$.

In the next lemma, we establish upper bounds for the terms in Lemma 3.5. The main approach in the proof of this lemma is that using global convergence results, the iterates will enter the locally strongly convex regime after the global sublinear phase established in Theorem 3.3. Furthermore, once the iterates enters this phase, they will remain in this regime thereby achieving the linear convergence as established in Theorem 3.2.

Lemma 3.6. *Suppose Assumptions 3.1, 3.3 and 3.4 hold. For any $w_0 \in \mathbb{R}^d$, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by (2.1) where the Hessian approximation is given in (2.4) with $\gamma_i = \frac{1}{k+1}$ for all $i = 0, \cdots, k$, and the gradient approximations g_k satisfies the Condition 2.1 with $\sum_{i=0}^{\infty} \iota_k = \tilde{\iota} < \infty$. If A_k and the corresponding step size α_k are chosen according to Theorem 3.3. Then there exists $k_{lin} \geqslant 0$ such that for any $k \geqslant k_{lin}$ if $\iota_{k+1} = \iota_k a_g$ for some $\iota_{k_{lin}} \geqslant 0$ and $a_g \in [0, 1)$, we have that if g_k satisfies deterministic norm condition (2.6a), then there exists a constant $C_{p,d}$ such that*

$$\sum_{i=0}^{k} \gamma_i \|(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*))(w_k - w^*)\| \leqslant \frac{C_{p,d}}{k+1} \|w_k - w^*\|. \tag{3.26}$$

Proof. Since the conditions of Theorem 3.3 are satisfied, from (3.13) we have that, for any positive integer T with $\alpha_k = \alpha = \frac{\tilde{\mu}}{L}$,

$$\min_{0 \leqslant k \leqslant T-1} \|\nabla f(w_k)\|^2 \leqslant \frac{\tilde{L}}{\left(1 - \tilde{\theta}_g^2\right) T} \left(\frac{2(f(w_0) - f_{\min})}{\alpha} + \tilde{\iota} \right).$$

Choosing

$$T \geqslant \tilde{k}_{\text{lin}} := \frac{\tilde{L} L}{\left(1 - \tilde{\theta}_g^2\right) \mu \nu^2} \left(\frac{2(f(w_0) - f_{\min})}{\alpha} + \tilde{\iota} \right), \tag{3.27}$$

we get

$$\min_{0 \leqslant k \leqslant T-1} \|\nabla f(w_k)\|^2 \leqslant \frac{\nu^2 \mu}{L} \leqslant \nu^2.$$

Let $k_{\text{lin}} \leqslant \tilde{k}_{\text{lin}}$ be the first iterate at which $\|\nabla f(w_{k_{\text{lin}}})\|^2 \leqslant \nu^2$. Due to Assumption 3.4, it follows that $\nabla^2 f(w_{k_{\text{lin}}}) \succeq \mu I$. We will now show that starting with this iterate $w_{k_{\text{lin}}}$, all the following iterates will remain in this locally strongly convex phase. That is, we will show that $\|\nabla f(w_k)\| \leqslant \nu$ for all $k \geqslant k_{\text{lin}}$. We use induction to prove this statement. Note that for the base case of $k = k_{\text{lin}}$, it is trivially satisfied. Let us assume that the statement is true till some iteration $k-1 > k_{\text{lin}}$. Using the strongly convex results established in Theorem 3.2 with $\alpha_k = \alpha = \frac{\tilde{\mu}}{L}$, starting with iterate $w_{k_{\text{lin}}}$, from (3.10), we get,

$$\|\nabla f(w_k)\|^2 \leqslant 2L(f(w_k) - f(w^*)) \leqslant 2L\tilde{C}_1 \tilde{\rho}_1^{k-k_{\text{lin}}} \leqslant 2L\tilde{C}_1,$$

where the first inequality is a well-known result for functions with Lipschitz continuous gradients [12, Eq (3.16)]. Moreover, choosing $\iota_{k_{\text{lin}}} \leqslant \frac{\mu(1-\tilde{\theta}_g^2)\nu^2}{2L\tilde{L}}$, we have,

$$\begin{aligned}
\|\nabla f(w_k)\|^2 \leqslant 2L\tilde{C}_1 &= 2L \max\left\{ f(w_{k_{\text{lin}}}) - f(w^*), \frac{\tilde{L}\iota_{k_{\text{lin}}}}{\mu(1-\tilde{\theta}_g^2)} \right\} \\
&\leqslant 2L \max\left\{ f(w_{k_{\text{lin}}}) - f(w^*), \frac{\nu^2}{2L} \right\} \\
&\leqslant 2L \max\left\{ \frac{\|\nabla f(w_{k_{\text{lin}}})\|^2}{2\mu}, \frac{\nu^2}{2L} \right\} \leqslant \nu^2,
\end{aligned} \tag{3.28}$$

where the third inequality is due to (3.25) and the last inequality is due to $\|\nabla f(w_{k_{\text{lin}}})\|^2 \leqslant \frac{\nu^2\mu}{L}$.

Therefore, for all $k \geqslant k_{\text{lin}}$, we conclude that $\|\nabla f(w_k)\| \leqslant \nu$ and consequently from Assumption 3.4, we have that $\nabla^2 f(w_k) \succeq \mu I$. Let $e_k = w_k - w^*$, and consider

$$\begin{aligned}
&\sum_{i=0}^{k} \gamma_i \|(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*))e_k\| \\
&\quad = \frac{1}{k+1}\left(\sum_{i=0}^{k_{\text{lin}}-1} \|(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*))e_k\| + \sum_{i=k_{\text{lin}}}^{k} \|(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*))e_k\| \right) \\
&\quad \leqslant \frac{2Lk_{\text{lin}}}{k+1}\|w_k - w^*\| + \frac{M}{k+1}\sum_{i=k_{\text{lin}}}^{k} \|w_i - w^*\|\|w_k - w^*\|,
\end{aligned} \tag{3.29}$$

where the first term in the inequality is due to $\|\nabla^2 F_{S_i}(\cdot)\| \leqslant L$ and the second term is due to Assumption 3.3. Now, starting with $k = k_{\text{lin}}$, the iterates are in locally strongly convex regime. Therefore, from (3.10), (3.25), and (3.28), we have for any $k \geqslant k_{\text{lin}}$,

$$\|w_k - w^*\|^2 \leqslant \frac{2}{\mu}(f(w_k) - f(w^*)) \leqslant \frac{2\tilde{C}_1\tilde{\rho}_1^{k-k_{\text{lin}}}}{\mu} \leqslant \frac{\nu^2\tilde{\rho}_1^{k-k_{\text{lin}}}}{\mu^2}. \tag{3.30}$$

Summing this inequality from k_{lin} to k yields,

$$\sum_{i=k_{\text{lin}}}^{k} \|w_i - w^*\| \leqslant \sum_{i=k_{\text{lin}}}^{k} \frac{\nu(\sqrt{\tilde{\rho}_1})^{i-k_{\text{lin}}}}{\mu} \leqslant \frac{\nu}{\mu}\sum_{i=0}^{k-k_{\text{lin}}} (\sqrt{\tilde{\rho}_1})^i < \frac{\nu}{\mu\left(1-\sqrt{\tilde{\rho}_1}\right)}. \tag{3.31}$$

Substituting (3.31) in (3.29) and choosing

$$C_{p,d} := 2Lk_{\text{lin}} + \frac{\nu M}{\mu(1-\sqrt{\tilde{\rho}_1})}, \tag{3.32}$$

yields

$$\sum_{i=0}^{k} \gamma_i \|(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*))e_k\| \leqslant \frac{\|e_k\|}{k+1}\left(2Lk_{\text{lin}} + \frac{\nu M}{\mu\left(1-\sqrt{\tilde{\rho}_1}\right)}\right) = \frac{C_{p,d}}{k+1}\|w_k - w^*\|. \tag{3.33}$$

□

Remark 3.3. If the functions are globally strongly convex (Assumption 3.2 holds) then there is no sublinear convergent phase in the algorithm. That is, $k_{\text{lin}} = 0$ in this setting.

We will now establish that the nonconvex error term in Lemma 3.5 vanishes for sufficiently large number of iterations.

Lemma 3.7. *Suppose conditions of Lemma 3.6 hold. Let $\tilde{\mu} \leqslant \frac{\mu}{2}$, and g_k satisfies deterministic norm condition* (2.6a) *and the sample sets S_k are chosen deterministically without replacement in a cyclic fashion such that $n|S_k| = N$ for some $n \in \mathbb{N}$ with $n \geqslant 1$. Let $\lambda_{\min}(\nabla^2 F_{S_i}(w_i)) \geqslant -\hat{\lambda}$ for some $\hat{\lambda} \in [0, \infty)$. Then there exists $k_{non} \geqslant k_{lin}$ such that for all $k \geqslant k_{non}$, $\widetilde{H}_k = \widehat{H}_k$.*

Proof. Any iteration $k \geqslant k_{\text{lin}}$ can be written as $k = k_{\text{lin}} + nm - 1 + k_{\text{rem}}$ where $m \in \mathbb{Z}^+$, $k_{\text{rem}} \in \mathbb{N}$ and $1 \leqslant k_{\text{rem}} \leqslant n-1$. Let $v \in \mathbb{R}^n$ be any vector and consider

$$\begin{aligned}
v^T \widehat{H}_k v &= \frac{1}{k+1}\sum_{i=0}^{k} v^T \nabla^2 F_{S_i}(w_i) v \\
&= \frac{1}{k+1}\sum_{i=0}^{k_{\text{lin}}-1} v^T \nabla^2 F_{S_i}(w_i) v + \frac{1}{k+1}\sum_{i=k_{\text{lin}}}^{k} v^T \nabla^2 F_{S_i}(w_i) v \\
&\geqslant \frac{-\hat{\lambda} k_{\text{lin}}}{k+1}\|v\|^2 + \frac{1}{k+1}\sum_{i=k_{\text{lin}}}^{k} v^T \nabla^2 F_{S_i}(w_i) v \qquad (3.34) \\
&= \frac{-\hat{\lambda} k_{\text{lin}}}{k+1}\|v\|^2 + \frac{1}{k+1}\sum_{i=k_{\text{lin}}}^{k} v^T (\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*)) v + \frac{1}{k+1}\sum_{i=k_{\text{lin}}}^{k} v^T \nabla^2 F_{S_i}(w^*) v. \qquad (3.35)
\end{aligned}$$

Using (3.31), we get

$$\begin{aligned}
\frac{1}{k+1}\sum_{i=k_{\text{lin}}}^{k} v^T (\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*)) v &\geqslant -\frac{1}{k+1}\sum_{i=k_{\text{lin}}}^{k} \|\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*)\|\|v\|^2 \\
&\geqslant \frac{-M\|v\|^2}{k+1}\sum_{i=k_{\text{lin}}}^{k} \|w_i - w^*\| \\
&\geqslant \frac{-M\nu}{(k+1)\mu(1-\sqrt{\tilde{\rho}_1})}\|v\|^2. \qquad (3.36)
\end{aligned}$$

Now consider

$$\begin{aligned}
&\frac{1}{k+1}\sum_{i=k_{\text{lin}}}^{k} v^T\nabla^2 F_{S_i}(w^*)v \\
&\quad= \frac{1}{k+1}\sum_{i=k_{\text{lin}}}^{k_{\text{lin}}+nm-1} v^T\nabla^2 F_{S_i}(w^*)v + \frac{1}{k+1}\sum_{i=k_{\text{lin}}+nm}^{k_{\text{lin}}+nm-1+k_{\text{rem}}} v^T\nabla^2 F_{S_i}(w^*)v \\
&\quad= \frac{nm}{k+1}v^T\nabla^2 f(w^*)v + \frac{1}{k+1}\sum_{i=k_{\text{lin}}+nm}^{k_{\text{lin}}+nm-1+k_{\text{rem}}} v^T\nabla^2 F_{S_i}(w^*)v \\
&\quad\geqslant \frac{nm\mu}{k+1}\|v\|^2 - \frac{\hat{\lambda}k_{\text{rem}}}{k+1}\|v\|^2, \qquad (3.37)
\end{aligned}$$

where the second equality is due to the fact that sample sets S_k are chosen deterministically without replacement in a cyclic fashion which implies that $\frac{1}{n}\sum_{i=j}^{j+n-1}\nabla^2 F_{S_i}(w^*) = \nabla^2 f(w^*)$ for any $j \in \mathbb{N}$. Let

$$k_{\text{non}} := \max\left\{4(1+\tfrac{\hat{\lambda}}{\mu})(k_{\text{lin}}+n-1), \tfrac{4M\nu}{\mu^2(1-\sqrt{\tilde{\rho}_1})}\right\} - 1. \qquad (3.38)$$

Combining (3.35), (3.36), (3.37), and using (3.38), we get

$$\begin{aligned}
\frac{v^T\widehat{H}_k v}{\|v\|^2} &\geqslant \frac{nm\mu}{k+1} - \frac{\hat{\lambda}(k_{\text{lin}}+k_{\text{rem}})}{k+1} - \frac{M\nu}{(k+1)\mu(1-\sqrt{\tilde{\rho}_1})} \\
&= \mu\left(\frac{k+1-(k_{\text{lin}}+k_{\text{rem}})}{k+1} - \frac{\hat{\lambda}(k_{\text{lin}}+k_{\text{rem}})}{\mu(k+1)} - \frac{M\nu}{(k+1)\mu^2(1-\sqrt{\tilde{\rho}_1})}\right) \\
&\geqslant \mu(1-\tfrac{1}{4}-\tfrac{1}{4}) = \frac{\mu}{2} \geqslant \tilde{\mu}. \qquad (3.39)
\end{aligned}$$

Therefore, $\lambda_{\min}(\widehat{H}_k) \geqslant \tilde{\mu}$ and $\widetilde{H}_k = \widehat{H}_k$ for all $k \geqslant k_{\text{non}}$.

□

To analyze the last term in Lemma 3.5, we make the following standard assumption about individual Hessian components.

Assumption 3.5. *(Hessian approximations, deterministic case). The individual component Hessians are bounded relative to the Hessian of the objective function f at the optimal solution w^*. That is, for the finite-sum problem, there exist constants $\beta_{1,H}, \beta_{2,H} \geqslant 0$ such that*

$$\|\nabla^2 F_i(w^*)\|^2 \leqslant \beta_{1,H}\|\nabla^2 f(w^*)\|^2 + \beta_{2,H}. \qquad (3.40)$$

Remark 3.4. We note that Assumption 3.5 is relatively weak compared to similar assumptions made in the literature [13], as it requires the Hessian components (or its variance) to be bounded only at the optimal solution instead at all iterates $w \in \mathbb{R}^d$.

Lemma 3.8. *Suppose Assumptions 3.3 and 3.5 hold. If $\gamma_i = \frac{1}{k+1}$ for all $i = 0, \cdots, k$, and the sample sets S_k are chosen deterministically without replacement in a cyclic fashion such that $n|S_k| = N$ for some $n \in \mathbb{N}$.*

$$\left\|\left(\sum_{i=0}^{k}\gamma_i\nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*)\right)(w_k - w^*)\right\| \leqslant \frac{C_{s,d}}{k+1}\frac{(n-1)^2}{n}\|w_k - w^*\|. \qquad (3.41)$$

Proof. Any iteration can be written as $k = nm - 1 + k_{\text{rem}}$ where $m \in \mathbb{Z}^+$, $k_{\text{rem}} \in \mathbb{N}$ and $1 \leqslant k_{\text{rem}} \leqslant n-1$. Consider,

$$\begin{aligned}
&\sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \\
&= \frac{1}{k+1} \sum_{i=0}^{nm-1} \nabla^2 F_{S_i}(w^*) - \frac{nm}{k+1} \nabla^2 f(w^*) + \frac{1}{k+1} \sum_{i=nm}^{k} \nabla^2 F_{S_i}(w^*) - \frac{k_{\text{rem}}}{k+1} \nabla^2 f(w^*) \\
&= \frac{nm}{k+1} \left(\frac{1}{nm} \sum_{i=0}^{nm-1} \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \right) + \frac{k_{\text{rem}}}{k+1} \left(\frac{1}{k_{\text{rem}}} \sum_{i=nm}^{k} \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \right) \\
&= \frac{k_{\text{rem}}}{k+1} \left(\nabla^2 F_{S^\dagger}(w^*) - \nabla^2 f(w^*) \right) \leqslant \frac{n-1}{k+1} \left(\nabla^2 F_{S^\dagger}(w^*) - \nabla^2 f(w^*) \right), \qquad (3.42)
\end{aligned}$$

where the third equality is due to the fact that $\frac{1}{n} \sum_{i=0}^{n-1} \nabla^2 F_{S_i}(w^*) = \nabla^2 f(w^*)$ and $S^\dagger = \cup_{i=(n-1)m+1}^{k} S_i$. Following a similar approach in establishing the deterministic bounds on gradient approximation error given in [35, Section 3.1] (which we restate in Appendix 9.1 for completeness) and using $|S^\dagger| \geqslant |S_k|$, we get

$$\begin{aligned}
\|\nabla^2 F_{S^\dagger}(w^*) - \nabla^2 f(w^*)\|^2 &\leqslant 4 \left(\frac{N - |S^\dagger|}{N} \right)^2 \left(\beta_{1,H} \|\nabla^2 f(w^*)\|^2 + \beta_{2,H} \right) \\
&\leqslant 4 \left(1 - \frac{1}{n} \right)^2 \left(\beta_{1,H} L^2 + \beta_{2,H} \right). \qquad (3.43)
\end{aligned}$$

Substituting (3.43) in (3.42) and choosing

$$C_{s,d} := 2\sqrt{\beta_{1,H} L^2 + \beta_{2,H}}, \qquad (3.44)$$

yields

$$\begin{aligned}
\left\| \left(\sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \right) (w_k - w^*) \right\| &\leqslant \frac{2\sqrt{\beta_{1,H} L^2 + \beta_{2,H}} (n-1)^2}{(k+1)n} \|w_k - w^*\| \\
&= \frac{C_{s,d}}{k+1} \frac{(n-1)^2}{n} \|w_k - w^*\|.
\end{aligned}$$

□

Remark 3.5. Lemma 3.8 establishes the bound on the sampling error in terms of the sample size $|S_0|$ and the iteration number k. Deterministic sampling without replacement (cyclic manner) has rate $\mathcal{O}\left(\frac{1}{k}\right)$, instead of the $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ rate for stochastic Hessians; this result is proven probabilistically in [46, 64], and in expectation in the next section. Moreover, the sampling error becomes zero after finishing every cycle in the deterministic sampling case.

Before proceeding to the main result, we provide the following technical lemma that establishes the linear and superlinear convergence of generic sequences where each term is bounded above by the previous term in a relaxed linear-quadratic manner.

Lemma 3.9. *Suppose* $\{z_k : k \in \mathbb{Z}^+\}$ *is a non-negative sequence that satisfies*

$$z_{k+1} \leqslant q z_k^2 + \tau_k z_k + o_k \qquad (3.45)$$

for any given non-negative constant q, non-negative sequence $\{\tau_k : k \in \mathbb{Z}^+\}$ with $\tau_{k+1} \leqslant \tau_k$ for all $k \in \mathbb{N}$, and non-negative sequence $\{o_k : k \in \mathbb{Z}^+\}$. If $z_0 \leqslant \frac{\upsilon}{3q}$, $\tau_0 \leqslant \frac{\upsilon}{3}$ and $o_0 \leqslant \frac{\upsilon^2}{9q}$ and $o_{k+1} = o_k \upsilon t_{k+1}^2$ where $\upsilon \in [0, 1)$ is any given constant and t_k is a non-negative sequence with $t_1 \leqslant 1$ and $t_{k+1} \leqslant t_k$ for all $k \in \mathbb{N}$. Then, for all $k \in \mathbb{N}$,

$$z_k \leqslant r_k, \quad r_{k+1} = \max\left\{r_k\rho_k, r_0\upsilon^{k+1}\prod_{i=0}^{k} t_{i+1}\right\}, \quad r_0 = \max\{z_0, \tfrac{3o_0}{\upsilon}\}, \quad \rho_k = qr_k + \tau_k + \tfrac{o_0}{r_0}\prod_{i=0}^{k-1} t_{i+1} \in [0, \upsilon]. \tag{3.46}$$

Therefore, $z_k \to 0$ at an R-linear rate. Furthermore, if $\lim_{k\to\infty} t_k = 0$ then $z_k \to 0$ at an R-superlinear rate.

Proof. Let $b_{k+1} = b_k t_{k+1}$ for all $k \in \mathbb{N}$ with $b_0 = \frac{o_0}{r_0} \leqslant \frac{\upsilon}{3}$. We have that $b_k \leqslant b_0 \leqslant \frac{\upsilon}{3}$ and $\frac{o_{k+1}}{b_{k+1}} = \frac{o_k}{b_k}\upsilon t_{k+1} \leqslant \frac{o_k}{b_k}$ for all $k \in \mathbb{N}$. We will use induction to prove that

$$z_k \leqslant r_k, \quad r_{k+1} = \max\left\{r_k\rho_k, \tfrac{o_{k+1}}{b_{k+1}}\right\}, \quad r_0 = \max\{z_0, \tfrac{3o_0}{\upsilon}\}, \quad \rho_k = qr_k + \tau_k + b_k \in [0, \upsilon]. \tag{3.47}$$

Note that the base case of $k = 0$ is trivially satisfied since $z_0 \leqslant r_0$. Suppose that this result is true for some k. From (3.45), we get

$$z_{k+1} \leqslant qz_k^2 + \tau_k z_k + o_k \leqslant r_k\left(qr_k + \tau_k + \frac{o_k}{r_k}\right) \leqslant r_k(qr_k + \tau_k + b_k) = r_k\rho_k \leqslant r_{k+1}.$$

Next, we will use induction to show that $r_k \leqslant \frac{\upsilon}{3q}$ and $\rho_k \leqslant \upsilon$ for all $k \in \mathbb{N}$. Note the base case of $k = 0$ is satisfied since $r_0 \leqslant \frac{\upsilon}{3q}$ and $\rho_0 = qr_0 + \tau_0 + b_0 \leqslant \upsilon$. Let us assume that this result is true for some k. Consider,

$$\begin{aligned} r_{k+1} &= \max\{r_k\rho_k, \tfrac{o_k}{b_k}\} \leqslant \max\{r_k, \tfrac{o_0}{b_0}\} \leqslant \tfrac{\upsilon}{3q} \\ \rho_{k+1} &= qr_{k+1} + \tau_{k+1} + b_k \leqslant \tfrac{\upsilon}{3} + \tau_0 + \tfrac{\upsilon}{3} \leqslant \upsilon. \end{aligned}$$

Therefore, from (3.47), we have that

$$\frac{r_{k+1}}{r_k} = \max\left\{\rho_k, \frac{o_{k+1}}{b_{k+1}r_k}\right\} \leqslant \max\left\{\rho_k, \frac{o_{k+1}b_k}{b_{k+1}o_k}\right\} = \max\left\{\rho_k, \upsilon t_{k+1}\right\} \leqslant \upsilon < 1.$$

Hence, $r_k \to 0$ at a Q-linear rate and consequently $z_k \to 0$ at an R-linear rate. Furthermore, if $\lim_{k\to\infty} t_k = 0$ and $\lim_{k\to\infty} \tau_k = 0$, then

$$\lim_{k\to\infty} \rho_k \leqslant \lim_{k\to\infty} qr_k + \tau_k + \tfrac{o_0}{r_0}t_1^k = 0. \tag{3.48}$$

Therefore,

$$\lim_{k\to\infty} \frac{r_{k+1}}{r_k} = \lim_{k\to\infty} \max\left\{\rho_k, \upsilon t_{k+1}\right\} = 0.$$

Hence, $r_k \to 0$ at a Q-superlinear rate and consequently $z_k \to 0$ at an R-superlinear rate. □

We are now ready to provide the main theoretical results in this section.

Theorem 3.10. *(Deterministic local linear and superlinear convergence). Suppose Assumptions 3.1, 3.3, 3.4, and 3.5 hold. For any $w_0 \in \mathbb{R}^d$, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by (2.1) where the Hessian approximation is given in (2.4) with $\tilde{\mu} \leqslant \frac{\mu}{2}$, the sample sets S_k chosen deterministically without replacement in a cyclic fashion such that $n|S_k| = N$ for some $n \in \mathbb{N}$ with $n \geqslant 1$, and $\gamma_i = \frac{1}{k+1}$ for all $i = 0, \cdots, k$. Furthermore, the gradient approximation g_k satisfies the deterministic norm condition (2.6a) with $A_k = \widetilde{H}_k^{-1}$ for all $k \in \mathbb{N}$, $\sum_{i=0}^{\infty} \iota_i = \tilde{\iota} < \infty$, and $\alpha_k = \alpha \leqslant \frac{\tilde{\mu}}{L}$ for all $k < k_{sup}$ for some $k_{sup} \in \mathbb{N}$. Furthermore, for all $k_{lin} \leqslant k < k_{sup}$, $\iota_{k+1} = \iota_k a_g$ for some $\iota_{k_{lin}} \geqslant 0$ and $a_g \in [0, 1)$, where k_{lin} is given in Lemma 3.6. Let $\tilde{\theta}_l = \frac{\sqrt{a_l}}{6}\left(\frac{\tilde{\mu}}{\tilde{L}}\right)^{3/2}$, $q = \frac{7M}{2\tilde{\mu}}$, and $\tau_k := \frac{1}{\tilde{\mu}}\left(\frac{nC_{p,d}+C_{s,d}(n-1)^2}{n(k+1)} + \theta_k L\sqrt{\frac{\tilde{L}}{\tilde{\mu}}}\right)$ for some $a_l \in [0, 1)$, and let r_k be a sequence such that $r_{k_{sup}} = \max\left\{\|w_{k_{sup}} - w^*\|, \frac{\sqrt{\tilde{L}\iota_{k_{sup}}}}{\tilde{\mu}}\right\} \leqslant \frac{\sqrt{a_l}}{3q}$.*

1. *If $\alpha_k = 1$, $\theta_k = \tilde{\theta}_l$, and $\iota_{k+1} = \iota_k a_l$ for all $k \geqslant k_{sup}$. Then*

$$\|w_k - w^*\| \leqslant r_k, \quad r_{k+1} = \max\left\{r_k\rho_k, r_{k_{sup}}(\sqrt{a_l})^{k-k_{sup}+1}\right\}, \quad \rho_k = qr_k + \tau_k + \frac{\sqrt{\tilde{L}\iota_{k_{sup}}}}{r_{k_{sup}}\tilde{\mu}} \in [0, \sqrt{a_l}]. \tag{3.49}$$

   Therefore, $\|w_k - w^*\| \to 0$ at an R-linear rate with rate constant upper bounded by $\sqrt{a_l} \in [0, 1)$.

2. *If $\alpha_k = 1$, $\theta_k = \frac{\tilde{\theta}_l}{k+1}$, and $\iota_{k+1} = \iota_k a_l t_{k+1}^4$, $t_k = \frac{1}{k}$ for all $k \geqslant k_{sup}$. Then*

$$\|w_k - w^*\| \leqslant r_k, \quad r_{k+1} = \max\left\{r_k\rho_k, r_{k_{sup}}(\sqrt{a_l})^{k-k_{sup}+1}\prod_{i=k_{sup}}^{k} t_{i+1}\right\},$$
$$\rho_k = qr_k + \tau_k + \frac{\sqrt{\tilde{L}\iota_{k_{sup}}}}{r_{k_{sup}}\tilde{\mu}}\prod_{i=k_{sup}}^{k-1} t_{i+1} = \mathcal{O}\left(\tfrac{1}{k}\right) \in [0, \sqrt{a_l}]. \tag{3.50}$$

   Therefore, $\|w_k - w^*\| \to 0$ at an R-superlinear rate with rate constant upper bounded by $\max\{\rho_k, \sqrt{a_l}t_{k+1}\} = \mathcal{O}\left(\frac{1}{k}\right)$.

Proof. Let

$$k_{\text{sup}} = \left\lceil \max\left\{4\left(1 + \frac{\hat{\lambda}}{\mu}\right)(k_{\text{lin}} + n - 1), \frac{4M\nu}{\mu^2(1-\sqrt{\tilde{\rho}_1})}, \frac{6(nC_{p,d}+C_{s,d}(n-1)^2)}{n\tilde{\mu}\sqrt{a_l}}, \right.\right.$$
$$\left.\left. k_{\text{lin}} + 2\log_{1/\tilde{\rho}_1}\left(\frac{3q\nu}{\mu\sqrt{a_l}}\right), k_{\text{lin}} + \log_{1/a_g}\left(\frac{81\tilde{L}q^2\iota_{k_{\text{lin}}}}{a_l^2\tilde{\mu}^2}\right)\right\}\right\rceil, \tag{3.51}$$

where k_{lin} is defined in (3.27), $\hat{\lambda}$ is defined in Lemma 3.7, $C_{p,d}, C_{s,d}$ are given in (3.32) and (3.44) respectively, and $\tilde{\rho}_1$ is the linear convergence rate defined in Theorem 3.2. We note that $k_{\text{sup}} \geqslant k_{\text{non}}$ due to the first and second terms in (3.51) where k_{non} is defined in (3.38). Using (3.15), (3.20), (3.26), and (3.41), we get for all $k \geqslant k_{\text{sup}}$,

$$\|w_{k+1} - w^*\| \leqslant \frac{7M}{2\tilde{\mu}}\|w_k - w^*\|^2 + \frac{1}{\tilde{\mu}}\left(\frac{nC_{p,d} + C_{s,d}(n-1)^2}{n(k+1)} + \theta_k L\sqrt{\frac{\tilde{L}}{\tilde{\mu}}}\right)\|w_k - w^*\| + \frac{\sqrt{\tilde{L}\iota_k}}{\tilde{\mu}}$$
$$= q\|w_k - w^*\|^2 + \tau_k\|w_k - w^*\| + o_k,$$

where $o_k := \frac{\sqrt{\tilde{L}\iota_k}}{\tilde{\mu}}$. Using the third term in (3.51), $L \leqslant \tilde{L}$, and $\theta_k = \tilde{\theta}_l = \frac{\sqrt{a_l}}{6}\left(\frac{\tilde{\mu}}{\tilde{L}}\right)^{3/2}$, we get $\tau_k \leqslant \frac{\sqrt{a_l}}{3}$ for $k = k_{\text{sup}}$. In addition, using the fourth term in (3.51) and (3.30), we get $\|w_k - w^*\| \leqslant \frac{\sqrt{a_l}}{3q}$ for $k = k_{\text{sup}}$. Moreover, $\iota_{k_{\text{sup}}} = \iota_{k_{\text{lin}}} a_g^{k_{\text{sup}}-k_{\text{lin}}} \leqslant \frac{a_l^2\tilde{\mu}^2}{81\tilde{L}q^2}$ due to the fifth term in (3.51) which implies that $o_k \leqslant \frac{a_l}{9q}$ for $k = k_{\text{sup}}$.

Therefore, starting with $k = k_{\text{sup}}$ and using Lemma 3.9 with $\upsilon = \sqrt{a_l}$ and $\iota_{k+1} = \iota_k a_l$ yields (3.49). Moreover, from (3.48), we have that $\frac{r_{k+1}}{r_k} \leqslant \sqrt{a_l} < 1$.

Similarly starting with $k = k_{\text{sup}}$ and using Lemma 3.9 with $\upsilon = \sqrt{a_l}$ and $\iota_{k+1} = \frac{\iota_k a_l}{(k+1)^4}$ yields (3.50). Moreover, from (3.48), we get

$$\begin{aligned}\frac{r_{k+1}}{r_k} \leqslant \max\{\rho_k, \sqrt{a_l}t_{k+1}\} &\leqslant \max\left\{qr_k + \tau_k + \frac{o_{k_{\text{sup}}}}{r_0}\prod_{i=k_{\text{sup}}}^{k-1} t_{i+1}, \frac{\sqrt{a_l}}{k+1}\right\} \\ &< \max\left\{qr_{k_{\text{sup}}}(\sqrt{a_l})^{k-k_{\text{sup}}} + \frac{\tau_{k_{\text{sup}}}(k_{\text{sup}}+1)}{k+1} + \frac{o_{k_{\text{sup}}}}{r_{k_{\text{sup}}}(k+1)}, \frac{\sqrt{a_l}}{k+1}\right\} = \mathcal{O}\left(\tfrac{1}{k}\right).\end{aligned}$$

□

Remark 3.6. We make the following remarks about this result.

- **Case:** $\theta_k = 0$, $\iota_k = 0$, $(t_k = 0)$ (exact gradient). When exact gradients are employed, using Lemma 3.9, we get deterministic Q-superlinear convergence where the rate constant is $\mathcal{O}\left(\frac{1}{k}\right)$. We note that this is an improvement over the final phase superlinear convergence results established in probability established for stochastic Hessian sampling with mean zero sub-exponential Hessian noise where the rate constant is $\mathcal{O}\left(\sqrt{\frac{\log k}{k}}\right)$ [46, 64].

- **Case:** $\theta_k \neq 0$, $\iota_k = 0$ (inexact adaptive gradient). When inexact gradients are employed where the gradient accuracies are chosen solely relative to the gradient norm itself, we get Q-linear and Q-superlinear convergence results based on the choice of the θ_k parameter.

- **Rate constant**. We note that the local linear convergence rate constant ($\sqrt{a_l} \in [0, 1)$) is a hyperparameter that doesn't depend on the problem characteristics. Therefore, this local linear convergence result is better than global linear convergence results that typically depend on the condition number of the problem. This result is similar to other local linear convergence results established in the literature [13, 75], although those results are established either in probability or in expectation as opposed to the deterministic result presented here.

- **Step size**. Two different step sizes are chosen in the the global phase ($\alpha_k = \frac{\tilde{\mu}}{L}$) and local superlinear phase ($\alpha_k = 1$). While such two phase approaches are common for Newton-type methods [13], these results can be unified using an inexact line search approach that employs inexact function evaluations where the unit step size is automatically selected in the second (local) phase with an appropriately modified Armijo sufficient decrease condition [16, 70].

We will now characterize the number of iterations required to transition from one convergent phase of the algorithm to the other. For the sake of simplicity and to make it possible to compare our results with other existing results in the literature, we will only consider global strongly convex functions (see Assumption 3.2). Therefore, the algorithm only encounters two phases: Global linear convergence and local linear or superlinear convergence depending on the choice of gradient accuracies as established in Theorem 3.10. It is possible to account for global sublinear convergence phase too by analyzing k_{lin} given in (3.27). However, for strongly convex functions $k_{\text{lin}} = 0$.

Corollary 3.11. *Suppose Assumption 3.2 and conditions of Theorem 3.10 hold where we choose* $a_g = 1-\frac{\mu\tilde{\mu}}{2L\tilde{L}} \in [0,1)$. *Then, the number of iterations required for the iterates to reach local linear or superlinear convergence phase is given as*

$$k_{sup} = \tilde{\mathcal{O}}\left(\max\left\{\tfrac{N}{|S_0|}(1+\tfrac{\hat{\lambda}}{\mu}), \kappa^2(1+\tfrac{M}{\mu^{3/2}})\right\}\right), \tag{3.52}$$

where $\hat{\lambda}$ *is such that* $\lambda_{\min}(\nabla^2 F_{S_i}(w_i)) \succeq -\hat{\lambda}$ *for all* $i \in \mathbb{N}$ *and* $\kappa \leqslant \frac{\tilde{L}}{\tilde{\mu}}$.

Proof. Since the function is strongly convex, we no longer require the iterates to enter the basin where $\|\nabla f(w_k)\|^2 \leqslant \nu^2$ for invoking local strong convex properties. Therefore, (3.31) is updated using (3.10) as

$$\sum_{i=0}^{k}\|w_i - w^*\| \leqslant \sqrt{\frac{2}{\mu}}\sqrt{(f(w_k)-f(w^*))} \leqslant \sqrt{\frac{2\tilde{C}_1}{\mu}}\sum_{i=0}^{k}(\sqrt{\tilde{\rho}_1})^i < \frac{\sqrt{2\tilde{C}_1}}{\sqrt{\mu}(1-\sqrt{\tilde{\rho}_1})}. \tag{3.53}$$

Using $k_{\text{lin}} = 0$, (3.53) is updated as $C_{p,d} := \frac{M\sqrt{2\tilde{C}_1}}{\sqrt{\mu}(1-\sqrt{\tilde{\rho}_1})}$. Using these update formulae, we get

$$\begin{aligned} k_{\text{sup}} = \Bigg\lceil \max\Bigg\{ & 4(1+\tfrac{\hat{\lambda}}{\mu})(n-1), \tfrac{4M\sqrt{2\tilde{C}_1}}{\mu^{3/2}(1-\sqrt{\tilde{\rho}_1})}, \tfrac{6(nC_{p,d}+C_{s,d}(n-1)^2)}{n\tilde{\mu}\sqrt{a_l}}, \\ & 2\log_{1/\tilde{\rho}_1}\left(\tfrac{3\sqrt{2}q\sqrt{\tilde{C}_1}}{\sqrt{\mu a_l}}\right), \log_{1/a_g}\left(\tfrac{81\tilde{L}q^2\iota_0}{a_l^2\tilde{\mu}^2}\right)\Bigg\}\Bigg\rceil, \end{aligned} \tag{3.54}$$

Now, consider the second term in (3.54), we note that

$$\tfrac{4M\sqrt{2\tilde{C}_1}}{\mu^{3/2}(1-\sqrt{\tilde{\rho}_1})} \leqslant \tfrac{8M\sqrt{2\tilde{C}_1}}{\mu^{3/2}(1-\tilde{\rho}_1)} \leqslant \tfrac{16\sqrt{2}\tilde{C}_1L^2M}{\mu^{7/2}} = \tilde{\mathcal{O}}(\tfrac{\kappa^2 M}{\mu^{3/2}}). \tag{3.55}$$

The third term in (3.54) is given as,

$$\tfrac{6(nC_{p,d}+C_{s,d}(n-1)^2)}{n\tilde{\mu}\sqrt{a_l}} = \tfrac{6M\sqrt{2\tilde{C}_1}}{\mu^{3/2}(1-\sqrt{\tilde{\rho}_1})\sqrt{a_l}} + \tfrac{12\sqrt{\beta_{1,H}L^2+\beta_{2,H}}(n-1)^2}{n\tilde{\mu}\sqrt{a_l}} = \tilde{\mathcal{O}}(\tfrac{\kappa^2 M}{\mu^{3/2}}). \tag{3.56}$$

Now considering the last two terms in (3.54) and using $a_g = 1-\frac{\mu\tilde{\mu}}{2L\tilde{L}}$, $\log(1-x) \approx -x$ for small $x \in (0,1)$, we have that

$$\max\left\{2\log_{1/\tilde{\rho}_1}\left(\tfrac{3\sqrt{2}q\sqrt{\tilde{C}_1}}{\sqrt{\mu a_l}}\right), \log_{1/a_g}\left(\tfrac{81\tilde{L}q^2\iota_0}{a_l^2\tilde{\mu}^2}\right)\right\} = \tfrac{2L\tilde{L}}{\mu\tilde{\mu}}\max\left\{2\log\left(\tfrac{3\sqrt{2}q\sqrt{\tilde{C}_1}}{\sqrt{\mu a_l}}\right), \log\left(\tfrac{81\tilde{L}q^2\iota_0}{a_l^2\tilde{\mu}^2}\right)\right\} = \tilde{\mathcal{O}}(\kappa^2). \tag{3.57}$$

Combining (3.55), (3.56), (3.57), and using $n = \frac{N}{|S_0|}$ yields the desired result. □

Remark 3.7. We note that in the case where the subsampled functions are convex, i.e. $\lambda_{\min}(\nabla^2 F_{S_i}(x_i)) \succeq 0$, we have $\hat{\lambda} = 0$. Therefore,

$$k_{\text{sup}} = \tilde{\mathcal{O}}\left(\max\left\{\tfrac{N}{|S_0|}, \kappa^2(1+\tfrac{M}{\mu^{3/2}})\right\}\right). \tag{3.58}$$

These transition phases are better than the final transition phases established for uniform weighted scheme when $N < \kappa^6$ and comparable to nonuniform weighted scheme when $N = \mathcal{O}(\kappa^2)$ and $\frac{M}{\mu^{3/2}}$ is small in [64]. Furthermore, one could speed up the transition to local phase by modifying Newton's method using proximal extra gradient methods, at additional per-iteration costs [46].

4 Stochastic Sampling Analysis

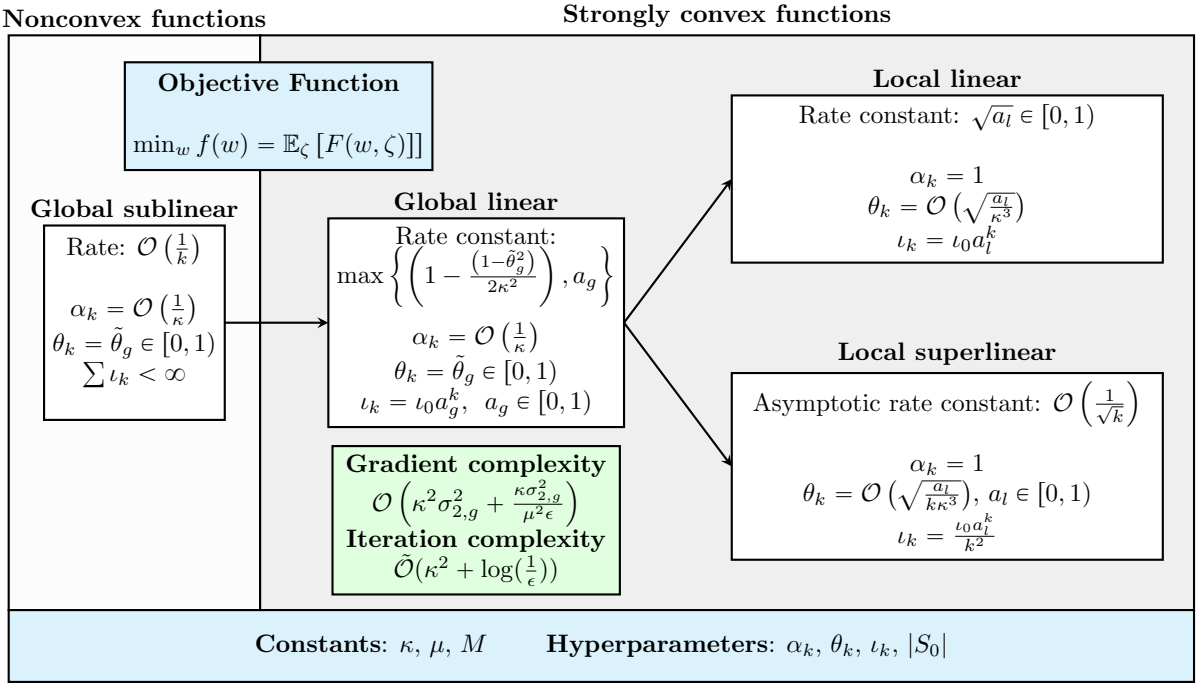

Figure 2: Overview of the results presented in this section. We characterize the main results for global and local convergence results, and their relationship to the problem constants and algorithmic hyperparameters. Here μ is the Hessian spectral lower bound, $\kappa = \frac{L}{\mu}$ is a condition-number like constant, and M is the Hessian Lipschitz constant. We note that there are additional global sublinear and linear results that allow any $0 < \tilde{\theta}_g < \infty$, but we do not annotate them in this figure for simplicity.

We continue our analysis by focusing on the stochastic sampling-based algorithms for the solution of the expectation problem (1.2). Our analysis mirrors the previous section and builds off of many of its assumption and derivations. A schematic for the analysis in this section is given in Figure 2. In addition to building on the results of the previous section, we also provide total number of gradients evaluated (gradient complexity) and Hessians computed to achieve an ϵ-accurate solution for the strongly convex functions.

We begin our analysis by providing the stochastic analog of Lemma 3.1, which establishes an upper bound on the difference between the objective function values at successive iterations, albeit in conditional expectation.

Lemma 4.1. *Suppose Assumption 3.1 holds. For any w_0, let $\{w_k : k \in \mathbb{Z}^+\}$ be iterates generated by (2.1) with the Hessian approximation given in (2.4). If the step size α_k at each iteration k is chosen such that $\alpha_k \leqslant \frac{\tilde{\mu}}{L}$, and if g_k is an unbiased estimator of $\nabla f(w_k)$. Then, for all $k \in \mathbb{Z}^+$, it follows that,*

$$\mathbb{E}_k[f(w_{k+1})] \leqslant f(w_k) - \alpha_k \left(1 - \frac{L\alpha_k}{2\tilde{\mu}}\right) \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{L\alpha_k^2}{2\tilde{\mu}} \mathbb{E}_k[\delta_k^T \widetilde{H}_k^{-1} \delta_k]. \tag{4.1}$$

Proof. Using (3.6), taking conditional expectation, and using $\mathbb{E}_k[g_k] = \nabla f(w_k)$, we get

$$\mathbb{E}_k[f(w_{k+1})] \leqslant f(w_k) - \alpha_k \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{L\alpha_k^2}{2} \mathbb{E}_k[(\widetilde{H}_k^{-1/2} \nabla f(w_k))^T \widetilde{H}_k^{-1} (\widetilde{H}_k^{-1/2} \nabla f(w_k)] + \frac{L\alpha_k^2}{2} \mathbb{E}_k[(\widetilde{H}_k^{-1/2} \delta_k)^T \widetilde{H}_k^{-1} (\widetilde{H}_k^{-1/2} \delta_k)].$$

Using the fact that $\widetilde{H}_k^{-1} \preceq \frac{1}{\tilde{\mu}} I$ in the above inequality yields (4.1). □

4.1 Global convergence

We denote the expectation with respect to all the random variables as $\mathbb{E}[\cdot]$. That is,

$$\mathbb{E}[f(w_k)] = \mathbb{E}_0 \mathbb{E}_1 \cdots \mathbb{E}_{k-1}[f(w_k)].$$

We proceed with the stochastic analog of Theorem 3.2 (in expectation).

Theorem 4.2. *(Global linear convergence). Suppose Assumptions 3.1 and 3.2 hold. For any $w_0 \in \mathbb{R}^d$, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by (2.1) where the Hessian approximation is given in (2.4) and the gradient approximations g_k satisfies the Condition 2.1 with $\iota_{k+1} = \iota_k a_g$ for some $\iota_0 > 0$ and $a_g \in [0, 1)$. Then, if g_k satisfies stochastic norm condition (2.6b) with*

1. *$A_k = \widetilde{H}_k^{-1}$, $\theta_k = \tilde{\theta}_g \in [0, 1)$ and $\alpha_k = \alpha \leqslant \frac{\tilde{\mu}}{L}$:*

$$\mathbb{E}[f(w_k) - f(w^*)] \leqslant \tilde{C}_1 \tilde{\rho}_1^k, \tag{4.2a}$$

$$\tilde{C}_1 := \max\left\{ f(w_0) - f(w^*), \frac{\tilde{L}\iota_0}{\mu\left(1 - \tilde{\theta}_g^2\right)} \right\}, \text{ and } \tilde{\rho}_1 := \max\left\{ 1 - \frac{\alpha\mu\left(1 - \tilde{\theta}_g^2\right)}{2\tilde{L}}, a_g \right\}.$$

2. *g_k being an unbiased estimator of $\nabla f(w_k)$, $\theta_k = \tilde{\theta}_g \geqslant 0$, and either of the following two conditions hold:*

   (a) $A_k = \widetilde{H}_k^{-1}$, $\alpha_k = \alpha \leqslant \frac{\tilde{\mu}}{L\left(1 + \tilde{\theta}_g^2\right)}$, (or)

   (b) $A_k = I$, $\alpha_k = \alpha \leqslant \frac{\tilde{\mu}}{L\left(1 + \frac{\tilde{\theta}_g^2 \tilde{L}}{\tilde{\mu}}\right)}$,

   then,

$$\mathbb{E}[f(w_k) - f(w^*)] \leqslant \tilde{C}_2 \tilde{\rho}_2^k, \tag{4.2b}$$

$$\tilde{C}_2 := \max\left\{ f(w_0) - f(w^*), \frac{L\tilde{L}\alpha\iota_0}{\mu\tilde{\mu}\min\{1, \tilde{\mu}\}} \right\}, \text{ and } \tilde{\rho}_2 := \max\left\{ 1 - \frac{\alpha\mu}{2\tilde{L}}, a_g \right\}.$$

Proof. Case (1) From (2.6b) and (3.5), we have

$$\begin{aligned}
\mathbb{E}_k[f(w_{k+1})] &\leqslant f(w_k) - \frac{\alpha_k}{2} \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{\alpha_k}{2} \mathbb{E}_k\left[\mathbb{E}\left[\delta_k^T \widetilde{H}_k^{-1} \delta_k | w_k, \widetilde{H}_k^{-1}\right]\right] \\
&\leqslant f(w_k) - \frac{\alpha_k}{2} \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{\alpha_k \theta_k^2}{2} \mathbb{E}_k[\nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k)] + \frac{\alpha_k \iota_k}{2} \\
&\leqslant f(w_k) - \frac{\alpha_k(1 - \theta_k^2)}{2\tilde{L}} \|\nabla f(w_k)\|^2 + \frac{\alpha_k \iota_k}{2}.
\end{aligned} \tag{4.3a}$$

Case (2) From (2.6b) and (4.1), and using condition (a), we have

$$\begin{aligned}\mathbb{E}_k[f(w_{k+1})] &\leqslant f(w_k) - \alpha_k \left(1 - \frac{L\alpha_k}{2\tilde{\mu}}(1+\theta_k^2)\right) \nabla f(w_k)^T \widetilde{H}_k^{-1} \nabla f(w_k) + \frac{L\alpha_k^2 \iota_k}{2\tilde{\mu}} \\ &\leqslant f(w_k) - \frac{\alpha_k}{2\tilde{L}} \|\nabla f(w_k)\|^2 + \frac{L\alpha_k^2 \iota_k}{2\tilde{\mu}}.\end{aligned}$$

Similarly, from (2.6b) and (4.1), and using condition (b), we have

$$\begin{aligned}\mathbb{E}_k[f(w_{k+1})] &\leqslant f(w_k) - \frac{\alpha_k}{\tilde{L}} \left(1 - \frac{L\alpha_k}{2\tilde{\mu}}\right) \|\nabla f(w_k)\|^2 + \frac{L\alpha_k^2}{2\tilde{\mu}^2} \mathbb{E}_k[\|\delta_k\|^2] \\ &\leqslant f(w_k) - \frac{\alpha_k}{\tilde{L}} \left(1 - \frac{L\alpha_k}{2\tilde{\mu}} \left(1 + \tfrac{\theta^2 \tilde{L}}{\tilde{\mu}}\right)\right) \|\nabla f(w_k)\|^2 + \frac{L\alpha_k^2 \iota_k}{2\tilde{\mu}^2} \\ &\leqslant f(w_k) - \frac{\alpha_k}{2\tilde{L}} \|\nabla f(w_k)\|^2 + \frac{L\alpha_k^2 \iota_k}{2\tilde{\mu}^2}.\end{aligned}$$

Combining the above two results yield

$$\mathbb{E}_k[f(w_{k+1})] \leqslant f(w_k) - \frac{\alpha_k}{2\tilde{L}} \|\nabla f(w_k)\|^2 + \frac{L\alpha_k^2 \iota_k}{2\tilde{\mu} \min\{1, \tilde{\mu}\}}. \tag{4.3b}$$

The rest of the proof to attain (4.2a) and (4.2b) follows by using similar arguments as in the deterministic norm condition analysis give in the proof of Theorem 3.2. □

As was discussed in Remark 3.1, we note that the rate constant $\left(1 - \frac{\alpha\mu}{2\tilde{L}}\right)$ being worse than that of steepest descent is an artifact of the analysis and not essentially algorithmic in nature. We do not observe deteriorated global convergence in our numerical experiments. Additionally we do not consider the biased stochastic norm condition with $A_k = I$, as it leads to restrictive choices of $\tilde{\theta}_g$.

Theorem 4.3. *(Global sublinear convergence, stochastic case). Suppose Assumption 3.1 holds and the objective function f is bounded below by f_{min}. For any w_0, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by* (2.1) *where the gradient approximations g_k satisfy the Condition 2.1with $\sum_{i=0}^{\infty} \iota_k = \tilde{\iota} < \infty$. Then, for any positive integer T, if g_k satisfies stochastic norm condition* (2.6b) *with*

1. *$A_k = \widetilde{H}_k^{-1}$, $\theta_k = \tilde{\theta}_g \in [0, 1)$ and $\alpha_k = \alpha \leqslant \frac{\tilde{\mu}}{\tilde{L}}$:*

$$\min_{0 \leqslant k \leqslant T-1} \mathbb{E}[\|\nabla f(w_k)\|^2] \leqslant \frac{\tilde{L}}{(1 - \tilde{\theta}_g^2)T} \left(\frac{2(f(w_0) - f_{min})}{\alpha} + \tilde{\iota}\right). \tag{4.4a}$$

2. *g_k being an unbiased estimator of $\nabla f(w_k)$, $\theta_k = \tilde{\theta}_g \geqslant 0$, and either of the following two conditions hold:*

   (a) $A_k = \widetilde{H}_k^{-1}$, $\alpha_k = \alpha \leqslant \frac{\tilde{\mu}}{L(1+\tilde{\theta}_g^2)}$, (or)

   (b) $A_k = I$, $\alpha_k = \alpha \leqslant \frac{\tilde{\mu}}{L\left(1 + \frac{\tilde{\theta}_g^2 \tilde{L}}{\tilde{\mu}}\right)}$,

   then,

$$\min_{0 \leqslant k \leqslant T-1} \mathbb{E}[\|\nabla f(w_k)\|^2] \leqslant \frac{\tilde{L}}{T} \left(\frac{2(f(w_0) - f_{min})}{\alpha} + \frac{L\alpha\tilde{\iota}}{\tilde{\mu} \min\{1, \tilde{\mu}\}}\right). \tag{4.4b}$$

Moreover, $\sum_{k=0}^{\infty} \|\nabla f(w_k)\|^2 < \infty$ *almost surely and consequently* $\|\nabla f(w_k)\|^2 \to 0$ *as* $k \to \infty$ *almost surely.*

Proof. Starting with inequalities (4.3a) and (4.3b) and following the same procedure as in the deterministic case, albeit in expectation, yields (4.4a) and (4.4b) respectively. Furthermore, applying Robbins-Siegmund Theorem [74] to (4.3a) or (4.3b) yields $\sum_{k=0}^{\infty} \|\nabla f(w_k)\|^2 < \infty$ almost surely. Using this inequality, it is not difficult to show that $\|\nabla f(w_k)\|^2 \to 0$ as $k \to \infty$ almost surely. □

4.2 Local convergence

We now provide local superlinear rates of convergence results for the iterates generated by (2.1) when unit step size is eventually employed. We begin by extending Lemma 3.4 to the stochastic setting.

Lemma 4.4. *Suppose Assumptions 3.1 and 3.3 hold. For any* $w_0 \in \mathbb{R}^d$*, let* $\{w_k : k \in \mathbb{N}\}$ *be iterates generated by* (2.1) *where the Hessian approximation is given in* (2.4) *and the gradient approximations* g_k *satisfies the Condition 2.1 with* $\lambda_{\min}(A_k) \geqslant \lambda_A$ *and* $\frac{\lambda_{\max}(A_k)}{\lambda_{\min}(A_k)} \leqslant \kappa_A$ *for some positive constants* $\lambda_A, \kappa_A < \infty$*. If at any iteration* $k \in \mathbb{N}$*, unit step size is chosen* ($\alpha_k = 1$)*, then, if* g_k *satisfies stochastic norm condition* (2.6b)

$$\begin{aligned}\mathbb{E}\left[\|w_{k+1} - w^*\|\right] \leqslant \frac{M}{2\tilde{\mu}}\mathbb{E}\left[\|w_k - w^*\|^2\right] + \frac{1}{\tilde{\mu}}\mathbb{E}\left[\|(\widetilde{H}_k - \nabla^2 f(w_k))(w_k - w^*)\|\right] \\ + \frac{L\sqrt{\kappa_A}}{\tilde{\mu}}\theta_k \mathbb{E}\left[\|w_k - w^*\|\right] + \frac{\sqrt{\iota_k}}{\tilde{\mu}\sqrt{\lambda_A}},\end{aligned} \tag{4.5}$$

where w^* *is an optimal solution.*

Proof. We use a similar approach to the proof of Lemma 3.4, see the proof there for the decomposition of errors. When g_k satisfies stochastic norm condition (2.6b), taking conditional expectations of the gradient term in (3.16) and using Jensen's inequality $\mathbb{E}[\|g_k - \nabla f(w_k)\|_{A_k}|w_k, A_k] \leqslant (\mathbb{E}[\|g_k - \nabla f(w_k)\|_{A_k}^2|w_k, A_k])^{1/2}$ followed by full expectation yields (4.5). □

Lemma 4.4 establishes the dependence of the iterate distance to optimality on the Hessian approximation error. Recall that Lemma 3.5 decomposes the error into several terms, one of which includes the sum of errors in the difference between subsampled Hessian at each iterate, and the subsampled Hessian at the optimum, $\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*)$. In the next lemma, we establish upper bounds for the terms in Lemma 3.5, similar to the bounds derived in the deterministic sampling case as was done in Lemma 3.6.

Lemma 4.5. *Suppose Assumptions 3.1, 3.3 and 3.4 hold. For any* $w_0 \in \mathbb{R}^d$*, let* $\{w_k : k \in \mathbb{N}\}$ *be iterates generated by* (2.1) *where the Hessian approximation is given in* (2.4) *with* $\gamma_i = \frac{1}{k+1}$ *for all* $i = 0, \cdots, k$*, and the gradient approximations* g_k *satisfies the Condition 2.1 with* $\sum_{i=0}^{\infty} \iota_k = \tilde{\iota} < \infty$*. If* A_k *and the corresponding step size* α_k *are chosen according to Theorem 4.3. Then there exists* $k_{lin} \geqslant 0$ *such that for any* $k \geqslant k_{lin}$ *if* $\iota_{k+1} = \iota_k a_g$ *for some* $\iota_{k_{lin}} \geqslant 0$ *and* $a_g \in [0, 1)$*, we have that if* g_k *satisfies stochastic norm condition* (2.6b) *with any of the choices for* $(A_k, \theta_k, \alpha_k)$ *provided in Theorem 4.3. In addition, suppose either* $\sum_{i=0}^{\infty} \|\nabla f(w_i)\|^2 < \infty$ *or Assumption 3.2 holds, then there exists a constant* $C_{p,s}$ *such that*

$$\sum_{i=0}^{k} \gamma_i \mathbb{E}\left[\left\|\left(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*)\right)(w_k - w^*)\right\|\right] \leqslant \frac{C_{p,s}}{k+1}\left(\mathbb{E}\left[\|w_k - w^*\|^2\right]\right)^{1/2}. \tag{4.6}$$

Proof. In the stochastic setting, we assume that $\sum_{i=0}^{\infty} \|\nabla f(w_i)\|^2 < \infty$ or Assumption 3.2 holds. If $\sum_{i=0}^{\infty} \|\nabla f(w_i)\|^2 < \infty$ then there exists a $k_{\text{lin}} \in \mathbb{N}$ such that for all $k \geqslant k_{\text{lin}}$, $\|\nabla f(w_k)\| \leqslant \nu$. Therefore,

for all $k \geqslant k_{\text{lin}}$ the iterates are all in locally strongly convex regime. Taking expectations in (3.29) we get,

$$\begin{aligned}
&\sum_{i=0}^{k} \gamma_i \mathbb{E}\left[\left\|\left(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*)\right) e_k\right\|\right] \\
&\quad \leqslant \frac{2Lk_{\text{lin}}}{k+1}\mathbb{E}\left[\|w_k - w^*\|\right] + \frac{M}{k+1}\sum_{i=k_{\text{lin}}}^{k} \mathbb{E}\left[\|w_i - w^*\|\|w_k - w^*\|\right] \\
&\quad \leqslant \frac{2Lk_{\text{lin}}}{k+1}\left(\mathbb{E}\left[\|w_k - w^*\|^2\right]\right)^{1/2} + \frac{M}{k+1}\sum_{i=k_{\text{lin}}}^{k}\left(\mathbb{E}\left[\|w_i - w^*\|^2\right]\right)^{1/2}\left(\mathbb{E}\left[\|w_k - w^*\|^2\right]\right)^{1/2}, \qquad (4.7)
\end{aligned}$$

where the last inequality is due to the fact that $(\mathbb{E}[a])^2 \leqslant \mathbb{E}[a^2]$ and $(\mathbb{E}[ab])^2 \leqslant \mathbb{E}[a^2]\mathbb{E}[b^2]$ for any $a, b > 0$. Now using the global convergence results given in Theorem 4.3 and following the similar approach as in the deterministic norm condition analysis, albeit in expectation, we get, for an appropriate choice of $\iota_{k_{\text{lin}}}$ that $\|\nabla f(w_k)\|^2 \leqslant \nu^2$ for $k \geqslant k_{\text{lin}}$. Furthermore, from Jensen's inequality, we also have

$$\left(\mathbb{E}\left[\|w_k - w^*\|\right]\right)^2 \leqslant \mathbb{E}\left[\|w_k - w^*\|^2\right] \leqslant \tfrac{\nu^2 \tilde{\rho}^{k-k_{\text{lin}}}}{\mu^2}, \qquad (4.8)$$

and

$$\sum_{i=k_{\text{lin}}}^{k}\left(\mathbb{E}\left[\|w_i - w^*\|^2\right]\right)^{1/2} < \frac{\nu}{\mu(1-\sqrt{\tilde{\rho}})}, \qquad (4.9)$$

where $\tilde{\rho}$ is the rate constant that depends on the choice of $(A_k, \theta_k, \alpha_k)$ given in Theorem 4.2. Substituting (4.9) in (4.7) yields

$$\begin{aligned}
\sum_{i=0}^{k} \gamma_i \mathbb{E}\left[\left\|\left(\nabla^2 F_{S_i}(w_i) - \nabla^2 F_{S_i}(w^*)\right) e_k\right\|\right] &\leqslant \frac{1}{k+1}\left(2Lk_{\text{lin}} + \frac{\nu M}{\mu(1-\sqrt{\tilde{\rho}})}\right)\left(\mathbb{E}\left[\|w_k - w^*\|^2\right]\right)^{1/2} \\
&= \frac{C_{p,s}}{k+1}\left(\mathbb{E}\left[\|w_k - w^*\|^2\right]\right)^{1/2},
\end{aligned}$$

where

$$C_{p,s} := 2Lk_{\text{lin}} + \frac{\nu M}{\mu(1-\sqrt{\tilde{\rho}})}. \qquad (4.10)$$

□

Remark 4.1. If the functions are globally strongly convex (Assumption 3.2 holds) then there is no sublinear convergent phase in the algorithm. That is, $k_{\text{lin}} = 0$ in this setting. Furthermore, we made the assumption that $\sum_{i=0}^{\infty} \|\nabla^2 f(w_i)\|^2 < \infty$. This assumption, although made on the iterates which are stochastic, is necessary to ensure that the iterates eventually lie within a locally strongly convex regime. Moreover, from Theorem 4.3, we have that $\sum_{i=0}^{\infty} \|\nabla^2 f(w_i)\|^2 < \infty$ almost surely, making this assumption relatively weak in this setting.

We will now establish that the nonconvex error term in Lemma 3.5 vanishes for sufficiently large number of iterations. To achieve this, for the expectation problem, we need an additional assumption about the subsampled functions when the iterates enter the strongly convex regime.

Assumption 4.1. *In the expectation problem* (1.2)*, for all $k \geqslant k_{lin}$, where k_{lin} is defined in Lemma 4.5,* $\nabla^2 F_{S_k}(w_k) \succeq \mu I$.

Remark 4.2. Assumption 4.1 implies that when the iterates enter the locally strongly convex regime, the subsampled functions are also strongly convex. In the case when the objective function is strongly convex, this assumption is typically satisfied due to regularziation in machine learning problems. Moreover, such assumptions have been previously made in [6, 13] and are required to establish strong convergence results in expectation.

Lemma 4.6. *Suppose conditions of Lemma 4.5 hold. Let* $\tilde{\mu} \leqslant \frac{\mu}{2}$*, and* g_k *satisfies stochastic norm condition* (2.6b) *and suppose that Assumption 4.1 holds. Then, there exists* $k_{non} \geqslant k_{lin}$ *such that for all* $k \geqslant k_{non}$*,* $\widetilde{H}_k = \widehat{H}_k$*.*

Proof. Let

$$k_{\text{non}} = 2k_{\text{lin}} \left(1 + \tfrac{\hat{\lambda}}{\mu}\right). \tag{4.11}$$

Then, from (3.34), for all $k \geqslant k_{\text{non}}$, we have

$$v^T \widehat{H}_k v \geqslant \frac{-\hat{\lambda} k_{\text{lin}}}{k+1}\|v\|^2 + \frac{\mu(k+1-k_{\text{lin}})}{k+1}\|v\|^2 \geqslant \frac{\mu}{2}\|v\|^2 \geqslant \tilde{\mu}\|v\|^2,$$

where we used $\tilde{\mu} \leqslant \frac{\mu}{2}$. Therefore, $\lambda_{\min}(\widehat{H}_k) \geqslant \tilde{\mu}$ and $\widetilde{H}_k = \widehat{H}_k$ for all $k \geqslant k_{\text{non}}$. □

To analyze the last term in Lemma 3.5, we make the following standard assumption about individual (stochastic) Hessian components.

Assumption 4.2. *(Hessian approximations, stochastic case). The individual component Hessians are bounded relative to the Hessian of the objective function* f *at the optimal solution* w^**. That is, for the expectation problem, there exists constant* $\sigma_H^2 \geqslant 0$ *such that*

$$\mathbb{E}_\zeta \left[\left\| \nabla^2 F(w^*, \zeta) - \nabla^2 f(w^*) \right\|^2 \right] \leqslant \sigma_H^2. \tag{4.12a}$$

Remark 4.3. As with Assumption 3.5, we note that Assumption 4.2 is relatively weak compared to similar assumptions made in the literature [13], as it only requires bounded variance at the optimum, instead at all possible iterates $w \in \mathbb{R}^d$. In addition, we also note that this assumption is trivially satisfied with the bound $\sigma_H = 2L$ when the functions are double differentiable with Lipschitz continuous gradients.

Lemma 4.7. *Suppose Assumptions 3.3 and 4.2 hold. If* $\gamma_i = \frac{1}{k+1}$ *for all* $i = 0, \cdots, k$*, and the sample sets* S_k *are randomly chosen such that* $|S_k| = |S_0| \in \mathbb{N}$*. Then*

$$\mathbb{E}\left[\left\| \left(\sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \right)(w_k - w^*) \right\| \right] \leqslant \frac{\sigma_H}{\sqrt{(k+1)|S_0|}} (\mathbb{E}[\|w_k - w^*\|^2])^{1/2}. \tag{4.13}$$

Proof. Consider,

$$\begin{aligned} &\mathbb{E}\left[\left\| \sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*)(w_k - w^*) \right\| \right] \\ &\quad \leqslant \left(\mathbb{E}\left[\left\| \sum_{i=0}^{k} \gamma_i \nabla^2 F_{S_i}(w^*) - \nabla^2 f(w^*) \right\|^2 \right] \right)^{1/2} \left(\mathbb{E}\left[\|w_k - w^*\|^2 \right] \right)^{1/2}. \end{aligned} \tag{4.14}$$

where the inequality is due to $(\mathbb{E}[ab])^2 \leqslant \mathbb{E}[a^2]\mathbb{E}[b^2]$ for any $a, b > 0$. Let $S^\dagger = \cup_{i=0}^{k} S_i$ with $|S^\dagger| = (k+1)|S_0|$. Using $\nabla^2 F_{S^\dagger}(w^*) = \frac{1}{(k+1)|S_0|}\sum_{i\in S^\dagger}\nabla^2 F_i(w^*)$, we get

$$\begin{aligned}\mathbb{E}\left[\left\|\nabla^2 F_{S^\dagger}(w^*) - \nabla^2 f(w^*)\right\|^2\right] &= \frac{1}{(k+1)^2|S_0|^2}\mathbb{E}\left[\left\|\sum_{i\in S^\dagger}(\nabla^2 F_i(w^*) - \nabla^2 f(w^*))\right\|^2\right] \\ &= \frac{1}{(k+1)^2|S_0|^2}\sum_{i\in S^\dagger}\mathbb{E}\left[\left\|\nabla^2 F_i(w^*) - \nabla^2 f(w^*)\right\|^2\right] \\ &= \frac{1}{(k+1)|S_0|}\mathbb{E}_\zeta\left[\left\|\nabla^2 F(w^*,\zeta) - \nabla^2 f(w^*)\right\|^2\right] \leqslant \frac{\sigma_H^2}{(k+1)|S_0|}, \qquad (4.15)\end{aligned}$$

where the second and third equalities are due to the fact that the sample sets S_k's consists of i.i.d samples of ζ, and the inequality is due to Assumption 3.5. Substituting (4.15) in (4.14) yields the desired result. □

Remark 4.4. Lemma 4.7 establishes the bound on the sampling error in terms of the sample size $|S_0|$ and the iteration number k. We note that deterministic sampling without replacement in a cyclic manner has a better dependence on k as compared to the stochastic subsampled Hessian (see Lemma 3.8).

To prove similar results for the expectation problem, we need an additional assumption on the second moments of the iterates as employed in stochastic second-order methods [7, 13].

Assumption 4.3. *There exists a non-negative constant η such that for all $k \in \mathbb{Z}^+$,*

$$\mathbb{E}\left[\|w_k - w^*\|^2\right] \leqslant \eta\left(\mathbb{E}[\|w_k - w^*\|]\right)^2.$$

Remark 4.5. Although this assumption seems to be restrictive, it is imposed on non-negative numbers and is less restrictive than assuming that the iterates are bounded. It might be stronger than the sub-exponential assumption on the stochastic Hessian [46, 64], it is however required to establish results in expectation instead of results in probability. This assumption has been employed in other works to the same effect [7, 13].

Finally we provide linear and superlinear local convergence rates for the expectation problem. This theorem is the stochastic analog of Theorem 3.10, and the proof is similar. We provide the entire proof for completeness.

Theorem 4.8. *(Expectation local linear and superlinear convergence). Suppose Assumptions 3.1, 3.3, 3.4, 4.1, 4.2, and 4.3 hold. For any $w_0 \in \mathbb{R}^d$, let $\{w_k : k \in \mathbb{N}\}$ be iterates generated by (2.1) where the Hessian approximation is given in (2.4) with $\tilde{\mu} \leqslant \frac{\mu}{2}$, the sample sets S_k are randomly chosen such that $|S_k| = |S_0|$ for all $k \in \mathbb{N}$, and $\gamma_i = \frac{1}{k+1}$ for all $i = 0, \cdots, k$. Furthermore, the gradient approximations g_k satisfies stochastic norm condition (2.6b) with $\sum_{i=0}^{\infty}\iota_i = \tilde{\iota} < \infty$ and any of the choices for $(A_k, \theta_k, \alpha_k)$ with $\frac{\lambda_{\max}(A_k)}{\lambda_{\min}(A_k)} \leqslant \kappa_A > 0$ provided in Theorem 4.3 for all $k < k_{sup}$ for some $k_{sup} \in \mathbb{N}$. In addition, suppose either $\sum_{i=0}^{\infty}\|\nabla f(w_i)\|^2 < \infty$ or Assumption 3.2 holds. Furthermore, for all $k_{lin} \leqslant k < k_{sup}$, $\iota_{k+1} = \iota_k a_g$ for some $\iota_{k_{lin}} \geqslant 0$ and $a_g \in [0,1)$, where k_{lin} is given in Lemma 4.5. Let $\tilde{\theta}_l = \frac{\sqrt{a_l}\tilde{\mu}}{9\sqrt{\kappa_A}L}$, $q = \frac{7M\eta}{2\tilde{\mu}}$, and $\tau_k := \frac{1}{\tilde{\mu}}\left(\frac{C_{p,s}\sqrt{\eta}}{k+1} + \frac{\sigma_H\sqrt{\eta}}{\sqrt{(k+1)|S_0|}} + \theta_k L\sqrt{\kappa_A}\right)$ for some $a_l \in [0,1)$, and let r_k be a sequence with and $r_{k_{sup}} = \max\left\{\mathbb{E}[\|w_{k_{sup}} - w^*\|], \frac{3\sqrt{\iota_{k_{sup}}}}{\tilde{\mu}\sqrt{\lambda_A a_l}}\right\} \leqslant \frac{\sqrt{a_l}}{3q}$.*

1. *If* $\alpha_k = 1$, $\theta_k = \tilde{\theta}_l$, *and* $\iota_{k+1} = \iota_k a_l$ *for all* $k \geqslant k_{sup}$. *Then*

$$\mathbb{E}[\|w_k - w^*\|] \leqslant r_k, \quad r_{k+1} = \max\left\{r_k\rho_k, r_{k_{sup}}(\sqrt{a_l})^{k-k_{sup}+1}\right\}, \quad \rho_k = qr_k + \tau_k + \frac{\sqrt{\iota_{k_{sup}}}}{\tilde{\mu}\sqrt{\lambda_A}r_{k_{sup}}} \in [0, \sqrt{a_l}]. \tag{4.16}$$

Therefore, $\mathbb{E}[\|w_k - w^*\|] \to 0$ *at an R-linear rate with rate constant upper bounded by* $\sqrt{a_l} \in [0, 1)$.

2. *If* $\alpha_k = 1$, $\theta_k = \frac{\tilde{\theta}_l}{\sqrt{k+1}}$, *and* $\iota_{k+1} = \iota_k a_l t_{k+1}^4$, $t_k = \frac{1}{\sqrt{k}}$ *for all* $k \geqslant k_{sup}$. *Then*

$$\mathbb{E}[\|w_k - w^*\|] \leqslant r_k, \quad r_{k+1} = \max\left\{r_k\rho_k, r_{k_{sup}}(\sqrt{a_l})^{k-k_{sup}+1}\prod_{i=k_{sup}}^{k} t_{i+1}\right\},$$
$$\rho_k = qr_k + \tau_k + \frac{\sqrt{\iota_{k_{sup}}}}{\tilde{\mu}\sqrt{\lambda_A}r_{k_{sup}}}\prod_{i=k_{sup}}^{k-1} t_{i+1} = \mathcal{O}(\tfrac{1}{\sqrt{k}}) \in [0, \sqrt{a_l}]. \tag{4.17}$$

Therefore, $\mathbb{E}[\|w_k - w^*\|] \to 0$ *at an R-superlinear rate with rate constant upper bounded by* $\max\{\rho_k, \sqrt{a_l}t_{k+1}\} = \mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$.

Proof. Let

$$k_{\text{sup}} = \left\lceil \max\left\{2k_{\text{lin}}\left(1 + \frac{\hat{\lambda}}{\mu}\right), \frac{9C_{p,s}\sqrt{\eta}}{\tilde{\mu}\sqrt{a_l}}, \frac{81\sigma_H^2\eta}{\tilde{\mu}^2|S_0|a_l}, k_{\text{lin}} + 2\log_{1/\tilde{\rho}}\left(\frac{3q\nu}{\mu\sqrt{a_l}}\right), k_{\text{lin}} + \log_{1/a_g}\left(\frac{81q^2\iota_{k_{\text{lin}}}}{a_l^2\tilde{\mu}^2\lambda_A}\right)\right\}\right\rceil, \tag{4.18}$$

where k_{lin} is defined in Lemma 4.5, $\hat{\lambda}$ is defined in Lemma 3.7, $C_{p,s}$ is defined in (4.10), and $\tilde{\rho}$ is the linear convergence rate defined in Theorem 4.2. We note that $k_{\text{sup}} \geqslant k_{\text{non}}$ due to the first term in (4.18) where k_{non} is defined in (4.11). Using (4.5), (3.20), (4.6), and (4.13), and Assumption 4.3 we get for all $k \geqslant k_{\text{sup}}$,

$$\begin{aligned}\mathbb{E}[\|w_{k+1} - w^*\|] &\leqslant \tfrac{7M\eta}{2\tilde{\mu}}(\mathbb{E}[\|w_k - w^*\|])^2 + \tfrac{1}{\tilde{\mu}}\left(\frac{C_{p,s}\sqrt{\eta}}{k+1} + \frac{\sigma_H\sqrt{\eta}}{\sqrt{(k+1)|S_0|}} + \theta_k L\sqrt{\kappa_A}\right)\mathbb{E}[\|w_k - w^*\|] \\ &\quad + \tfrac{\sqrt{\iota_k}}{\tilde{\mu}\sqrt{\lambda_A}} \\ &= q\|w_k - w^*\|^2 + \tau_k\|w_k - w^*\| + o_k,\end{aligned} \tag{4.19}$$

where $o_k := \frac{\sqrt{\iota_k}}{\tilde{\mu}\sqrt{\lambda_A}}$. Using the second and third terms in (4.18) and $\theta_k = \tilde{\theta}_l = \frac{\sqrt{a_l}\tilde{\mu}}{9\sqrt{\kappa_A}L}$, we get $\tau_k \leqslant \frac{\sqrt{a_l}}{3}$ for $k = k_{\text{sup}}$. In addition, using the fourth term in (4.18) and (4.8), we get $\mathbb{E}[\|w_k - w^*\|] \leqslant \frac{\sqrt{a_l}}{3q}$ for $k = k_{\text{sup}}$. Moreover, $\iota_{k_{\text{sup}}} = \iota_{k_{\text{lin}}} a_g^{k_{\text{sup}}-k_{\text{lin}}} \leqslant \frac{a_l^2\tilde{\mu}^2\lambda_A}{81q^2}$ due to the fifth term in (3.51) which implies that $o_k \leqslant \frac{a_l}{9q}$ for $k = k_{\text{sup}}$.

Therefore, starting with $k = k_{\text{sup}}$ and using Lemma 3.9 with $\upsilon = \sqrt{a_l}$ and $\iota_{k+1} = \iota_k a_l$ yields (4.16). Moreover, from (3.48), we have that $\frac{r_{k+1}}{r_k} \leqslant \sqrt{a_l} < 1$.

Similarly starting with $k = k_{\text{sup}}$ and using Lemma 3.9 with $\upsilon = \sqrt{a_l}$ and $\iota_{k+1} = \frac{\iota_k a_l}{(k+1)^2}$ yields (4.17). Moreover, from (3.48), we get

$$\begin{aligned}\tfrac{r_{k+1}}{r_k} \leqslant \max\{\rho_k, \sqrt{a_l}t_{k+1}\} &\leqslant \max\left\{qr_k + \tau_k + \tfrac{o_{k_{\text{sup}}}}{r_0}\prod_{i=k_{\text{sup}}}^{k-1} t_{i+1}, \sqrt{\tfrac{a_l}{k+1}}\right\} \\ &< \max\left\{qr_{k_{\text{sup}}}(\sqrt{a_l})^{k-k_{\text{sup}}} + \tau_k + \tfrac{o_{k_{\text{sup}}}}{r_{k_{\text{sup}}}\sqrt{k+1}}, \sqrt{\tfrac{a_l}{k+1}}\right\} = \mathcal{O}(\tfrac{1}{\sqrt{k}}).\end{aligned}$$

□

Remark 4.6. We note that unlike the deterministic Hessian sampling, stochastic sampling leads to a slower rate of superlinear convergence $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ similar to the final phase results established in [46, 64]. Moreover, for the exact gradient settings, although we only presented the final superlinear convergence results, from (4.19), it can be seen that there are two phases in the superlinearly convergent regime, where initially rate is $\mathcal{O}\left(\frac{1}{k}\right)$ and the later (final) phase is $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$, similar to results in [46, 64], albeit in expectation instead of high probability. We also note that the proof techniques employed in analyzing the Hessian approximations in this analysis are different from those established in [46, 64] where we analyze the variance in the Hessian sampling error at the optimal solution.

4.3 Complexity Analysis

In this section, we establish the iteration, gradient, and Hessian computational complexity bounds, i.e., the total number of iterations, individual gradient evaluations and Hessian evaluations required to get an ϵ-accurate solution, characterized as any iterate w_k satisfying

$$\left(\mathbb{E}[\|w_k - w^*\|]\right)^2 \leqslant \epsilon, \tag{4.20}$$

where w^* is an optimal solution. For the sake of simplicity, we will only consider global strongly convex functions (see Assumption 3.2). The global sublinear convergence results for general nonconvex functions established in Theorem 4.3 are required only to establish that the iterates will eventually enter a basin with $\|\nabla f(w_k)\|^2 \leqslant \nu^2$ and therefore, a complete complexity analysis can be performed by including the analysis for this sublinear phase too. Furthermore, to make our analysis simple and make it possible to compare our results with other methods in the literature, we limit our analysis to specific settings where we choose $\theta_k = 0$ and $\iota_k = \iota_0 a_g^k$ for all $k < k_{\text{sup}}$ and $\iota_k = \iota_0 a_l^k$ for all $k \geqslant k_{\text{sup}}$ which leads to fast local linear convergence (see Theorem 4.8).

Corollary 4.9. *Suppose Assumption 3.2 and conditions of Theorem 4.8 hold with $\theta_k = 0$, $\iota_k = \iota_0 a_g^k$ for all $k < k_{sup}$ where $a_g = 1 - \frac{\mu\tilde{\mu}}{2L\tilde{L}}$, and $\iota_k = \iota_0 a_l^k$ for all $k \geqslant k_{sup}$ with $a_l \in [0, 1)$, and $A_k = I$. Then, for any given (sufficiently small) $\epsilon > 0$, we get an ϵ-accurate solution after performing K_ϵ iterations where*

$$K_\epsilon = k_{sup} + \left\lceil \log_{1/\sqrt{a_l}} \left(\frac{r_{k_{sup}}}{\sqrt{\epsilon}} \right) \right\rceil = \tilde{\mathcal{O}}\left(\kappa^2 \left(1 + \frac{M}{\mu^{3/2}} \right) + \log\left(\frac{1}{\sqrt{\epsilon}} \right) \right), \tag{4.21}$$

where $r_{k_{sup}}$ is defined as in Theorem 4.8.

Proof. From (4.16), we have that for all $k \geqslant k_{\text{sup}}$,

$$\mathbb{E}[\|w_k - w^*\|] \leqslant r_{k_{\text{sup}}} (\sqrt{a_l})^{k - k_{\text{sup}}}. \tag{4.22}$$

We use induction to prove this statement. It is trivially satisfied for $k = k_{\text{sup}}$. Suppose this statement is true for some $k \geqslant k_{\text{sup}}$. Then, we have

$$\begin{aligned} \mathbb{E}[\|w_{k+1} - w^*\|] &\leqslant \max\left\{ r_k \rho_k, r_{k_{\text{sup}}} (\sqrt{a_l})^{k - k_{\text{sup}} + 1} \right\} \leqslant \max\left\{ r_{k_{\text{sup}}} (\sqrt{a_l})^{k - k_{\text{sup}}} \rho_k, r_{k_{\text{sup}}} (\sqrt{a_l})^{k - k_{\text{sup}} + 1} \right\} \\ &= r_{k_{\text{sup}}} (\sqrt{a_l})^{k - k_{\text{sup}} + 1}, \end{aligned}$$

where ρ_k is defined as in Theorem 4.8. Substituting K_ϵ in (4.22), we get

$$\mathbb{E}[\|w_k - w^*\|] \leqslant r_{k_{\text{sup}}} (\sqrt{a_l})^{\left\lceil \log_{1/\sqrt{a_l}} \left(\frac{r_{k_{\text{sup}}}}{\sqrt{\epsilon}} \right) \right\rceil} \leqslant \sqrt{\epsilon}.$$

We will now analyze k_{sup}. Following similar steps in the proof of Corollary 3.11, we have that $k_{\text{lin}} = 0$ and also utilizing

$$\sum_{i=k_0}^{k} (\mathbb{E}[\|w_i - w^*\|^2])^{1/2} < \frac{\sqrt{2\tilde{C}_2}}{\sqrt{\mu}(1-\sqrt{\tilde{\rho}_2})}, \tag{4.23}$$

we get $C_{p,s} := \frac{M\sqrt{2\tilde{C}_2}}{\sqrt{\mu}(1-\sqrt{\tilde{\rho}_2})}$. Therefore k_{sup} in (4.18), is updated as,

$$\begin{aligned} k_{\text{sup}} &= \left\lceil \max\left\{ \tfrac{9C_{p,s}\sqrt{\eta}}{\tilde{\mu}\sqrt{a_l}}, \tfrac{81\sigma_H^2\eta}{\tilde{\mu}^2|S_0|a_l}, 2\log_{1/\tilde{\rho}_2}\left(\tfrac{3q\sqrt{2\tilde{C}_2}}{\sqrt{\mu}\sqrt{a_l}}\right), \log_{1/a_g}\left(\tfrac{81q^2\iota_0}{a_l^2\tilde{\mu}^2}\right) \right\} \right\rceil \\ &= \tilde{\mathcal{O}}\left(\max\left\{ \tfrac{\kappa^2 M}{\mu^{3/2}}, \tfrac{\sigma_H^2}{\tilde{\mu}^2|S_0|}, \kappa^2, \kappa^2 \right\}\right) = \tilde{\mathcal{O}}\left(\kappa^2\left(1 + \tfrac{M}{\mu^{3/2}}\right)\right), \end{aligned} \tag{4.24}$$

where we employed $\log(1-x) \approx -x$ for sufficiently small x and $\sigma_H^2 \leqslant L^2$.

□

Remark 4.7. We make the following remarks about this result.

- The number of iterations required to transition from global linear phase to fast local linear or superlinear phase k_{sup} is similar to that of the deterministic sampling results established in Corollary 3.11 (excluding the dependence on N). Furthermore, k_{sup} is better (smaller) than the results established for uniform weighting scheme and comparable to nonuniform weighting scheme when $\frac{M}{\mu^{3/2}}$ is sufficiently small. However, we note that the analysis is in expectation and requires additional assumptions related to bounded moments and strong convexity of subsampled functions, which are not required in the deterministic sampling settings (see Section 3).
- While Corollary 4.9 requires ϵ to be sufficiently small, we can establish iteration complexity results for the global phase when ϵ is large. In this the iteration complexity result is given as $K_\epsilon = \tilde{\mathcal{O}}\left(\kappa^2 \log\left(\frac{1}{\mu\sqrt{\epsilon}}\right)\right)$.
- This iteration complexity compared to a stochastic gradient method or an adaptive sampling gradient method has better dependence on ϵ as seen in Table 2.

We will now establish the total gradient evaluations required to achieve an ϵ-accurate solution when the starting iterate is close enough to the optimum, w^*. We will only consider simpler settings common in stochastic gradient analysis where $\sigma_{1,g} = 0$ in Assumption 2.7b.

Corollary 4.10. *Suppose conditions of Corollary 4.9 are satisfied and $\sigma_{1,g} = 0$ in Assumption 2.7b. In addition, if w_0 is sufficiently close to w^* such that $f(w_0) - f(w^*) \leqslant \frac{L\iota_0}{\mu^2}$ and $\iota_0 \leqslant \frac{\mu^5}{324\eta^2 LM^2}$. Let $a_l = \frac{1}{2}$, and $|S_0| = \lceil \frac{81\eta}{2} \rceil$. Then, after computing*

$$\mathcal{W}_g = \mathcal{O}\left(\kappa^2\sigma_{2,g}^2 + \tfrac{\kappa\sigma_{2,g}^2}{\mu^2\epsilon}\right) \tag{4.25}$$

stochastic gradients, we achieve an ϵ-accurate solution.

Proof. From (2.8a), choosing minimum number of samples $|X_k|$ at each iteration to satisfy the stochastic norm condition [22], we get

$$|X_k| = \left\lceil \tfrac{\sigma_{2,g}^2}{\iota_k} \right\rceil \leqslant \tfrac{\sigma_{2,g}^2}{\iota_k} + 1.$$

The total number of gradient evaluations required to achieve an ϵ-accurate solution is then given as,

$$\begin{aligned}
\mathcal{W}_g &= \sum_{i=0}^{K_{\sup}-1} |X_k| + \sum_{i=k_{\sup}}^{K_\epsilon} |X_k| \\
&= \tfrac{\sigma_{2,g}^2}{\iota_0} \sum_{i=0}^{K_{\sup}-1} \tfrac{1}{a_g^i} + \tfrac{\sigma_{2,g}^2}{\iota_{k_{\sup}}} \sum_{i=k_{\sup}}^{K_\epsilon} \tfrac{1}{a_l^{i-k_{\sup}}} + k_{\sup} + 1 \\
&\leqslant \tfrac{\sigma_{2,g}^2}{\frac{1}{a_g}-1} (\tfrac{1}{a_g})^{k_{\sup}} + \tfrac{\sigma_{2,g}^2 a_l}{\iota_{k_{\sup}}(1-a_l)} (\tfrac{1}{a_l})^{K_\epsilon - k_{\sup}} + k_{\sup} + 1 \\
&\leqslant \underbrace{2\kappa^2 \sigma_{2,g}^2 (1-\tfrac{1}{2\kappa^2})^{-k_{\sup}}}_{①} + \underbrace{\tfrac{\sigma_{2,g}^2 a_l}{\iota_{k_{\sup}}(1-a_l)} (\tfrac{1}{a_l})^{K_\epsilon - k_{\sup}}}_{②} + \underbrace{k_{\sup}+1}_{③}
\end{aligned} \tag{4.26}$$

where we used $a_g = 1 - \frac{\mu\tilde{\mu}}{2L\tilde{L}} = 1 - \frac{1}{2\kappa^2}$. We will now analyze the terms on the right hand side of (4.26). Each term requires the analysis of $k_{\sup}$, so we proceed by analyzing that term. Using $f(w_0) - f(w^*) \leqslant \frac{L\iota_0}{\mu^2}$ and $\iota_0 \leqslant \frac{\mu^5}{324L\eta^2M^2}$, we get,

$$\tilde{C}_2 = \max\left\{ f(w_0) - f(w^*), \tfrac{\tilde{L}\iota_0}{\mu\tilde{\mu}} \right\} = \tfrac{L\iota_0}{\mu^2} \leqslant \tfrac{\mu^3}{324\eta^2M^2}. \tag{4.27}$$

From (4.24), we have that

$$\begin{aligned}
k_{\sup} &= \left\lceil \max\left\{ \tfrac{9C_{p,s}\sqrt{\eta}}{\tilde{\mu}\sqrt{a_l}}, \tfrac{81\sigma_H^2\eta}{\tilde{\mu}^2|S_0|a_l}, 2\log_{1/\tilde{\rho}_2}\left(\tfrac{3q\sqrt{2\tilde{C}_2}}{\sqrt{\mu}\sqrt{a_l}}\right), \log_{1/a_g}\left(\tfrac{81q^2\iota_0}{a_l^2\tilde{\mu}^2}\right) \right\} \right\rceil \\
&\leqslant \max\left\{ \tfrac{9C_{p,s}\sqrt{\eta}}{\tilde{\mu}\sqrt{a_l}}, \tfrac{81\sigma_H^2\eta}{\tilde{\mu}^2|S_0|a_l}, 2\log_{1/\tilde{\rho}_2}\left(\tfrac{3q\sqrt{2\tilde{C}_2}}{\sqrt{\mu}\sqrt{a_l}}\right), \log_{1/a_g}\left(\tfrac{81q^2\iota_0}{a_l^2\tilde{\mu}^2}\right) \right\} + 1
\end{aligned}$$

We will now analyze each term in the above bound. Consider,

$$\tfrac{9C_{p,s}\sqrt{\eta}}{\tilde{\mu}\sqrt{a_l}} \leqslant \tfrac{9M\sqrt{2\tilde{C}_2}\sqrt{\eta}}{\sqrt{\mu}\tilde{\mu}\sqrt{a_l}(1-\sqrt{\tilde{\rho}_2})} \leqslant \tfrac{1}{\sqrt{\eta}(1-\sqrt{\tilde{\rho}_2})} \leqslant \tfrac{2}{1-\tilde{\rho}_2} = 4\kappa^2, \tag{4.28}$$

where the first inequality is due to $C_{p,s} := \frac{M\sqrt{2\tilde{C}_2}}{\sqrt{\mu}(1-\sqrt{\tilde{\rho}_2})}$ for strongly convex functions, the second inequality is due to (4.27) and $\eta \geqslant 1$, the last inequality is due to the fact that $1 - \sqrt{x} \geqslant \frac{1-x}{2}$ for any $x \in [0,1]$ and $\eta \geqslant 1$, and the equality is due to $\tilde{\rho}_2 = 1 - \frac{\mu\tilde{\mu}}{2L\tilde{L}} = 1 - \frac{1}{2\kappa^2}$.

Considering the second term and using $|S_0| = \lceil \frac{81\eta}{2} \rceil$ and $\sigma_H^2 \leqslant L^2$, we get,

$$\tfrac{81\sigma_H^2\eta}{\tilde{\mu}^2|S_0|a_l} \leqslant \tfrac{4\sigma_H^2}{\tilde{\mu}^2} \leqslant 4\kappa^2. \tag{4.29}$$

Using (4.27), $\log(1 - \frac{1}{2\kappa^2}) \approx -\frac{1}{2\kappa^2}$, and substituting $q = \frac{7M\eta}{2\tilde{\mu}}$ in the third term, we get,

$$2\log_{1/\tilde{\rho}_2}\left(\tfrac{3q\sqrt{2\tilde{C}_2}}{\sqrt{\mu}\sqrt{a_l}}\right) \leqslant -2\tfrac{\log(7/6)}{\log(\tilde{\rho}_2)} \approx 4\kappa^2. \tag{4.30}$$

Similarly, using $\log(1 - \frac{1}{2\kappa^2}) \approx -\frac{1}{2\kappa^2}$, $\iota_0 \leqslant \frac{\mu^5}{324L\eta^2M^2}$, and substituting $q = \frac{7M\eta}{2\tilde{\mu}}$ in the fourth term, we get,

$$\log_{1/a_g}\left(\tfrac{81q^2\iota_0}{a_l^2\tilde{\mu}^2}\right) = -\tfrac{\log\left(\frac{3969M^2\eta^2\iota_0}{\mu^4}\right)}{\log\left(1-\frac{\mu\tilde{\mu}}{2L\tilde{L}}\right)} \leqslant -\tfrac{\log(\frac{49}{4\kappa})}{\log\left(1-\frac{\mu\tilde{\mu}}{2L\tilde{L}}\right)} \approx 5\kappa^2. \tag{4.31}$$

Therefore, combining all the above bounds, we get, $k_{\sup} \approx 5\kappa^2 + 1$.

We are ready to analyze the terms in (4.26). For the first term, we have that

$$2\kappa^2\sigma_{2,g}^2\left(1-\tfrac{1}{2\kappa^2}\right)^{-k_{\sup}} = \mathcal{O}\left(\kappa^2\sigma_{2,g}^2\right), \tag{4.32}$$

where we used $(1-1/x)^{-x} \approx \exp$, for any $x > 0$ that is sufficiently large. Consider,

$$\frac{r_{k_{\sup}}^2}{\iota_{k_{\sup}}} \leqslant \max\left\{\frac{(\mathbb{E}[\|w_{k_{\sup}}-w^*\|])^2}{\iota_0 a_g^{k_{\sup}}}, \tfrac{18}{\bar{\mu}^2}\right\} \leqslant \max\left\{\frac{2\tilde{C}_2\rho_2^{k_{\sup}}}{\mu\iota_0 a_g^{k_{\sup}}}, \tfrac{18}{\bar{\mu}^2}\right\} \leqslant \max\left\{\tfrac{2\tilde{C}_2}{\mu\iota_0}, \tfrac{18}{\bar{\mu}^2}\right\} \leqslant \max\left\{\tfrac{2\kappa}{\mu^2}, \tfrac{18}{\bar{\mu}^2}\right\} \tag{4.33}$$

where the last inequality is due to (4.27). Now, consider the second term in (4.26), we get

$$\frac{\sigma_{2,g}^2 a_l}{\iota_{k_{\sup}}(1-a_l)}\left(\frac{1}{a_l}\right)^{K_\epsilon - k_{\sup}} = \frac{\sigma_{2,g}^2 a_l}{\iota_{k_{\sup}}(1-a_l)}\left(\frac{1}{a_l}\right)^{\left\lceil \log_{1/\sqrt{a_l}}\left(\frac{r_{k_{\sup}}}{\sqrt{\epsilon}}\right)\right\rceil} \leqslant \frac{\sigma_{2,g}^2 r_{k_{\sup}}^2}{\epsilon\iota_{k_{\sup}}} + \frac{2\sigma_{2,g}^2}{\iota_{k_{\sup}}} = \mathcal{O}\left(\frac{\kappa\sigma_{2,g}^2}{\mu^2\epsilon}\right) \tag{4.34}$$

where the first equality is due to (4.21) and the last equality is due to (4.33). Combining (4.32) and (4.34) completes the proof. □

Remark 4.8. We make the following remarks about this result and make a comparison with existing results in Table 2.

- We employ fixed number of stochastic Hessian samples at each iteration. Therefore, the Hessian complexity, i.e. number of Hessian computations required to get an ϵ-accurate solution is same as the total number of iterations $\left(\tilde{\mathcal{O}}\left(\kappa^2 + \log\left(\frac{1}{\epsilon}\right)\right)\right)$.
- The total number of stochastic gradients required improves upon stochastic gradient method in terms of the dependence on condition number (κ) even though the number of gradients evaluated per-iteration are increasing at each iteration. Furthermore, we should note that our local results match with that of first-order adaptive gradient sampling methods even after employing inferior global convergence results associated with Newton's method compared to a gradient method.
- Although we only presented the gradient iteration complexity results here for the phase where the the starting iterate is sufficiently close to the optimal solution and ϵ is sufficiently small (see Corollaries 4.9 and 4.10), we can also establish the results for the global phase where ι_0 is chosen independent of the problem characteristics and $K_\epsilon = \tilde{\mathcal{O}}\left(\kappa^2 \log\left(\frac{1}{\epsilon}\right)\right)$. However such results are inferior to those of first-order adaptive gradient methods due to the artifact of global convergence analysis of Newton's method.

Method	Iteration	Gradient
Stochastic Gradient [18]	$\mathcal{O}\left(\frac{\kappa^2\sigma_{2,g}^2}{\mu^2\epsilon}\right)$	$\mathcal{O}(\frac{\kappa^2\sigma_{2,g}^2}{\mu^2\epsilon})$
Adaptive Gradient Sampling [22, 35]	$\mathcal{O}(\kappa\log\left(\frac{1}{\epsilon}\right))$	$\mathcal{O}\left(\frac{\kappa\sigma_{2,g}^2}{\mu^2\epsilon}\right)$
This Paper (Corollary 4.9 & 4.10)	$\tilde{\mathcal{O}}\left(\kappa^2 + \log\left(\frac{1}{\epsilon}\right)\right)$	$\mathcal{O}\left(\kappa^2\sigma_{2,g}^2 + \frac{\kappa\sigma_{2,g}^2}{\mu^2\epsilon}\right)$

Table 2: Comparison of (local) iteration and gradient complexity results for strongly convex functions. Note: The complexity results in terms of expected function values ($\mathbb{E}[f(w_k) - f(w^*)] \leqslant \epsilon$) for stochastic gradient and adaptive gradient sampling summarized in [22, Table 4.1] have been converted to get the results in terms of $(\mathbb{E}[\|w_k - w^*\|])^2 \leqslant \epsilon$.

5 Practical Algorithms

In the previous sections we developed a convergence and complexity theory that establish the convergence benefits of Hessian-averaged Newton methods for the finite-sum and expectation problem settings. However, hurdles remain to their deployment in very high-dimensional settings due to the costs of inverting Hessian matrices. In this section we discuss considerations relating to deploying Hessian-averaged Newton methods in practice. We begin by investigating a generic averaged Newton methods, which we refer to as fully-averaged Newton (FAN) for the purposes of our exposition. Due to the substantial costs of Hessian inversion this method is infeasible for even moderate problems. We propose a simple, diagonally averaged Newton method (Dan) for high-dimensional optimization problems such as those arising in machine learning (ML) settings. We investigate additional practical heuristics such as different weightings and norms that are used in the Hessian averaging, as well as practical gradient sampling strategies, which in turn lead to their own variant algorithms.

5.1 Fully-Averaged Newton

When the dimension of the optimization variable, d is not too large, one may opt to construct a Newton-like algorithm using a weighted average of the full Hessian, which requires d^2 storage:

$$\text{Fully-Averaged Newton (FAN):} \qquad p_k = \tilde{H}_k^{-1} g_k. \tag{5.1}$$

Here $\tilde{H}_k$ is defined as in (2.4), or alternatively using the following formula:

$$\tilde{H}_k = \sum_{i=0}^{k} \gamma_i |\nabla^2 F_{S_i}(w_i)| + \mu I, \tag{5.2}$$

where μI can help improve the conditioning of the problem. This requires however costly eigenvalue or Cholesky decomposition at each iteration, thus incurring $\mathcal{O}(d^3)$ computation. Alternative methods to approximate inversion of the full Hessian include Krylov methods [65, Chapter 5], however these methods do not simply extend to the averaging setting, since in this case one requires storage of the Hessian matrix in a given format, but not as a matrix-vector product callable as is necessary for Krylov methods.

5.2 Overcoming high-dimensionality with Hessian-subspace products

When the dimension d is large, approximations of the Hessian that can be formed for less than $\mathcal{O}(d^2)$ operations and inverted for less than $\mathcal{O}(d^3)$ operations are necessary. Randomized sketching is an easily extensible tool to construct efficient representations of matrices [62]. In this process one can construct a compressed representation of a matrix via its action on a matrix $V_r \in \mathbb{R}^{d\times r}$, where r is a small number, that is often independent of d. In addition to diagonal approximations, other factorizations such as low rank [40] and hierarchical [53, 91] approximations can be computed from the action of a given matrix on the r-dimensional subspace V_r.

This subspace action is easy to implement in modern ML workflows as it can be constructed from simple automatic differentiation tools around embarrassingly parallelizable linear algebra. For example, the Hessian subspace product can be computed with the same tools utilized to form the gradient; first by forming the gradient, then forming its transpose action on V_r, followed by taking the gradient (Jacobian) of this combined term:

$$\text{Hessian subspace products:} \qquad \nabla^2 F_{S_k}(w_k) V_r = \nabla \left(\nabla F_{S_k}(w_k)^T V_r \right). \tag{5.3}$$

E.g., forward-over reverse automatic differentiation [4] with vectorized GPU computing makes these Hessian approximations very computationally efficient. In our numerical tests, Hessian subspace products had approximately constant compute time until running out of GPU memory, see Figure 3.

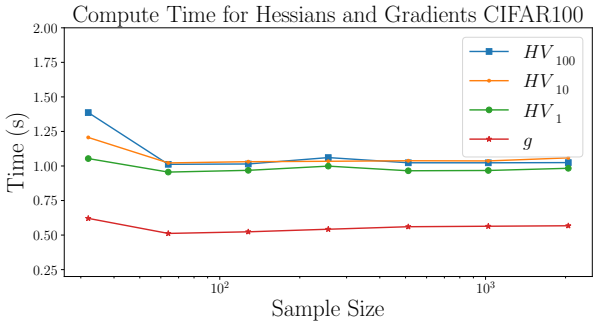

Figure 3: Efficiently implemented (vectorized) Hessian subspace products for varying ranks and sample sizes have approximately constant compute time until running out of GPU memory. These experiments are for a ResNet used in CIFAR100 classification, as shown in Section 6.4. The dimension of the neural network weights w was $d = 11,247,052$. This set of experiments was run on an NVIDIA L40S GPU which has 48GB of GPU RAM.

5.3 Diagonally-Averaged Newton

In practical settings for stochastic optimization, such as deep learning, d can be very large, and any algorithm requiring more than $\mathcal{O}(1)$ Hessian-vector products, and $\mathcal{O}(d)$ memory footprint at each iteration will be infeasible in modern compute settings, since they are typically memory bound [36].

These constraints are achievable utilizing both diagonal or low rank Hessian approximations. As diagonal preconditioning of gradients (with and without momentum) dominates modern ML optimization methods, it is sensible to utilize diagonal Hessian approximations in Newton-like stochastic gradient updates that are targeted to these problems. The use of diagonal Hessian preconditioners was first introduced, to the best of our knowledge, by [90], and was later used in [55].

The Hessian diagonal can be computed matrix-free via randomized Hutchinson diagonal estimation

$$D_k = \operatorname{diag}\left(\nabla^2 F_{S_k}(w_k)\right) = \mathbb{E}_{z\sim\pi_z}\left[\operatorname{diag}\left(\frac{z(\nabla^2 F_{S_k}(w_k)z)^T}{z^T z}\right)\right], \tag{5.4}$$

for a suitable choice of distribution π_z [31]. The diagonal matrix is trivially invertible in d operations, overcoming the major computational hurdle for Newton-like methods. This approximation, and other matrix representations that are easily inverted and have $\mathcal{O}(d)$ memory footprint can be constructed via the use of randomized sketching.

This naturally leads to the diagonally-averaged Newton method

$$\text{Diagonally-Averaged Newton (Dan):} \qquad p_k = \left[\tilde{D}_k\right]^{-1} g_k, \tag{5.5}$$

where in our implementation $\tilde{D}_k = \sum_{i=1}^{k} \gamma_i |D_k|$, that is we approximate the diagonal of (5.2) instead of (2.4). Diagonal approximations are useful for diagonally dominant Hessians, however these approximations may not be suitable for optimization problems with large off-diagonal components. We empirically observe that Dan and Adahessian perform well on difficult ML optimization problems.

5.3.1 Differences with Adahessian

Dan is similar to Adahessian [90] but differs in two key ways:

1. Dan does not have any momentum in the gradient, so the relative performance of Dan to Adahessian helps isolate the effects of utilizing the averaged Hessian approximation in isolation from the effects of gradient momentum.

2. Dan utilizes a ℓ^1 averaged approximation of the Hessian diagonal; this choice is motivated by our analysis which averages the Hessian and not its square. Adahessian and Sophia [55] utilize an ℓ^2 norm averaging, similar to what is used in Adam [47]. We denote by ℓ^p_γ the averaging protocol:

$$\ell^p_\gamma \text{ averaging:} \qquad \sqrt[p]{\sum_{i=1}^{k} \gamma_i D_i^p}. \tag{5.6}$$

Since Adahessian and Dan utilize different averagings, we thus propose an ℓ^2_γ modification of Dan, which we name Dan2:

$$\text{(Dan2)}: \qquad p_k = \left[\left(\sum_{i \leqslant k} \gamma_i D_i^2 \right)^{\frac{1}{2}} \right]^{-1} g_k. \tag{5.7}$$

5.4 Additional algorithmic considerations

In this section we consider additional adaptations of Dan and Dan2 that are common to other practical ML optimization methods.

5.4.1 Infrequent Hessian computations

The per-iteration costs of the Hessian approximations can add significant additional costs relative to first-order methods. In traditional stochastic Newton methods, this burden can be lessened by (i) lower dimensional Hessian approximations (e.g., few samples for Hutchinson diagonal estimation), (ii) smaller Hessian sample size (i.e., $|S_k| < |X_k|$). In the context of Hessian averaging we can additionally lessen the burden by updating the Hessian approximation less frequently than the gradient, as is done in Adahessian. We utilize this in numerical experiments in Section 6.4.2 where we maintain a fair cost-basis comparison between the first and second-order methods.

5.4.2 Non-uniform weightings

Adam, Adahessian and other popular ML optimizers utilize exponentially decaying sum averaging for their weightings, which is defined by the recurrence relationship

$$\text{Decaying Weights:} \qquad \widehat{D}_{k+1} = \frac{\beta_2}{1-\beta_2^k} \widehat{D}_k + \frac{(1-\beta_2)}{1-\beta_2^k} D_k, \tag{5.8}$$

where $\beta_2 \in (0,1)$ is a hyperparameter that controls the rate of decay in the averaging of past iterates, $\widehat{D}_k$ is the diagonal preconditioner being updated, and D_k is the estimator of the diagonal at w_k. The benefits of this approach are that the effects of past iterates are de-emphasized, which is useful when the landscape is highly nonlinear, and the diagonal preconditioner's local information is changing rapidly. This weighting is widely used due to its ease of implementation and effectiveness.

Our local convergence analyses in Section 3 and 4 utilize uniform averaging to concentrate the Hessian statistical sampling errors. In order to prove superlinear convergence for this weighting deterministically,

or in expectation, while still maintaining a fixed per-iteration Hessian sample size, other assumptions may be required. In [64] when the Hessian errors are assumed to be sub-Gaussian, superlinear local convergence rates are proven in probability for more general weightings such as (5.8). In numerical experiments we demonstrate the performance of this weighting and the uniform weighting.

5.4.3 Gradient sample size selection

While our algorithmic framework allows for fixed per-iteration computational costs associated with the Hessian approximation, in order to achieve fast convergence one has to control the gradient statistical sampling error in relation to the *true* gradient norm. This can be achieved either by geometrically increasing sample sizes (with sequence ι_k), or the norm test (with sequence θ_k), or a combination of both. The former (i.e., $\theta_k = 0$) is simple to implement in practice while the latter requires additional approximations as numerically verifying the bound $\mathbb{E}_k \left[\|\nabla F_{X_k}(w_k) - \nabla f(w_k)\|^2 \right] \leqslant \theta_k^2 \|\nabla f(w_k)\|^2$ may be prohibitive due to its sampling costs. To address this challenge, one can employ an approximate norm test in practice, as was done in previous works [12, 16, 22]:

$$\text{Approximate norm test:} \qquad \frac{1}{|S_k|} \sum_{i \in S_k} \|\nabla F_i(w_k) - \nabla F_{S_k}(w_k)\|^2 \leqslant \theta_k^2 \|\nabla F_{S_k}(w_k)\|^2 + \iota_k. \tag{5.9}$$

This test approximates the expectations via Monte Carlo, and will give a rough indicator of the convergence of sample gradient to the true gradient.

In large scale subsampled optimization problems (e.g., deep learning), methods with fixed gradient sample sizes show empirically good performance. Therefore, in numerical experiments we also consider variants of our algorithms that do not increase the gradient sample sizes, but instead use step size schedulers. This allows for direct comparison with state-of-the-art ML optimization routines. In particular Dan and Dan2 perform comparably to and often better than state-of-the-art methods in the ML optimization problems that we investigate.

6 Numerical Experiments

In this section, we experiment with Hessian-averaged subsampled Newton methods on a variety of problems, separated into two classes with different metrics: (1) subsampled convex problems where we look at $f(w) - f(w^*)$, and (2) subsampled nonconvex (deep learning) problems where we are interested in the performance of the trained models on unseen data. First we investigate algorithmic trade-offs for stochastic quadratic minimization and logistic regression problems where computational costs allow us to consider full Hessian inversions (FAN). We then consider large-scale neural network problems: CIFAR[10,100] classification with ResNets and neural operator training, where the weight dimensions d_W make full Hessian inversion prohibitive. In the large-scale context we investigate the performance of the Dan relative to Adam, Adahessian and SGD.

Overall, the Hessian-averaged Newton methods were run in regimes (e.g., choice sample sizes and step sizes) that led to immediate instabilities for subsampled Newton methods not utilizing Hessian averaging. This point demonstrates the key algorithmic motivation for this type of method: to alleviate the instabilities of subsampled Newton methods at manageable costs. Additionally in the large-scale machine learning (ML) problems, Dan and Dan2 performed comparably to and often better than Adam [47], overall better than Adahessian [90], and substantially better than SGD. This result suggests that the effects of Hessian averaging may be more beneficial than gradient momentum in some relevant practical settings.

6.1 Problem setups

We give a high level overview of the experiments below. The approximate solution for a given method is given by $w^\dagger$, which we do not denote w^* since it is not necessarily the true minimizer.

6.1.1 Subsampled quadratic

First we investigate a stochastic quadratic minimization problem:

$$\text{(Subsampled quadratic):} \qquad f(w) = \mathbb{E}_{P_A,P_b}\left[\|P_A A w - P_b b\|^2\right], \tag{6.1a}$$

$$\text{Evaluation criteria:} \qquad f(w^\dagger) \tag{6.1b}$$

where P_A, and P_b are linear operators that randomly zero out entries in A and b respectively. That is at each iteration we randomly sample index sets for entries in A and b that are set to zero. This problem is a simple analogue to empirical risk minimization over a dataset.

6.1.2 Logistic regression

Second we consider two logistic regression for binary classification with ℓ^2 regularization. Let x_i be an input vector and $y_i \in \{-1, 1\}$ be the corresponding output label

$$\text{(Logistic Regression):} \qquad f(w) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i(w^T x_i)) + \frac{1}{2n}\|w\|^2, \tag{6.2a}$$

$$\text{Evaluation criteria:} \qquad f(w^\dagger) \tag{6.2b}$$

We note that this objective function is strongly convex [6]. We compare the performance of various methods on both the `ijcnn1` and `mushroom` datasets [27].

6.1.3 CIFAR[10,100] Classification

Third, we consider the CIFAR10 and CIFAR 100 [50] classification using ResNet architectures [41], and a softmax cross-entropy loss function. Let x_i be an input image, and y_i be the corresponding vector label. Let $\phi(x_i, w)$ be the ResNet prediction in the pre-image of the softmax, then the problem is formulated as

$$\text{(Softmax cross entropy):} \qquad f(w) = -\frac{1}{n}\sum_{i=1}^{n} y_i^T \log\left(p(x_i, w)\right) \text{ where } p_j(x_i, w) = \frac{e^{\phi_j(x_i,w)}}{\sum\limits_{i\in\text{Classes}} e^{\phi_j(x_i,w)}} \tag{6.3a}$$

$$\text{Evaluation criteria:} \qquad \text{Correct classification percentage on unseen data.} \tag{6.3b}$$

6.1.4 Parametric PDE Regression

Finally, we consider regression problems for the approximation of parametric PDE input-output maps via neural networks (e.g., neural operators). We consider a coefficient-to-observable nonlinear reaction diffusion problem in a physical domain $\Omega \subset \mathbb{R}^2$. Here the input parameter $x \in \mathcal{X} = L^2(\Omega)$ is a heterogeneous spatially varying random field, with measure π. The PDE state $u \in \mathcal{U} = u_0 + H_0^1(\Omega)$, an affine space account for boundary conditions u_0. The outputs $y \in \mathbb{R}^{d_Y}$ represent finite observations of u on a line in the domain.

$$\text{PDE:} \qquad -\nabla \cdot (e^x \nabla u) + cu^3 = f \qquad \text{in } \Omega = (0,1)^2 \tag{6.4a}$$

$$\text{Boundary conditions:} \qquad u = 1 \text{ on } \Gamma_{\text{top}}, \qquad u = 0 \text{ on } \Gamma_{\text{bottom}}, \qquad \nabla u \cdot \mathbf{n} \quad = 0 \text{ on } \Gamma_{\text{sides}} \tag{6.4b}$$

$$\text{Parametric map:} \qquad x \mapsto y(x) = Bu(x), \tag{6.4c}$$

where $\mathbf{n}$ is the unit normal to the domain Ω, and $B : \mathcal{U} \to \mathbb{R}^{d_Y}$ is a linear restriction operator to points on line in the domain. The infinite-dimensional input functions are encoded to a finite input basis $\{\psi_i \in \mathcal{X}\}_{i=1}^r$ that is chosen to correspond to directions of the input space that the map is most sensitive to as in [68, 69]. The corresponding coefficients $x_r \in \mathbb{R}^r$ are $(x_r)_j = (x, \psi_j)_{\mathcal{X}}$, where $(\cdot,\cdot)_{\mathcal{X}}$ is the inner product for $\mathcal{X}$. This choice of encoding makes the regression task naturally finite-dimensional, while still maintaining discretization invariance via the use of the infinite-dimensionally consistent basis vectors ψ_i.

The regression task is thus to learn an approximation of the map $x_r \mapsto y$ by a neural network ϕ_w. We consider two formulations, first a parametric least squares formulation for learning the reduced coefficient map, which we refer to as the L^2_π formulation. Second we consider a least squares formulation for learning the reduced coefficient map and its first Fréchet derivatives (e.g., derivative-informed neural operator (DINO) [67]), which we refer to as the H^1_π formulation.

$$(L^2_\pi \text{ training}): \qquad \min_w f(w) = \frac{1}{n} \sum_{i=1}^n \|y(x_r) - \phi_w(x_r)\|_2^2 \tag{6.5a}$$

$$\text{Evaluation criteria:} \qquad \frac{\|y(x_r) - \phi_w(x_r)\|_2}{\|y(x_r)\|_2} \quad \text{on unseen data} \tag{6.5b}$$

$$(H^1_\pi \text{ training}): \qquad \min_w f(w) = \frac{1}{n} \sum_{i=1}^n \left(\|y(x_r) - \phi_w(x_r)\|_2^2 + \|\nabla_{x_r} y(x_r) - \nabla_{x_r} \phi_w(x_r)\|_F^2 \right) \tag{6.5c}$$

$$\text{Evaluation criteria:} \qquad \frac{\|y(x_r) - \phi_w(x_r)\|_2}{\|y(x_r)\|_2}, \frac{\|\nabla_{x_r} y(x_r) - \nabla_{x_r} \phi_w(x_r)\|_F}{\|\nabla_{x_r} y(x_r)\|_F} \quad \text{on unseen data .} \tag{6.5d}$$

The H^1_π formulation is particularly relevant when the surrogate is to be deployed in a setting where accurate derivatives are required, such as for the solution of optimization problems [59], or efficient Bayesian inference in function spaces [24]. The H^1_π training problem can thus be considered optimizing to optimize. We note the recent work [94] has also investigated the performance of second-order optimization methods for training parametric PDE surrogates.

6.1.5 Additional details

In order to have a one-to-one comparison of methods, our implementation of Adahessian differs slightly from the method proposed in [90]. First we consider versions of Adahessian that update the Hessian approximation both at each iteration, and also infrequently (the latter is proposed in [90]). Additionally we do not use averaging of convolution layers in the diagonal approximation, as we propose generic Hessian-vector products for Dan and Dan2 that are agnostic to the structure of what is being differentiated. Further details on the implementation details are given in Appendix 9.2, accompanied by an extended discussion of the parametric PDE problem in Appendix 9.3.

6.2 Subsampled Quadratic Minimization

For this problem we compare FAN, Dan, Newton, and SGD all with and without adaptive gradient sampling (denoted "a.g."), which is implemented via the norm test. We do a sweep over fixed step sizes $\alpha \in [1.0, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}]$. For the averaged methods we consider both uniform and

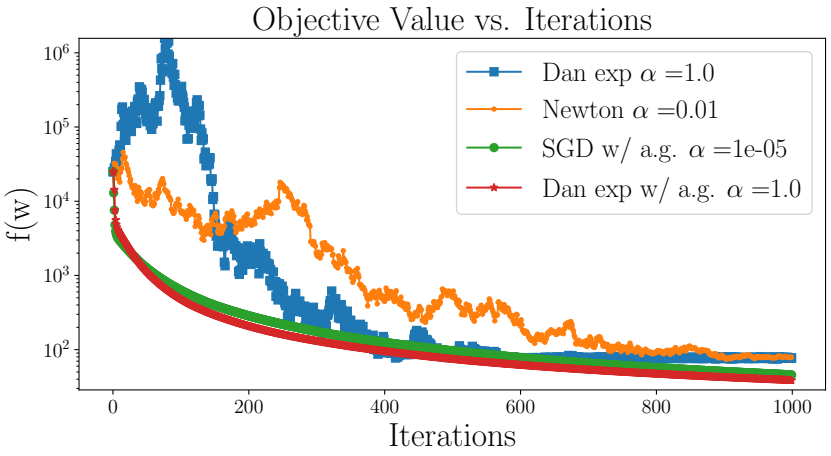

Figure 4: The performance of the best 4 methods for the stochastic quadratic minimization problem. Dan with adaptive gradient sampling performed the best, both in terms of fast convergence and $f(w^\dagger)$. SGD with adaptive gradient sampling also performed well, but required significant limitations on the step size. The methods without adaptive gradient sampling plateaued with larger $f(w^\dagger)$. The averaging of the Hessian can eventually overcome stability issues after enough iterations progress to reduce the Hessian variance as seen by Dan exp $\alpha = 1.0$. When not using Hessian averaging, Newton methods required smaller steps to maintain stability.

exponentially decaying weightings. Initially P_A, P_b are both taken to randomly zero out 50% of the entries in A, b respectively. All optimizers start from the same initial guess $w_0 \sim \mathcal{N}(0, I_{dW})$, where the stochastic gradient is approximately $13,500\%$ noisier than the true gradient. Each optimizer runs for $1,000$ iterations. We note that this does not constitute a fair comparison in terms of computational work, since we compare iterations. However, the methods that did not utilize adaptive gradient sampling were not making progress after $1,000$ iterations, so the gap in the objective function is indicative of superior performance for the adaptively sampled methods, and the trend would continue if run longer. We report general trends over the 72 different numerical runs. First all of the SGD runs without adaptive gradient sampling diverged except for $\alpha = 10^{-5}$; with adaptive gradient sampling all diverged except $\alpha \in \{10^{-4}, 10^{-5}\}$.

The performance of the ten best methods is reported below in Table 3, and the (rolling average) of the stochastic function costs are shown in Figure 4. The general trend for this problem is that the Dan methods outperformed all of the other Newton methods consistently. SGD required very small step sizes for stability, but performed well when taking many very small step sizes. The best performing methods utilized adaptive gradient sampling. Methods that did not utilize adaptive gradient sampling were able to reduce the objective function rapidly, but got stuck in neighborhoods where the objective function was not able to be reduced beyond the level of the noise. Perhaps surprising is the superior empirical performance of Dan to all other Newton methods. Dan is an economical Hessian approximation, where one might expect the omitted off-diagonal information would lead to deteriorated performance. For this problem, however, this is not the case. This bolsters the argument for this numerical approximation in high-dimensional settings where true Hessian inverse approximation is infeasible.

	Method	α	$f(w^\dagger) \searrow$		Method	α	$f(w^\dagger) \searrow$
1	Dan exp. w/ a.g.	1.0	**38.67**	6	Newton	10^{-3}	76.35
2	SGD w/ a.g.	10^{-5}	45.69	7	FAN exp	10^{-1}	76.98
3	Dan exp	10^{-1}	73.77	8	Dan exp	10^{-1}	77.01
4	Newton	10^{-2}	74.69	9	Dan exp	10^{-2}	77.11
5	SGD	10^{-5}	75.11	10	Dan uni	1.0	77.12

Table 3: The ten best performing methods for the subsampled quadratic minimization, and the objective function value at their approximated optimum $w^\dagger$. The two best performing methods utilized adaptive gradient sampling, but interestingly the adaptive gradient sampling didn't lead to optimal results outside of these methods. In general Dan outperforms all of the other second-order methods, in particular the exponentially decaying weights generally outperformed the uniform averaging.

For the remainder of the numerical results, we compare methods either for a fixed number of epochs or for fixed computational costs. In the case of a fixed total number of epochs, the adaptive sampling methods require fewer total computations than the same method without adaptive sampling, as these methods require fewer iterations and therefore fewer Hessian computations compared to the same method without adaptive sampling.

6.3 Logistic Regression

For a second numerical result we compare Dan, FAN, Newton, and SGD on two logistic regression problems. The `ijcnn1` dataset has 22 input features and $35,000$ training samples. The `mushroom` dataset has 112 features and $5,500$ training data. We compare fully subsampled methods (gradient and Hessian sample sizes fixed at 32), with adaptive gradient sampling methods. We run a total of 100 epochs for each method. For adaptive gradient sampling, we implement geometric growth in sample size instead of the norm test: we run 20 epochs with gradient sample sizes $|X_k| = \{32, 128, 512, 2048, 5500\}, \{32, 128, 512, 2048, 8192\}$ for `ijcnn1` and `mushroom` respectively. We note that in this case, since we consider epochs and not iterations, the adaptively sampled methods will require a lower total amount of computational work than the methods that do not utilize adaptive gradient sampling. For the ijcnn1 results the algorithms performed roughly the same, we show results for the step size of $\alpha = 0.1$, which are representative of other step sizes. For the mushroom dataset we show the performance of all methods as a function of the step size. The results are show below in Table 4.

	ijcnn1 $f(w^\dagger) \searrow$		mushroom $f(w^\dagger) \searrow$		
		a.g.	α		a.g.
Dan	0.398	**0.349**	0.1	(✗, 0.123)	(✗, 0.072)
			0.01	(✗, 0.123)	(✗, 0.078)
			0.001	(0.136, 0.137)	(0.105, 0.102)
FAN	0.401	**0.349**	0.1	0.121	**0.071**
			0.01	0.123	0.079
			0.001	0.142	0.111
Newton	0.405	**0.349**	0.1	0.132	0.072
			0.01	0.132	0.072
			0.001	0.150	0.103
SGD	0.389	**0.349**	0.1	0.123	0.092
			0.01	0.145	0.125
			0.001	0.198	0.201

Table 4: Results for the two logistic regression problems are reported as averages over five different initial guesses. The methods performance improved uniformly with adaptive gradient sampling. For the `ijcnn1` dataset the four methods performed comparably. The symbol ✗ denotes that these runs diverged. For the `mushroom` problem, Dan is reported for two different choices of the diagonal estimator rank, $r \in \{1, 40\}$; for larger step sizes Dan diverged with $r = 1$, but otherwise performed comparably to FAN; this demonstrates that using more Hessian-vector products in the diagonal estimation can lead to better performance. FAN performed better than subsampled Newton without adaptive gradient sampling, and worse with it. Both Dan and FAN uniformly outperformed SGD.

6.4 CIFAR[10,100] classification with ResNets

Next we demonstrate the performance of Dan in deep learning classication problems. We use the CIFAR10 and CIFAR 100 [50] classification using ResNet architectures [41], and a softmax cross-entropy loss function. For both problems, we use the standard 50,000 training data, and report generalization accuracy over the remaining test data. For all results we utilize a single learning rate scheduler that quarters the learning rate every 25% of total epochs. For both problems we investigate the performance of the methods in two regimes:

1. Comparison over 100 epochs. An extensive fixed data-access comparison of all methods with varying algorithmic hyperparameters. We note that this is not strictly a fair comparison, but it allows us to gain insight into the general performance of the different methods before running costlier, long trainings.

2. Comparison of longer runs, on a fixed computational cost basis. We do a limited number of much longer training runs to compare how Dan, Dan2 and Adahessian compare to Adam.

We note that other adaptive optimizers (e.g., RMSProp [44] and Adagrad [32]) performed substantially worse than Adam, so we omitted them. As these numerical results are very expensive to run, we rerun over two seeds and report the average results. The problems are quite computationally expensive and were run on NVIDIA A100 / L40S GPUs.

6.4.1 Comparison of methods over 100 epochs

We begin with our comparison of methods over 100 epochs. For this set of results we investigate how the performance of the three Hessian-averaging based methods (Adahessian, Dan and Dan2) is effected by the number of Hessian vector products used in the Hutchinson diagonal estimation (5.4). For a

fair baseline of comparison, we additionally implement a version of Adahessian where a new Hessian approximation is computed at every iteration, as was discussed in Section 5.4.1. We explore the effects of limiting the frequency of Hessian computations in the next set of numerical results. We investigate the performance of Dan, Dan2 and SGD with and without adaptive gradient sampling. As with the last example, we implement a geometric increase of the gradient sample sizes. We first run 75 epochs with $|X_k| = 32$, followed by five epochs each with the following sequence $X_k \in \{64, 128, 256, 512, 1024\}$. As a consequence the Dan w/ a.g. is notably substantially less expensive than regular Dan due to the lower iteration complexity in the final 25 epochs. Our results are shown below in Table 5.

Method	α_0	Hessian rank	CIFAR10 Accuracy ↗	CIFAR100 Accuracy ↗
Adahessian	0.05	$(1, 5, 10)$	$(92.85, 92.70, 92.55)$	$(71.97, 71.67, 72.15)$
	0.01	$(1, 5, 10)$	$(92.47, 93.20, 93.21)$	$(70.65, 71.93, 72.11)$
	0.001	$(1, 5, 10)$	$(83.35, 86.19, 87.52)$	$(57.49, 60.10, 62.82)$
Adam	0.05	-	✗	✗
	0.01	-	91.69	65.02
	0.001	-	93.08	72.10
Dan	0.05	$(1, 5, 10)$	$(91.80, 92.31, 90.74)$	$(70.62, 71.00, 71.01)$
	0.01	$(1, 5, 10)$	$(92.29, 93.30, 93.32)$	$(71.18, 72.10, 71.57)^{\dagger}$
	0.001	$(1, 5, 10)$	$(85.44, 86.87, 89.04)$	$(60.33, 61.78, 64.98)$
Dan w/ a.g.	0.05	$(1, 5, 10)$	$(92.99, 92.16, 92.56)$	$(\mathbf{72.48}, 70.96, 70.52)$
	0.01	$(1, 5, 10)$	$(92.62, 93.16, 93.36)$	$(71.11, 72.03, 72.22)$
	0.001	$(1, 5, 10)$	$(82.41, 86.78, 89.13)$	$(56.51, 62.06, 64.92)$
Dan2	0.05	$(1, 5, 10)$	$(92.81, 92.80, 92.40)$	$(71.34, 71.70, 71.53)$
	0.01	$(1, 5, 10)$	$(92.47, \mathbf{93.37}, 93.14)$	$(70.89, 72.22, 71.89)$
	0.001	$(1, 5, 10)$	$(80.98, 85.54, 87.78)$	$(54.74, 60.55, 62.40)$
Dan2 w/ a.g.	0.05	$(1, 5, 10)$	$(93.03, 92.66, 93.02)$	$(72.09, 72.25, 72.33)$
	0.01	$(1, 5, 10)$	$(91.90, 92.99, 92.96)$	$(70.38, 72.13, 71.98)$
	0.001	$(1, 5, 10)$	$(81.27, 86.02, 87.63)$	$(54.74, 60.55, 62.40)$
SGD	0.05	-	✗	✗
	0.01	-	88.09	64.84
	0.001	-	81.75	54.36
SGD w/ a.g.	0.05	-	✗	✗
	0.01	-	88.58	66.63
	0.001	-	82.00	53.77

Table 5: Comparison of optimization methods for CIFAR[10,100] ResNet training over 100 epochs. In both cases a Hessian-averaging based method produced the highest average generalization accuracy. Notable takeaways are that Dan with adaptive gradient led to the best result for the CIFAR100, showing the promise and competitiveness of our proposed framework in complicated deep learning tasks. Additionally, the Hessian averaging methods were able to take larger steps than the subsampled gradient methods. Adam was reliable as is usually the case, while SGD overfit substantially and produced poor generalization accuracys. †: See Appendix 9.2.2.

Our proposed methods Dan and Dan2 produced the best overall results in terms of generalization accuracy. Adam and Adahessian were both competitive, while SGD was not competitive. The adaptive gradient sampling was able to improve the performance, for example Dan w/ a.g. for CIFAR100 notably produced significantly better results than the other methods. This demonstrates not only is Hessian averaging without momentum a good algorithmic building block, but additionally can lead to better results at lower costs, due to the geometric reduction in Hessian computational costs in the later epochs.

Of note for other algorithmic considerations: in general the additional Hessian vector products in the diagonal estimations didn't clearly benefit performance, other than when taking small step sizes, in this case there is a clear trend that increasing the rank of the diagonal approximation led to better performance, however these methods performed substantially worth overall to the larger step sizes.

6.4.2 Comparison of methods with respect to computational cost

In this section we investigate the performance of Adahessian, Dan and Dan2 in relation to Adam in terms of a computational cost, taking the additional Hessian computations into account. For this section we introduce a new algorithmic hyperparameter: the Hessian computation frequency, as discussed in 5.4.1. The modifications of Adahessian, Dan and Dan2 to compute Hessians infrequently significantly reduces the overall cost, yielding per-iteration costs closer to Adam's per-iteration cost. We only start utilizing infrequent Hessian computations after the first epoch, as empirically the Hessian statistical errors didn't concentrate fast enough to keep the iterates from diverging; a reminder of the inherent instability of subsampled Newton iterates in this regime. After the first epoch we only update the Hessian diagonal approximations every 10 iterations. We investigate a study of Hessian ranks of 1 and 5 for these methods. We introduce an epoch equivalent compute metric, as a means of putting the second-order and first-order methods on an equivalent cost basis:

$$\text{(Epoch Equivalent Compute (E.E.C.))} = \left(\underbrace{1}_{\text{gradients}} + \underbrace{\frac{2 \times \text{rank}}{\text{h.f.}}}_{\text{Hessians}} \right) \times \text{epochs} + \underbrace{2 \times \text{rank}}_{\text{first epoch Hessian}} . \quad (6.6)$$

We count one Hessian vector product as twice the cost of a gradient, when in reality it is empirically cheaper when utilizing vectorization (see Figure 3), these costs comparison are therefore conservative and make the Hessian-based methods seem more expensive than they may be given an efficient GPU implementation. An additional means of computational economy is reducing the Hessian sample size relative to the gradient sample size. We do not investigate this; as with the previous set of numerical experiments we use a sample size of 32 for both the gradients and the Hessians. In these experiments we do not investigate adaptive gradient sampling. Adam with $\alpha_0 = 10^{-3}$ is our reference method, we run it for 500, 1000 and 2000 epochs. We investigate Adahessian, Dan and Dan2 in comparison to Adam on a similar cost basis. These results are shown below in Table 6. The best individual runs over all hyperparameters and seeds for Adam, Adahessian, Dan and Dan2 are shown below in Figure 5.

Method	epochs	Hessian rank	E.E.C.	α_0	CIFAR10	CIFAR100
Adam	2000	-	2000	0.001	94.00	73.06
	1000	-	1000	0.001	93.78	72.39
	500	-	500	0.001	93.58	72.39
Adahessian	1000	(1, 5)	(1202, 2010)	0.05	(✗,93.44)	(72.96, 73.03)
		(1, 5)	(1202, 2010)	0.01	(93.16, 93.97)	(69.57, 72.50)
	500	(1, 5)	(602, 1010)	0.05	(93.65, 93.32)	(72.30, 69.42)
		(1, 5)	(602, 1010)	0.01	(93.07, 93.60)	(70.72, 71.36)
Dan	1000	(1, 5)	(1202, 2010)	0.05	(91.98.90.83)	(**73.74**, 73.01)
		(1, 5)	(1202, 2010)	0.01	(93.56, **94.10**)	(71.62, 72.73)
	500	(1, 5)	(602, 1010)	0.05	(92.31, 91.32)	(73.30, 73.05)
		(1, 5)	(602, 1010)	0.01	(93.03, 93.70)	(71.30, 72.30)
Dan2	1000	(1, 5)	(1202, 2010)	0.05	(93.72, 93.71)	(73.30, 72.99)
		(1, 5)	(1202, 2010)	0.01	(93.07, 93.85)	(71.82, 72.91)
	500	(1, 5)	(602, 1010)	0.05	(93.80, 93.08)	(72.61, 73.05)
		(1, 5)	(602, 1010)	0.01	(92.80, 93.43)	(70.94, 72.29)

Table 6: Comparison of Adam, Adahessian, Dan and Dan2 for equivalent computational cost bases. For the CIFAR10 dataset the performance of Adam was comparable to the best performing second-order methods, with Dan performing the best, but with some variance in performance. For CIFAR100 the performance of the second-order methods was drastically better than Adam. This is consistent with the results in the preceding section.

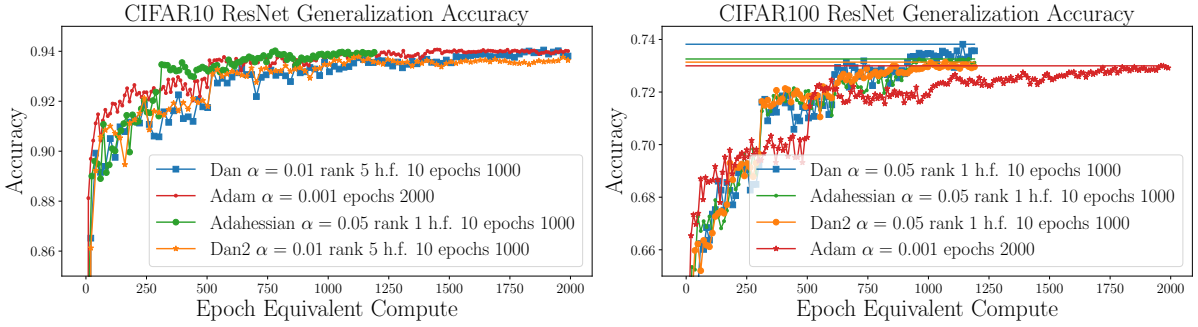

Figure 5: Comparison of the best runs for Adahessian, Adam, Dan and Dan2 in regards to epoch equivalent work. The best Adahessian happened to have lower Hessian ranks in the approximation than the best Dan and Dan2 for the CIFAR10, while for CIFAR100 the rank 1 methods shown all performed the best of individual runs. The CIFAR10 performance is quite similar for all four methods, while for CIFAR100 the Hessian-averaged methods substantially out-performed Adam; particularly Dan gave the best generalization accuracy (73.82% for the run shown).

These results overall show the power of Hessian-averaged methods for difficult deep learning problems. While it was previously known that Adahessian performs well for these problems, these results show that the momentum in the gradient utilized by Adahessian may not be necessary to achieve good performance. Our algorithms (Dan, Dan2) do not use gradient momentum, and are competitive in challenging deep learning classification problems.

6.5 Parametric PDE Regression

For a last numerical result we investigate the performance of Hessian averaged methods on some parametric PDE regression training problems; first where we learn a parametric map from input-output function data, and second where we include parametric (Fréchet) derivatives as additional training data. These problems differ from the previous deep learning examples because (1) they are regression tasks, (2) the neural network representation is more compact, (3) there are substantially fewer sample data and (4) the derivative-learning task includes very rich information per sample. For additional details on neural operators see Appendix 9.3

We train with each method for 200 epochs on 4500 samples of the PDE map (and its derivatives in the H^1_π case), and use 500 samples to compute generalization (relative) errors. We employ a one-step learning rate scheduler that reduces the step length by $10\times$ at the 150th epoch. We use a five layer feedforward network with gelu activation [42]; the corresponding weight dimension is $d_W = 742,050$. Since the weights are lower dimensional, we numerically experiment with using larger rank Hessian approximations $k = (1, 20, 40)$, and demarcate the corresponding errors for these methods in tuples as such. The results are shown below in Table 7. In general for these problems, Adam performed reliably well in both the L^2_π and H^1_π training problems. The Hessian averaged methods performed worse than Adam on the L^2_π problem. For the H^1_π problem Dan and Dan2 performed about the same as Adam, with Adahessian performing slightly worse. SGD performed substantially worse than all other methods on this problem. We believe the superior performance of the Hessian averaged methods on the H^1_π problem can be explain by the richer training data (e.g., the derivative training data) supplied for this problem; indeed the divergence experienced by Dan and Dan2 in the L^2_π training is mitigated in the H^1_π problem. Perhaps the additional data per iteration reduced statistical sampling errors that may have caused early iteration divergence for Dan and Dan2 in the L^2_π training problem.

6.6 Summary of findings

We summarize our findings as follows

- Hessian averaging can overcome the instabilities of subsampled Newton methods and still produce good Hessian-like operators for generating optimization iterates.
- Dan and Dan2 (as with Adahessian) are competitive with Adam in per-iteration computational work, particularly when reducing the frequency of Hessian computations. All three were empriri-cally superior to SGD in the experiments that we conducted. Additionally, efficiently implemented Hessian approximations using GPU vectorized computations leads runtime performance that is independent of the subspace rank r before GPU memory is exhausted.
- Dan and Dan2 had competitive and often better performance than state-of-the-art methods Adam and Adahessian in our experiments. Using adaptive gradient sampling can simultaneously improve performance (e.g., generalization accuracy) while also reducing total computation as the Hessian computations per-epoch are reduced as the gradient sample size is increased.

7 Conclusions

In this work we have advanced Hessian-averaged Newton methods with adaptive gradient sampling as a compelling class of fully-inexact methods with significant advantages both theoretically and practically. Our global and local convergence theory demonstrates that these methods are capable of fast local convergence for fixed per-iteration Hessian computations, when utilizing generalized norm test to determine gradient sample sizes. When utilizing deterministic sampling strategies without replacement,

Method	α_0	L^2_π training	H^1_π training	
		(y_r rel error $\searrow$)	(y_r rel error $\searrow$)	($\nabla_{x_r} y_r$ rel error $\searrow$)
Adahessian	0.01	$(0.366, 0.366, 0.362)$	$(0.012, 0.010,$✗$)$	$(0.260, 0.258,$✗$)$
	0.005	$(0.102, 0.097, 0.102)$	$(0.010, 0.010, 0.011)$	$(0.258, 0.258, 0.259)$
	0.001	$(0.121, 0.12, 0.122)$	$(0.015, 0.013, 0.012)$	$(0.264, 0.262, 0.260)$
Adam	0.01	0.361	0.028	0.274
	0.005	0.078	0.023	0.264
	0.001	**0.054**	0.008	**0.255**
Dan	0.01	(✗,✗,✗)	$(0.01, 0.008,$✗$)$	$(0.257, 0.256,$✗$)$
	0.005	$(0.09, 0.084,$✗$)$	$(0.009, 0.009,$ **0.007**$)$	$(0.257, 0.256, 0.256)$
	0.001	$(0.12, 0.118, 0.114)$	$(0.014, 0.013, 0.012)$	$(0.263, 0.261, 0.260)$
Dan w/ a.g.	0.01	(✗,✗,✗)	$(0.011, 0.009, ‡)$	$(0.258, 0.257, ‡)$
	0.005	$(0.108, 0.088, 0.078)$	$(0.013, 0.009, ‡)$	$(0.261, 0.256, ‡)$
	0.001	$(0.145, 0.134, 0.126)$	$(0.025, 0.014, ‡)$	$(0.273, 0.263, ‡)$
Dan2	0.01	(✗,✗,✗,)	$(0.009, 0.009, 0.009)$	$(0.257, 0.256, 0.256)$
	0.005	$(0.092, 0.088, 0.084)$	$(0.01, 0.009, 0.008)$	$(0.257, 0.257, 0.256)$
	0.001	$(0.12, 0.119, 0.118)$	$(0.015, 0.014, 0.012)$	$(0.264, 0.262, 0.260)$
Dan2 w/ a.g.	0.01	(✗,✗,✗)	$(0.01, 0.009, ‡)$	$(0.259, 0.256, ‡)$
	0.005	$(0.134, 0.088, 0.082)$	$(0.014, 0.009, ‡)$	$(0.262, 0.257, ‡)$
	0.001	$(0.144, 0.137, 0.131)$	$(0.027, 0.013, ‡)$	$(0.284, 0.263, ‡)$
SGD	0.01	0.283	0.046	0.313
	0.005	0.154	0.042	0.310
	0.001	0.204	0.056	0.355
SGD w/ a.g.	0.01	0.157	0.057	0.057
	0.005	0.206	0.073	0.403
	0.001	0.297	0.118	0.534

Table 7: Results of neural operator training for the reaction diffusion PDE problem. Hessian averaged methods use $(1, 20, 40)$ vectors for diagonal estimation. For the L^2_π training problem, Adam reliably performed the best; while Dan, Dan2 ran into some stability issues for $\alpha_0 = 0.01$. When Dan and Dan2 did not diverge they performed better than Adahessian which performed better than SGD. For the H^1_π training problem, the Hessian-averaged Newton methods performed much better; for these problems, Dan, Dan2 and Adam all performed somewhat comparably, with Adahessian performing slightly worse and SGD performing substantially worse. The H^1_π training problem has much richer information per datum, perhaps leading to more well-informed subsampled Hessian approximations, leading to better performance. ‡: these runs failed to complete in the final iterations due to out of memory error, this could be overcome with a more efficient implementation, such an implementation is however outside of the scope of this work.

we developed a local superlinear convergence rate, that improved the best existing rate from $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ to $\mathcal{O}\left(\frac{1}{k}\right)$, while simultaneously relaxing assumptions of gradient exactness and maintaining similar global to local transition phase iteration complexity. We additionally extended our analysis to the stochastic gradient sampling setting, establishing a full convergence theory in expectation, matching the $\mathcal{O}\left(\frac{1}{k}\right)$ rate of [46, 64], albeit using additional assumptions. Furthermore, we establish local gradient complexity results matching those of adaptive first-order gradient methods and improving upon stochastic gradient methods.

From a practical standpoint, we advanced Hessian-averaging as a variance reduction strategy that reduced stochasticity-driven instabilities of second-order methods. From a computational cost perspective, we emphasized how matrix-free Hessian approximations can be efficiently computed in modern computing frameworks for $\mathcal{O}(1)$ per-iteration Hessian-vector products and computational time, only requiring $\mathcal{O}(d)$ memory. We introduce the efficient diagonally-averaged Newton methods (Dan and Dan2) as practical extensions of the algorithmic framework that we investigate in this work. In numerical experiments, we demonstrated that these methods are not only computationally competitive with first-order methods, but consistently produce competitive and often superior generalization accuracies in complex deep learning tasks.

8 Acknowledgements

This work was partially supported by the National Science Foundation under award DMS-2324643. The authors would like to thank Albert Berahas and Shagun Gupta for feedback on some of the writing. The authors would like to thank Omar Ghattas and Umberto Villa for access to computing resources.

References

[1] N. Agarwal, B. Bullins, and E. Hazan, *Second-order stochastic optimization for machine learning in linear time*, Journal of Machine Learning Research, 18 (2017), pp. 1–40.

[2] J. Ba, R. B. Grosse, and J. Martens, *Distributed second-order optimization using kronecker-factored approximations.*, in ICLR (Poster), 2017.

[3] B. Bartan and M. Pilanci, *Distributed sketching for randomized optimization: Exact characterization, concentration, and lower bounds*, IEEE Transactions on Information Theory, 69 (2023), pp. 3850–3879.

[4] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, *Automatic differentiation in machine learning: a survey*, Journal of machine learning research, 18 (2018), pp. 1–43.

[5] F. Beiser, B. Keith, S. Urbainczyk, and B. Wohlmuth, *Adaptive sampling strategies for risk-averse stochastic optimization with constraints*, IMA Journal of Numerical Analysis, 43 (2023), pp. 3729–3765.

[6] A. S. Berahas, R. Bollapragada, and J. Nocedal, *An investigation of Newton-sketch and subsampled Newton methods*, Optimization Methods and Software, 35 (2020), pp. 661–680.

[7] A. S. Berahas, R. Bollapragada, and J. Shi, *Modified line search sequential quadratic methods for equality-constrained optimization with unified global and local convergence guarantees*, arXiv preprint arXiv:2406.11144, (2024).

[8] A. S. Berahas, R. Bollapragada, and B. Zhou, *An adaptive sampling sequential quadratic programming method for equality constrained stochastic optimization*, arXiv preprint arXiv:2206.00712, (2022).

[9] A. S. Berahas, M. Jahani, P. Richtárik, and M. Takáč, *Quasi-Newton methods for machine learning: forget the past, just sample*, Optimization Methods and Software, 37 (2022), pp. 1668–1704.

[10] A. S. Berahas, J. Nocedal, and M. Takáč, *A multi-batch L-BFGS method for machine learning*, Advances in Neural Information Processing Systems, 29 (2016).

[11] V. I. Bogachev, *Gaussian measures*, no. 62, American Mathematical Soc., 1998.

[12] R. Bollapragada, R. Byrd, and J. Nocedal, *Adaptive sampling strategies for stochastic optimization*, SIAM Journal on Optimization, 28 (2018), pp. 3312–3343.

[13] R. Bollapragada, R. H. Byrd, and J. Nocedal, *Exact and inexact subsampled Newton methods for optimization*, IMA Journal of Numerical Analysis, 39 (2019), pp. 545–578.

[14] R. Bollapragada, C. Karamanli, B. Keith, B. Lazarov, S. Petrides, and J. Wang, *An adaptive sampling augmented Lagrangian method for stochastic optimization with deterministic constraints*, Computers & Mathematics with Applications, 149 (2023), pp. 239–258.

[15] R. Bollapragada, C. Karamanli, and S. M. Wild, *Derivative-free optimization via adaptive sampling strategies*, arXiv preprint arXiv:2404.11893, (2024).

[16] R. Bollapragada, J. Nocedal, D. Mudigere, H.-J. Shi, and P. T. P. Tang, *A progressive batching L-BFGS method for machine learning*, in International Conference on Machine Learning, PMLR, 2018, pp. 620–629.

[17] R. Bollapragada and S. M. Wild, *Adaptive sampling quasi-newton methods for zeroth-order stochastic optimization*, Mathematical Programming Computation, 15 (2023), pp. 327–364.

[18] L. Bottou and O. Bousquet, *The tradeoffs of large scale learning*, Advances in neural information processing systems, 20 (2007).

[19] L. Bottou, F. E. Curtis, and J. Nocedal, *Optimization methods for large-scale machine learning*, SIAM review, 60 (2018), pp. 223–311.

[20] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, *JAX: composable transformations of Python+NumPy programs*, 2018.

[21] R. H. Byrd, G. M. Chin, W. Neveitt, and J. Nocedal, *On the use of stochastic Hessian information in optimization methods for machine learning*, SIAM Journal on Optimization, 21 (2011), pp. 977–995.

[22] R. H. Byrd, G. M. Chin, J. Nocedal, and Y. Wu, *Sample size selection in optimization methods for machine learning*, Mathematical programming, 134 (2012), pp. 127–155.

[23] R. H. Byrd, J. Nocedal, and F. Oztoprak, *An inexact successive quadratic approximation method for L-1 regularized optimization*, Mathematical Programming, 157 (2016), pp. 375–396.

[24] L. Cao, T. O'Leary-Roseberry, and O. Ghattas, *Derivative-informed neural operator acceleration of geometric MCMC for infinite-dimensional Bayesian inverse problems*, arXiv preprint arXiv:2403.08220, (2024).

[25] R. G. Carter, *On the global convergence of trust region algorithms using inexact gradient information*, SIAM Journal on Numerical Analysis, 28 (1991), pp. 251–265.

[26] C. Cartis and K. Scheinberg, *Global convergence rate analysis of unconstrained optimization methods based on probabilistic models*, Mathematical Programming, 169 (2018), pp. 337–375.

[27] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, ACM transactions on intelligent systems and technology (TIST), 2 (2011), pp. 1–27.

[28] P. H. Chen and C.-j. Hsieh, *A comparison of second-order methods for deep convolutional neural networks*, (2018).

[29] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton methods*, SIAM Journal on Numerical analysis, 19 (1982), pp. 400–408.

[30] M. Derezinski, J. Lacotte, M. Pilanci, and M. W. Mahoney, *Newton-LESS: Sparsification without trade-offs for the sketched Newton update*, Advances in Neural Information Processing Systems, 34 (2021), pp. 2835–2847.

[31] P. Dharangutte and C. Musco, *A tight analysis of Hutchinson's diagonal estimator*, in Symposium on Simplicity in Algorithms (SOSA), SIAM, 2023, pp. 353–364.

[32] J. Duchi, E. Hazan, and Y. Singer, *Adaptive subgradient methods for online learning and stochastic optimization.*, Journal of machine learning research, 12 (2011).

[33] S. C. Eisenstat and H. F. Walker, *Choosing the forcing terms in an inexact Newton method*, SIAM Journal on Scientific Computing, 17 (1996), pp. 16–32.

[34] M. A. Erdogdu and A. Montanari, *Convergence rates of sub-sampled Newton methods*, Advances in Neural Information Processing Systems, 28 (2015).

[35] M. P. Friedlander and M. Schmidt, *Hybrid deterministic-stochastic methods for data fitting*, SIAM Journal on Scientific Computing, 34 (2012), pp. A1380–A1405.

[36] A. Gholami, Z. Yao, S. Kim, C. Hooper, M. W. Mahoney, and K. Keutzer, *Ai and memory wall*, IEEE Micro, (2024).

[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016. `http://www.deeplearningbook.org`.

[38] R. Gower, D. Kovalev, F. Lieder, and P. Richtárik, *RSN: randomized subspace Newton*, Advances in Neural Information Processing Systems, 32 (2019).

[39] V. Gupta, S. Kadhe, T. Courtade, M. W. Mahoney, and K. Ramchandran, *Oversketched Newton: Fast convex optimization for serverless systems*, in 2020 IEEE International Conference on Big Data (Big Data), IEEE, 2020, pp. 288–297.

[40] N. Halko, P.-G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM review, 53 (2011), pp. 217–288.

[41] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[42] D. Hendrycks and K. Gimpel, *Gaussian error linear units (gelus)*, arXiv preprint arXiv:1606.08415, (2016).

[43] J. S. Hesthaven and S. Ubbiali, *Non-intrusive reduced order modeling of nonlinear problems using neural networks*, Journal of Computational Physics, 363 (2018), pp. 55–78.

[44] G. Hinton, N. Srivastava, and K. Swersky, *Neural networks for machine learning lecture 6a overview of mini-batch gradient descent*, Cited on, 14 (2012), p. 2.

[45] D. Z. Huang, N. H. Nelsen, and M. Trautner, *An operator learning perspective on parameter-to-observable maps*, arXiv preprint arXiv:2402.06031, (2024).

[46] R. Jiang, M. Dereziński, and A. Mokhtari, *Stochastic Newton Proximal Extragradient Method*, arXiv preprint arXiv:2406.01478, (2024).

[47] D. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in International Conference on Learning Representations (ICLR), San Diega, CA, USA, 2015.

[48] D. P. Kouri and A. Shapiro, *Optimization of PDEs with uncertain inputs*, Frontiers in PDE-constrained optimization, (2018), pp. 41–81.

[49] N. B. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. M. Stuart, and A. Anandkumar, *Neural Operator: Learning Maps Between Function Spaces With Applications to PDEs.*, J. Mach. Learn. Res., 24 (2023), pp. 1–97.

[50] A. Krizhevsky, G. Hinton, et al., *Learning multiple layers of features from tiny images*, (2009).

[51] J. Lacotte, Y. Wang, and M. Pilanci, *Adaptive Newton sketch: Linear-time optimization with quadratic convergence and effective hessian dimensionality*, in International Conference on Machine Learning, PMLR, 2021, pp. 5926–5936.

[52] J. D. Lee, Y. Sun, and M. A. Saunders, *Proximal Newton-type methods for minimizing composite functions*, SIAM Journal on Optimization, 24 (2014), pp. 1420–1443.

[53] J. Levitt and P.-G. Martinsson, *Linear-complexity black-box randomized compression of rank-structured matrices*, SIAM Journal on Scientific Computing, 46 (2024), pp. A1747–A1763.

[54] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, *Fourier neural operator for parametric partial differential equations*, International Conference on Learning Representations, (2021).

[55] H. Liu, Z. Li, D. Hall, P. Liang, and T. Ma, *Sophia: A scalable stochastic second-order optimizer for language model pre-training*, arXiv preprint arXiv:2305.14342, (2023).

[56] I. Loshchilov and F. Hutter, *Decoupled weight decay regularization*, in International Conference on Learning Representations, 2019.

[57] L. Lu, P. Jin, G. Pang, and G. E. Karniadakis, *DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators*, Nature Machine Intelligence, (2021).

[58] D. Luo, P. Chen, T. O'Leary-Roseberry, U. Villa, and O. Ghattas, *SOUPy: Stochastic PDE-constrained optimization under high-dimensional uncertainty in Python*, Journal of Open Source Software, 9 (2024), p. 6101.

[59] D. Luo, T. O'Leary-Roseberry, P. Chen, and O. Ghattas, *Efficient PDE-Constrained optimization under high-dimensional uncertainty using derivative-informed neural operators*, arXiv preprint arXiv:2305.20053, (2023).

[60] J. Martens et al., *Deep learning via Hessian-free optimization.*, in ICML, vol. 27, 2010, pp. 735–742.

[61] J. Martens and R. Grosse, *Optimizing neural networks with kronecker-factored approximate curvature*, in International conference on machine learning, PMLR, 2015, pp. 2408–2417.

[62] P.-G. Martinsson and J. A. Tropp, *Randomized numerical linear algebra: Foundations and algorithms*, Acta Numerica, 29 (2020), pp. 403–572.

[63] A. Mokhtari and A. Ribeiro, *Global convergence of online limited memory bfgs*, The Journal of Machine Learning Research, 16 (2015), pp. 3151–3181.

[64] S. Na, M. Dereziński, and M. W. Mahoney, *Hessian averaging in stochastic Newton methods achieves superlinear convergence*, Mathematical Programming, 201 (2023), pp. 473–520.

[65] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer, 1999.

[66] T. O'Leary-Roseberry, *hessianaveraging: Hessian-Averaged Newton Methods for Stochastic Optimization in jax*, 2024.

[67] T. O'Leary-Roseberry, P. Chen, U. Villa, and O. Ghattas, *Derivative-Informed Neural Operator: An efficient framework for high-dimensional parametric derivative learning*, Journal of Computational Physics, 496 (2024), p. 112555.

[68] T. O'Leary-Roseberry, X. Du, A. Chaudhuri, J. R. Martins, K. Willcox, and O. Ghattas, *Learning high-dimensional parametric maps via reduced basis adaptive residual networks*, Computer Methods in Applied Mechanics and Engineering, 402 (2022), p. 115730.

[69] T. O'Leary-Roseberry, U. Villa, P. Chen, and O. Ghattas, *Derivative-informed projected neural networks for high-dimensional parametric maps governed by PDEs*, Computer Methods in Applied Mechanics and Engineering, 388 (2022), p. 114199.

[70] C. Paquette and K. Scheinberg, *A stochastic line search method with expected complexity analysis*, SIAM Journal on Optimization, 30 (2020), pp. 349–376.

[71] R. Pasupathy, P. Glynn, S. Ghosh, and F. S. Hashemi, *On sampling rates in simulation-based recursions*, SIAM Journal on Optimization, 28 (2018), pp. 45–73.

[72] J. Pathak, S. Subramanian, P. Harrington, S. Raja, A. Chattopadhyay, M. Mardani, T. Kurth, D. Hall, Z. Li, K. Azizzadenesheli, et al., *Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators*, arXiv preprint arXiv:2202.11214, (2022).

[73] M. Pilanci and M. J. Wainwright, *Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence*, SIAM Journal on Optimization, 27 (2017), pp. 205–245.

[74] H. Robbins and D. Siegmund, *Boundary crossing probabilities for the Wiener process and sample sums*, The Annals of Mathematical Statistics, (1970), pp. 1410–1429.

[75] F. Roosta-Khorasani and M. W. Mahoney, *Sub-sampled Newton methods*, Mathematical Programming, 174 (2019), pp. 293–326.

[76] J. Ruseckas, *CIFAR10 classification using Flax.* https://juliusruseckas.github.io/ml/flax-cifar10.html, 2024.

[77] L. Ruthotto and E. Haber, *An introduction to deep generative modeling*, GAMM-Mitteilungen, 44 (2021), p. e202100008.

[78] N. N. Schraudolph, J. Yu, and S. Günter, *A stochastic quasi-newton method for online convex optimization*, in Artificial intelligence and statistics, PMLR, 2007, pp. 436–443.

[79] S. Shashaani, F. S. Hashemi, and R. Pasupathy, *Astro-df: A class of adaptive sampling trust-region algorithms for derivative-free stochastic optimization*, SIAM Journal on Optimization, 28 (2018), pp. 3145–3176.

[80] B. Taylor, *Methodus incrementorum directa & inversa*, Inny, 1717.

[81] J. Tropp, *Freedman's inequality for matrix martingales*, (2011).

[82] V. Vapnik, *Principles of risk minimization for learning theory*, Advances in neural information processing systems, 4 (1991).

[83] C.-C. Wang, C.-H. Huang, and C.-J. Lin, *Subsampled Hessian Newton methods for supervised learning*, Neural computation, 27 (2015), pp. 1766–1795.

[84] J. Wang and T. Zhang, *Utilizing second order information in minibatch stochastic variance reduced proximal iterations*, Journal of Machine Learning Research, 20 (2019), pp. 1–56.

[85] Y. Xie, R. Bollapragada, R. Byrd, and J. Nocedal, *Constrained and composite optimization via adaptive sampling methods*, IMA Journal of Numerical Analysis, 44 (2024), pp. 680–709.

[86] G. Xing, J. Gu, and X. Xiao, *Convergence analysis of a subsampled Levenberg-Marquardt algorithm*, Operations Research Letters, 51 (2023), pp. 379–384.

[87] P. Xu, F. Roosta, and M. W. Mahoney, *Newton-type methods for non-convex optimization under inexact Hessian information*, Mathematical Programming, 184 (2020), pp. 35–70.

[88] ——, *Second-order optimization for non-convex machine learning: An empirical study*, in Proceedings of the 2020 SIAM International Conference on Data Mining, SIAM, 2020, pp. 199–207.

[89] P. Xu, J. Yang, F. Roosta, C. Ré, and M. W. Mahoney, *Sub-sampled Newton methods with non-uniform sampling*, Advances in Neural Information Processing Systems, 29 (2016).

[90] Z. Yao, A. Gholami, S. Shen, M. Mustafa, K. Keutzer, and M. Mahoney, *Adahessian: An adaptive second order optimizer for machine learning*, in proceedings of the AAAI conference on artificial intelligence, vol. 35, 2021, pp. 10665–10673.

[91] A. Yesypenko and P.-G. Martinsson, *Randomized strong recursive skeletonization: Simultaneous compression and factorization of $\mathcal{H}$-matrices in the black-box setting*, arXiv preprint arXiv:2311.01451, (2023).

[92] M. Yousefi and A. Martinez, *On the efficiency of stochastic quasi-Newton methods for deep learning*, arXiv preprint arXiv:2205.09121, (2022).

[93] O. Zahm, P. G. Constantine, C. Prieur, and Y. M. Marzouk, *Gradient-based dimension reduction of multivariate vector-valued functions*, SIAM Journal on Scientific Computing, 42 (2020), pp. A534–A558.

[94] S. Zampini, U. Zerbinati, G. Turkyyiah, and D. Keyes, *PETScML: Second-order solvers for training regression problems in Scientific Machine Learning*, in Proceedings of the Platform for Advanced Scientific Computing Conference, 2024, pp. 1–12.

9 Appendices

9.1 Deterministic Sampling Bounds

Given Assumption 2.1, we have the following bound used in Lemma 2.1,

$$\|g_k - \nabla f(w_k)\|^2 \leqslant 4\left(\frac{N-|X_k|}{N}\right)(\beta_{1,g}\|\nabla f(w_k)\|^2 + \beta_{2,g}). \tag{9.1}$$

The bound is stated in the beginning of Section 3.1 in [35], we restate it here for completeness of our presentation:

$$\begin{aligned}
\|g_k - \nabla f(w_k)\|^2 &= \left\| \left(\frac{N-|X_k|}{N|X_k|}\right) \sum_{i\in X_k} \nabla F_i(w_k) - \frac{1}{N} \sum_{i\in[N]\setminus X_k} \nabla F_i(w_k) \right\|^2 \\
&\leqslant \left[\left(\frac{N-|X_k|}{N|X_k|}\right) \left\| \sum_{i\in X_k} \nabla F_i(w_k) \right\| + \frac{1}{N} \left\| \sum_{i\in[N]\setminus X_k} \nabla F_i(w_k) \right\| \right]^2 \\
&\leqslant \left[\left(\frac{N-|X_k|}{N|X_k|}\right) \sum_{i\in X_k} \|\nabla F_i(w_k)\| + \frac{1}{N} \sum_{i\in[N]\setminus X_k} \|\nabla F_i(w_k)\| \right]^2 \\
&\leqslant 4\left(\frac{N-|X_k|}{N}\right)(\beta_{1,g}\|\nabla f(w_k)\|^2 + \beta_{2,g}),
\end{aligned} \tag{9.2}$$

where $[N] := \{1, 2, \cdots, N\}$.

Similarly, given Assumption 3.5, we have the following bound used in Lemma 3.8,

$$\|H_k - \nabla^2 f(w_k)\|^2 \leqslant 4\left(\frac{N-|S_k|}{N}\right)(\beta_{1,H}\|\nabla f(w_k)\|^2 + \beta_{2,H}). \tag{9.3}$$

The bound is stated in the beginning of Section 3.1 in [35], we restate it here for completeness of our presentation:

$$\begin{aligned}
\|H_k - \nabla^2 f(w_k)\|^2 &= \left\| \left(\frac{N-|S_k|}{N|S_k|}\right) \sum_{i\in S_k} \nabla^2 F_i(w_k) - \frac{1}{N} \sum_{i\in[N]\setminus S_k} \nabla^2 F_i(w_k) \right\|^2 \\
&\leqslant \left[\left(\frac{N-|S_k|}{N|S_k|}\right) \left\| \sum_{i\in S_k} \nabla^2 F_i(w_k) \right\| + \frac{1}{N} \left\| \sum_{i\in[N]\setminus S_k} \nabla F_i(w_k) \right\| \right]^2 \\
&\leqslant \left[\left(\frac{N-|S_k|}{N|S_k|}\right) \sum_{i\in S_k} \|\nabla^2 F_i(w_k)\| + \frac{1}{N} \sum_{i\in[N]\setminus S_k} \|\nabla^2 F_i(w_k)\| \right]^2 \\
&\leqslant 4\left(\frac{N-|S_k|}{N}\right)(\beta_{1,H}\|\nabla^2 f(w_k)\|^2 + \beta_{2,H}).
\end{aligned} \tag{9.4}$$

9.2 Numerical Results

In this appendix we overview additional details of the numerical results. Most implementation detail questions can be answered by reviewing the code in the accompanying repository `github.com/`

`tomoleary/hessianaveraging` [66]. The code was implemented in `jax` [20], and the numerical results were run on servers with NVIDIA A100 and L40S GPU. Access to large memory GPU may be required to run some of the results in the manuscript.

9.2.1 Stochastic Quadratic Minimization

In section 6.2 we consider a subsampled quadratic minimization problem:

$$\text{Subsampled quadratic:} \qquad \min_w f(w) = \mathbb{E}_{P_A,P_b}\left[\|P_A A w - P_b b\|^2\right], \tag{9.5}$$

where P_A, and P_b randomly zero out a certain number of entries in A and b respectively; this problem is a simple analogue to empirical risk minimization over a dataset.

In order to investigate adaptive gradient sampling, we use different P_A, P_b for the Hessian and gradient calculations, and in order to satisfy the norm test we reduce the number of zero entries in P_A, P_b in order to satisfy the norm test. The true A matrix is taken to be a positive definite matrix with spectrum $\lambda_i = 10^{-4} + (0.1i)^{\frac{3}{2}}$, with $d = 100$. In this case the Hessian condition number $\kappa(A^T A) \approx 10^6$, which gives a restrictive Lipschitz condition for gradient descent. When employing the norm test, we take $\theta_k = 0.5$ to be constant, so we are limited to the (fast) linear local convergence regime of our theory.

9.2.2 CIFAR[10,100] classification with ResNets

In section 6.4 we investigate image classification with CIFAR[10,100] datasets. We utilize a ResNet architecture based on [76], similar but not identical to those utilized in [41]. Our results are able to achieve similar accuracies to typical ResNet architectures, but do not reference established benchmarks. All training runs for a given seed are run from the same initial guess. We use learning rate schedulers that reduce the learning rate by a factor of four every 25% of epochs in order to obtain more practical performance. The architectures used for CIFAR10 and CIFAR100 are nearly identical, and only differ in the final layers, which map to $\mathbb{R}^{10}, \mathbb{R}^{100}$, respectively. The weight dimensions are correspondingly $d = 11,200,882$ and $d = 11,247,052$. The details of the network architecture are much easier to define in code, and are taken from [76], so we refer the interested reader to the repository [66] for the implementation, and encourage them to run the code. Access to GPUs with large RAM will be necessary, all results we used were run on NVIDIA A100 (40 and 80 GB) and L40S (48 GB). For all cases except one we report averages over `jax` seeds 0 and 1.

(†): Note from Table 5: in the case of the CIFAR100 Dan with $\alpha = 0.01$ all methods suffered from early iteration divergence using seed 0. Averaged over seeds 0 and 1 the corresponding accuracies of this method goes from the reported $(71.18, 72.10, 71.57)$ down to $(48.68, 62.52, 64.03)$. This result is an anomaly due to unlucky initial guess and/or sampling order, which was resolved by running a different seed, however we document this issue for full transparency and reproducibility.

9.2.3 Parametric PDE Learning Numerical Details

Note that the loss function in (6.5c) requires one derivative of the neural network, and the Hessian-vector products used in all optimization problem require two more. For this reason with consider C^3 continuous activation functions, and thus uze the Gaussian error linear unit, `gelu` activation function. The networks are generic feedforward multi-layer perceptrons that have inputs of 200, followed by five laters with dimension 400, which then gets reduced to the output, which has dimension 50.

9.3 A Note on (Derivative-Informed) Parametric PDE Learning

We give a brief synopsis of parametric PDE learning, which is of great interest to the authors, in order to give additional context to the numerical results in section 6.5. Learning parametric PDE

maps via neural network representations has become a research topic of great interest in recent years. In the typical setup there is a parameter function $x \in \mathcal{X}$, which is mapped out to an output $y \in \mathcal{Y}$ implicitly, through the solution of an expensive-to-evaluate PDE model; this map is then $x \mapsto y(x)$. The parametric PDE learning problem is typically motivated through computationally expensive tasks such as Bayesian inverse problems, optimization problems under uncertainty, optimal design, optimal experimental design, rare event estimation, all of which have very large computational costs through iteration and sampling complexities. The goal of the parametric PDE learning problem is to construct a surrogate for the parametric PDE maps showing up in the aforementioned tasks that can be substituted for direct forward simulation within these algorithms, and specifically to do so at a lower end-to-end cost, including accounting for the costs of sampling the PDE map to obtain training data.

The spaces $\mathcal{X}, \mathcal{Y}$ are generally separable Banach spaces, but for the remainder of this presentation we will assume them to be separable Hilbert spaces. The inputs $x \in \mathcal{X}$ are equipped with an input distribution π. We seek to construct and train a neural network approximation $y_w(x) \approx y(x)$. In this setting we consider $\mathcal{X}$ to represent an infinite-dimensional space, while $\mathcal{Y}$ is either an infinite-dimensional space (e.g., to represent the PDE state) or a finite-dimensional vector-valued function on the state (as was the case in the numerical results in section 6.5). The so-called neural operator formulation is to formulate both the approximation and the training problem in a function space setting (following the so-called "optimize-then-discretize" approach) [49][1]. A typical formulation is in the parametric Bochner space $L^2_\pi = L^2(\mathcal{X}, \pi; \mathcal{Y})$, i.e., to formulate the following optimization problem

$$\min_w \left(\|y - y_w\|^2_{L^2_\pi} = \mathbb{E}_\pi \left[\|y - y_w\|^2_{\mathcal{Y}} \right] \right). \tag{9.6}$$

By first formulating the neural network training problem in the function space setting one can derive neural network architectures that respect the continuum limit of the PDE map, and lead to efficient statistical learning formulations. Popular examples of this general approach include PCANet, Fourier Neural Operator (FNO) and DeepONet [43, 49, 54, 57]. These architectural representations utilize appropriate basis representations (e.g., proper orthogonal decomposition (POD), Fourier basis etc.) that allow for efficient finite dimensional approximations that have approximation properties independent of the discretization dimension of the problem. In the numerical results we considered we utilize an architecture that restricts the input function x to the dominant eigenfunctions of the expected sensitivity operator. For example when the inputs are distributed with Gaussian measure, $x \in \mathcal{N}(\overline{x}, \mathcal{C}_x)$, we compute the dominant eigenfunctions ψ_i of the following generalized eigenvalue problem for the eigenpairs λ_i, ψ_i with $\lambda_i \geqslant \lambda_j$ for $i < j$:

$$\mathbb{E}_\pi[D_x y^* D_x y]\psi_i = \lambda_i \mathcal{C}_x^{-1} \psi_i, \tag{9.7}$$

where $\cdot^*$ denotes the adjoint of $D_x y$. The architectures are referred to as derivative-informed projected neural networks DIPNets [68, 69], and are motivated by the existence of input-reduced approximations y_r of the map with bounds satisfying

$$\|y - y_r \circ \Psi_r \Psi_r^*\|^2_{L^2_\pi} \leqslant \sum_{i \geqslant r} \lambda_i. \tag{9.8}$$

The form of y_r is a conditional expectation ridge function that marginalizes out the orthogonal complement to the subspace spanned matrix of eigenfunctions $\Psi_r = [\psi_1, \ldots, \psi_r]$, with $r \in \mathbb{N}$. The bound is derived using the Poincaré inequality (in this case for Gaussian measures) [11, 93]. In our numerical results the output dimension was reduced via pointwise evaluation of the PDE state at finite points in the domain (as is relevant to inverse problems), however other dimension strategies such as POD

[1] We note that the term neural operator is typically reserved for circumstances that both x and y represent infinite-dimensional functions, however the general framework is useful in cases where one or both are functions, as it may lead to discretization-dimension independent representations [45].

could have also been employed to similar effect. The empirical risk minimization problem associated with (9.6) in this specific architecture leads to the efficient finite-dimensional reduced basis coefficient learning problem in (6.5a).

Accurately trained neural operators have been deployed to solve complex inference and uncertainty propagation tasks that would have been out of reach when using a traditional forward simulation [72]. However when they are deployed in the context of an optimization problem, the L^2_π formulation is insufficient. Suppose we have an optimization problem of the form:

$$\min_x f(y(x), x), \tag{9.9}$$

which is solved via gradient-based methods. Problems of the form (9.9) include traditional PDE-constrained optimization problems (additionally including uncertainty when f is a risk measure over additional parameters in the PDE system), but also captures other tasks such as variational inference, e.g., evidence-based lower bound optimization (ELBO) [77]. We will show in the following proposition that training only on the function values and not also the derivative (e.g., the L^p_π parametric Bochner space formulations) may be insufficient to ensure accurate gradients when substituting y_w for y. In the following we use D_x to denote the (total) Fréchet derivative of the objective function f with respect to the input function x, while we denote partial derivatives by ∂_x.

Proposition 9.1. *Error bound for parametric PDE gradients (similar to Proposition 3.1 in [59]).*

Supposed the function f has a Lipschitz partial (Fréchet) derivatives w.r.t both x and y with constant L_f, then we have the following bound:

$$\begin{aligned}\|D_x f(y(x), x) - D_x f(y_w(x), x)\| \leqslant \\ L_f(1 + \|D_x y(x)\|) \underbrace{\|y(x) - y_w(x)\|}_{\text{function error}} + \|\partial_y f(y_w(x), x)\| \underbrace{\|D_x y(x) - D_x y_w(x)\|}_{\text{derivative error}}.\end{aligned} \tag{9.10}$$

Proof. We have the following bounds

$$\begin{aligned}\|D_x f(y(x), x) - D_x f(y_w(x), x)\| &\leqslant \|\partial_x f(y(x), x) - \partial_x f(y_w(x), x)\| + \\ &\|\partial_y f(y(x), x) - \partial_y f(y_w(x), x)\| \|D_x y(x)\| + \|\partial_y f(y(x), x)\| \|D_x y(x) - D_x y_w(x)\| \\ &= L_f(1 + \|D_x y(x)\|)\|y(x) - y_w(x)\| + \|\partial_y f(y_w(x), x)\| \|D_x y(x) - D_x y_w(x)\|.\end{aligned} \tag{9.11}$$

□

This bound shows that the derivative of the PDE map also needs to be controlled in addition to the function error, in order to obtain accurate gradients, which are required for the solution of optimization problems of the form (9.9). For this reason the derivative-informed operator learning formulation was introduced in [67], which proposes the operator learning in the parametric Sobolev space $H^1_\pi = H^1(\mathcal{X}, \pi; \mathcal{Y})$, i.e. to solve the following optimization problem to train the surrogate:

$$\min_w \left(\|y - y_w\|^2_{H^1_\pi} = \mathbb{E}_\pi \left[\|y - y_w\|^2_{\mathcal{Y}} + \|D_x y - D_x y_w\|^2_{HS(\mathcal{X},\mathcal{Y}}} \right] \right), \tag{9.12}$$

where $\|\cdot\|_{HS(\mathcal{X},\mathcal{Y})}$ denotes the Hilbert Schmidt norm for linear operators $A : \mathcal{X} \to \mathcal{Y}$. Utilizing the linear reduced basis architecture discussed above the empirical risk minimization analogue of (9.12) takes the form of the efficient reduced basis coefficient derivative learning problem in (6.5c).

Neural operators trained in this formulation are referred to as derivative-informed neural operators (DINOs). They have favorable cost accuracy tradeoff over traditional formulations for learning the function in L^2_π; this includes the sampling costs for all PDE solves including the derivative computations

using adjoints and directional sensitivities. This phenomenon is apparent in the numerical results shown in Section 6.5. Additionally, they produce better gradients leading to better PDE-constrained optimization [59], and better (Gauss–Newton) Hessians used in the efficient solution of Bayesian inverse problems [24].