SAPPHIRE: Preconditioned Stochastic Variance Reduction for Faster Large-Scale Statistical Learning*

Jingruo Sun†, Zachary Frangella*, and Madeleine Udell*

Abstract. Regularized empirical risk minimization (rERM) has become important in data-intensive fields such as genomics and advertising, with stochastic gradient methods typically used to solve the largest problems. However, ill-conditioned objectives and non-smooth regularizers undermine the performance of traditional stochastic gradient methods, leading to slow convergence and significant computational costs. To address these challenges, we propose the `SAPPHIRE` (**S**ketching-based **A**pproximations for **P**roximal **P**reconditioning and **H**essian **I**nexactness with Variance-**RE**duced Gradients) algorithm, which integrates sketch-based preconditioning to tackle ill-conditioning and uses a scaled proximal mapping to minimize the non-smooth regularizer. This stochastic variance-reduced algorithm converges globally, and enjoys fast local condition number independent convergence, delivering an efficient and scalable solution for ill-conditioned composite large-scale convex machine learning problems. `SAPPHIRE` can solve sparse large-scale lasso problems with size $10^7 \times 10^6$ in less than a minute. Extensive experiments on lasso and logistic regression demonstrate that `SAPPHIRE` often converges 5 times faster than other commonly used methods such as `Catalyst`, `SAGA`, and `SVRG`. This advantage persists even when the preconditioner is infrequently updated, highlighting its robust and practical effectiveness.

Key words. Stochastic Optimization, Preconditioning, Variance Reduction, Large-scale Learning, Sparsity

AMS subject classifications. 90C15, 90C25, 90C53

1. Introduction.

Modern datasets in science and machine learning are massive in scale. As an example in genetics, whole genome sequencing efforts on large-scale population cohorts like the Million Veterans Program, AllofUS program, and the OurFutureHealth project are expected to collect data from more than millions of individuals on billions of genetic variants. Single-cell sequencing and epigenetic features such as DNA methylation levels, transcription factor binding, gene proximity, and other annotations can further increase the scale of the problem. Naively training a machine learning model on such data leads to an expensive optimization problem whose solution is uninterpretable and often fails to generalize to unseen data. Modern statistics and learning theory provide a solution to this challenge by using *structured regularization* to improve model interpretability and generalization. Mathematically, the optimization problem to solve is a regularized empirical risk minimization (rERM) problem,

$$\text{(rERM)} \qquad \underset{w \in \mathbb{R}^p}{\text{minimize}} \;\; \mathcal{R}(w) := \frac{1}{n} \sum_{i=1}^{n} \ell_i(w) + r(w),$$

*Submitted to the editors June 10th, 2025.

Funding: MU, JS, and ZF gratefully acknowledge support from the National Science Foundation (NSF) Award IIS-2233762, the Office of Naval Research (ONR) Awards N000142212825, N000142412306, and N000142312203, the Alfred P. Sloan Foundation, and from IBM Research as a founding member of Stanford Institute for Human-centered Artificial Intelligence (HAI).

†Department of Management Science and Engineering, Stanford University, CA (jingruo@stanford.edu, zfran@stanford.edu, udell@stanford.edu).

where n is the number of samples, p is the number of features, and $w \in \mathbb{R}^p$ represents the model weights. Here the $\ell_i(w)$'s are smooth loss functions, and $r(w)$ is a possibly non-convex and non-smooth regularizer that encourages a parsimonious solution. Popular regularizers include the l_1-norm, SCAD regularizer, or the indicator function for the l_0-ball. Problem (rERM) models many fundamental problems in machine learning, such as Lasso, elastic-net regression, l_1-logistic regression, dictionary learning, and matrix completion, as well as modern applications such as convex neural networks [40, 17], data models for deep learning [24], and pruned ensembles of trees [33].

Realistic problems in high dimensions n and p are generally ill-conditioned, with a loss whose Hessian eigenvalues span many orders of magnitude [19, Table 2]. Ill-conditioning requires first-order methods like stochastic gradient descent to use a small learning rate to avoid divergence, and hence to suffer from slow convergence. For example, if $\ell(\cdot, w)$ is the loss of a generalized linear model (GLM), the conditioning of (rERM) is controlled by the conditioning of the data matrix X. In large-scale datasets, the features are often highly correlated, so X is approximately low-rank and has a large condition number—possibly larger than the sample size n, leading to a difficult optimization problem in (rERM).

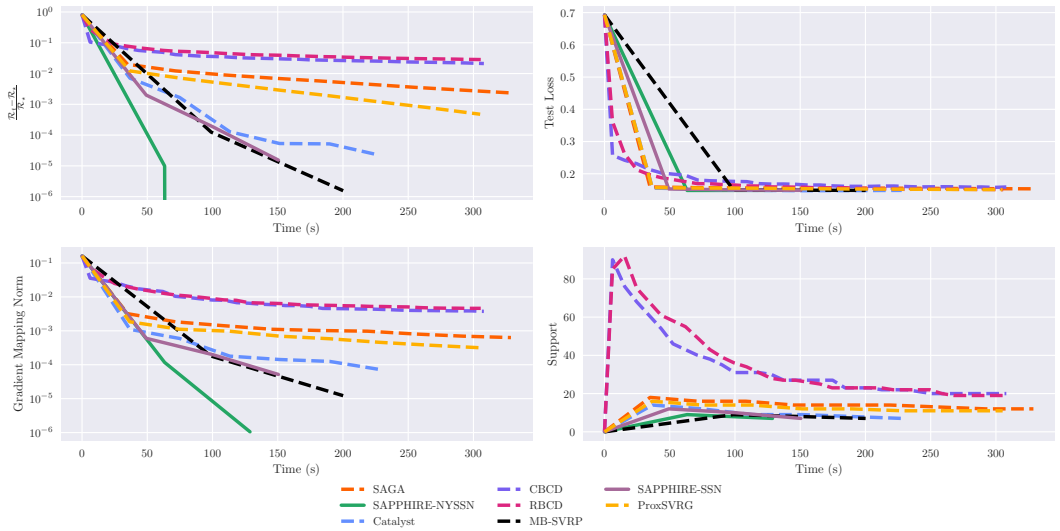

Figure 1. *Showcase experiment of Click Prediction.* `SAPPHIRE` *significantly outperforms competing stochastic optimizers on a large-scale click prediction problem with the avazu dataset* ($n = 12,642,186,\ p = 999,990$).

A traditional way to mitigate ill-conditioning in optimization is to use second-order methods, such as Newton's method or BFGS, which incorporate curvature information. These methods are robust and can achieve local superlinear convergence. While these classical methods do not scale to the big data regime, new stochastic second-order methods developed in the last decade can scale and deliver better practical performance than first-order methods [10, 16, 41, 46, 22, 35, 18]. Indeed, recent work [18] demonstrates that combining second-order information with variance-reduced gradients can yield fast stochastic second-order methods with strong theoretical and practical convergence. However, these methods work best for smooth and (strongly) convex problems, and cannot handle structured regularization with a

non-smooth regularizer, such as the ℓ_1 regularizer in the Lasso problem.

Structured regularization improves both interpretability and generalization. However, its effect on ill-conditioning is more nuanced. On one hand, near convergence, the additional structure can help the algorithm identify a lower-dimensional basis for the solution and reduce the effective dimensionality of the problem. On the other hand, many structured penalties are non-smooth, which complicates algorithmic design and can worsen conditioning. Thus, even with structured regularization, high-dimensional problems ($n, p \gg 1$) still suffer from ill-conditioning.

In this work, we address precisely these computational challenges, using stochastic second-order information to develop an efficient, scalable method that handles both non-smoothness and large-scale, ill-conditioned data. Our algorithm, `SAPPHIRE` (**S**ketching-based **A**pproximations for **P**roximal **P**reconditioning and **H**essian **I**nexactness with Variance-**RE**duced Gradients), is a preconditioned variance-reduced stochastic gradient algorithm that generalizes the approach in [18] to the (non-smooth) regularized problem (rERM). Figure 1 shows the performance of `SAPPHIRE` with two different preconditioners on a large-scale (and hence ill-conditioned) logistic regression problem with an elastic-net penalty. With either preconditioner, `SAPPHIRE` converges significantly faster than competing methods, demonstrating its robustness and efficiency.

1.1. Contributions.

We summarize our contributions as follows:

1. We introduce a robust framework, `SAPPHIRE`, to solve ill-conditioned composite large-scale convex optimization problems using variance reduction that requires only stochastic gradients and stochastic Hessians, and prove convergence of this framework under lazy preconditioner updates.
2. `SAPPHIRE` accesses the non-smooth regularizer through a scaled proximal mapping in the preconditioned norm. While this mapping does not have a closed form, we propose to solve it iteratively using accelerated proximal gradient (APG) algorithm and demonstrate that only a few APG iterations are required.
3. We provide default hyperparameter recommendations and verify they yield excellent performance across a broad testbed of datasets without further data-dependent tuning.
4. We prove that `SAPPHIRE` achieves global linear convergence for strongly convex objectives and global sublinear convergence for convex objectives. We also show that the algorithm converges locally at a linear rate that is independent of the condition number.
5. Through experiments with 28 diverse datasets, we demonstrate that `SAPPHIRE` often converges over 5 times faster than other popular stochastic optimizers on ill-conditioned problems.

1.2. Roadmap.

We organize the paper as follows. Section 2 reviews recent literature, highlighting connections to existing methods and the distinctions of our proposed algorithm. Section 3 proposes the `SAPPHIRE` algorithm formally and elaborates on its core components of sketch-based preconditioning and scaled proximal mapping. Section 4 establishes comprehensive convergence results for `SAPPHIRE`, covering both global and local convergence with various convexity assumptions. Section 5 demonstrates the superior performance of the algorithm over popular tuned stochastic optimizers through extensive numerical experiments.

1.3. Notation. Throughout the paper, $\|\cdot\|$ denotes the Euclidean norm, and denote $\|\cdot\|_A$ as the matrix norm induced by matrix A, where $\|x\|_A = \sqrt{x^\top Ax}$. For a positive definite matrix A, we write $A \succeq 0$. The Loewner order is denoted by $\preceq$, where $A \preceq B$ if the matrix $B - A \succeq 0$. Given a positive definite matrix $A \in \mathbb{R}^{p\times p}$, its eigenvalues in descending order are written as $\lambda_1(A) \geq \lambda_2(A) \geq \cdots \geq \lambda_p(A)$. We denote the smoothness constant of $L(w) = \frac{1}{n}\sum_{i=1}^n \ell_i(w)$ by L. For each $\ell_i(w)$ in (rERM), we denote the smoothness constant by L_i and define $L_{\max} = \max_{i\in[n]} L_i$. If $L(w)$ is μ-strongly convex we denote its condition number by $\kappa = L/\mu$, and define $\kappa_{\max} = L_{\max}/\mu$. The condition number of symmetric positive definite matirx A is defined as $\kappa(A) = \lambda_1(A)/\lambda_p(A)$. For any scalar $\beta > 0$, we define the effective dimension $d_{\text{eff}}^{\beta}(A) = \operatorname{tr}(A(A+\beta I)^{-1})$, which provides a smoothed measure of eigenvalues greater than or equal to β.

2. Related Work. Here we review prior work on stochastic second-order methods, with particular emphasis on those developed for convex optimization problems, which is the main focus of this paper.

Variance-reduced stochastic first-order methods for finite sum minimization. Due to the massive size of contemporary machine learning datasets, much of the research in the past decade has focused on developing efficient algorithms that only require a stochastic first-order oracle. The most successful of these algorithms are those that employ *variance reduction*, which results in the variance of the gradient approaching zero as the iterates near an optimum [26]. This technique yields global sublinear and linear convergence when the objective is convex and strongly convex, respectively. Popular variance-reduced optimizers include `SAGA` [13], `ProxSVRG` [52], `Catalyst` [32], and `Katyusha` [1]. These algorithms are also popular in practice for solving the empirical risk minimization problem (rERM). Indeed, the popular software package `scikit-learn` employs `SAGA` as the default stochastic gradient-based solver for problems such as logistic regression. In the non-convex case, convergence to approximate stationary points has been established for many variants of these algorithms [4, 44, 25, 39, 2]. The assumptions underlying these theoretical guarantees typically prescribe that these methods use a minimal learning rate that goes to zero with n. However, in practice, these algorithms are often run with a fixed learning rate as though the objective were convex, as this yields better performance [25, 39].

Stochastic second-order methods for finite sum minimization. Stochastic first-order methods suffer in the face of ill-conditioning. To address this limitation, many authors have worked on stochastic second-order algorithms capable of scaling to large-scale machine learning problems. We classify these schemes by their target problems and methods used to compute gradients and Hessian. We summarize these results in Table 1. Some methods require exact gradients at every iteration; some require only stochastic gradients; and some ("snapshot") require stochastic gradients and occasional exact gradients. All methods in the table require only stochastic samples of the Hessian. Many assume interpolation ($\inf_w \mathcal{R}(w) = 0$) to prove convergence to the global optimum.

These work vary in how they use second-order information: some directly apply the inverse of the subsampled Hessian to the stochastic gradient [46, 8, 35], or they use the subsampled Hessian-vector product to update the preconditioner rather than using the difference between two stochastic gradients [36, 9, 35]. However, the theory underlying these methods requires

Table 1
Stochastic Second-Order Methods in ERM Literature

Papers	Loss	Regularizer	Gradient	Fixed batchsize	Interpolation
[10, 16, 6, 41, 22, 56, 12, 57]	Convex	None	Exact	No	No
[36, 9, 46, 8, 35]	Convex	None	Stochastic	No	Yes
[14, 19, 18, 21, 51, 29]	Strongly convex	Smooth	Snapshot	Yes	No
This paper	Convex	Non-smooth	Snapshot	Yes	No

large or growing gradient batch sizes [46, 9, 8], periodic full gradient computation [36], or interpolation [35], which are unrealistic assumptions for large-scale convex problems. Further, many of these methods lack practical guidelines for setting hyperparameters such as batch sizes and learning rate, leading to the same tuning issues that plague stochastic first-order methods.

Recent work has developed more practical stochastic second-order algorithms that use variance-reduction and stochastic second-order information to improve convergence [14, 19, 18, 21]. The `PROMISE` framework in [18] leads to globally linearly convergent algorithms with *constant* gradient batch sizes and comes with theoretically-motivated default hyperparameter settings that outperform tuned stochastic first-order methods.

However, most of these improved algorithms still assume smoothness and strong convexity to show their convergence results. For instance, `SVRN` [14, 21] and `PROMISE` [18] require smooth and strongly convex objectives. `SketchySGD` [19] can be used in the convex case but only converges to a noise ball around the optimum. [51] and [29] can handle composite problems with a non-smooth regularizer in practice, but their convergence analyses are restricted to smooth and strongly convex problems. Therefore, `SAPPHIRE` fills a significant gap in the literature by providing condition-number-free linear convergence on convex composite problems (rERM).

Provably convergent stochastic second-order methods for smooth non-convex finite sum minimization have been developed. Most methods are based on using a randomized approximation to the Hessian (via subsampling or sketching) together with cubic regularization [28, 48, 53], Newton-CG [55, 43], or trust region methods [7, 54, 45] to (for example) guarantee convergence to a local minimum. However, many of these methods require solving a challenging subproblem at each iteration, such as a cubic Newton step or a trust-region problem. Consequently, these methods are often slower than stochastic first-order methods despite converging in fewer iterations.

2.1. Comparison with `SAPPHIRE`. Table 2 positions `SAPPHIRE` relative to existing work on state-of-the-art stochastic second-order optimizers for solving instances of (rERM) with a loss that depends only on the inner product of the parameters and the data, a model class that includes all (regularized) generalized linear models

$$\frac{1}{n}\sum_{i=1}^{n}\ell(x_i^T w) + \frac{\nu}{2}\|w\|^2 + r(w), \tag{2.1}$$

where $x_i \in \mathbb{R}^p$ is the ith row of data matrix X. A restriction to l_2-regularized GLMs makes comparison to previous work as straightforward as possible, as `MB-SVRP`, `PROMISE`, and `Proximal Subsampled Newton` restrict their analysis to GLMs. The table compares meth-

ods based on the properties they require to achieve condition number-free local convergence[1]. Table 2 considers whether the method allows for a non-trivial convex regularizer $r(w)$, its required gradient batchsize, and the size of the neighborhood of local convergence.

Table 2

SAPPHIRE vs. State-of-the-art competitors for solving (2.1). Of the methods in the table, SAPPHIRE is the only variance-reduced stochastic gradient algorithm whose local convergence guarantees allow for a non-trivial convex regularizer. SAPPHIRE also has the best gradient batchsize requirement without requiring a smaller neighborhood of local convergence.

Method	Regularizer	Gradient Batchsize	Radius of Local Convergence
SAPPHIRE (Algorithm 3.1)	Convex and Proxable	$\tilde{\mathcal{O}}\left(\tau_\star^\nu\right)$	$\mathcal{O}\left(\frac{\nu^{3/2}}{M}\right)$
Proximal SSN [29]	Convex and Proxable	n	$\mathcal{O}\left(\frac{\nu}{M}\right)$
MB-SVRP [51]	None	$\mathcal{O}\left(\chi^\nu(\nabla^2 L(w_\star))d_{\text{eff}}^\nu(\nabla^2 L(w_\star))\kappa_{\max}^{1/3}\right)$	$\mathcal{O}\left(\frac{\nu^4}{L_{\max}^2 M}\right)$
SVRN [14, 21]	None	$\tilde{\mathcal{O}}(\kappa_{\max})$	$\mathcal{O}\left(\frac{\nu^{3/2}}{M}\right)$
SketchySVRG [18]	None	$\tilde{\mathcal{O}}\left(\tau_\star^\nu\right)$	$\mathcal{O}\left(\frac{\nu^{3/2}}{M}\right)$

3. SAPPHIRE: A Fast Algorithm for Large-Scale Statistical Learning.

In this section, we formally introduce the SAPPHIRE algorithm.

3.1. SAPPHIRE algorithm.

SAPPHIRE is a preconditioned variance-reduced stochastic gradient algorithm based on the classic ProxSVRG algorithm from [52]. The most significant innovation of SAPPHIRE is the design of an effective preconditioner for the problem. Preconditioning is critical to problems with large-scale data, often improving the runtime by orders of magnitude. However, preconditioning complicates the computation of the proximal operator.

In the following sections, we discuss how to construct the preconditioner, efficiently solve the associated scaled proximal mapping, and set algorithmic hyperparameters.

3.2. Efficient preconditioning.

Preconditioning is a powerful technique to accelerate the convergence of optimization algorithms on ill-conditioned problems. A good preconditioner must effectively approximate the local Hessian while being fast to compute and to invert.

Classic methods from optimization, like Newton's method and BFGS, precondition the gradient using the (approximate) inverse Hessian. As a result, these methods enjoy fast local convergence rates that are independent of the condition number. Unfortunately, the Hessian or Hessian approximation used by these methods is expensive to compute and to invert for large-scale problems. These methods fail to scale to the problems commonly encountered in machine learning. Recent work [16, 46, 18] has shown in the smooth non-composite, effective preconditioners can be constructed only using a small fraction of the data, reducing the cost of preconditioning substantially. SAPPHIRE adopts the Subsampled Newton and the Nyström Subsampled Newton preconditioners, motivated by the authors' prior work [18].

3.2.1. Subsampled Newton Preconditioner.

The subsampled Newton (SSN) preconditioner first introduced in [46], approximates the Hessian matrix $\nabla^2 L(w) \in \mathbb{R}^{p\times p}$ of the smooth

[1] We compare based on local and not global convergence as global convergence analyses are often looser and sometimes absent from previous work.

Algorithm 3.1 SAPPHIRE

Input: starting point w_0, gradient and Hessian batch S_h, S_g with size b_h, b_g, preconditioner P, preconditioner update times $\mathcal{U}$, learning rate $\eta^{(0)}$, snapshot update frequency m

Initialize: snapshot $\tilde{w} = \tilde{w}_0$

for $s = 0, 1, \ldots$ **do**

  Compute full gradient $\bar{g} = \nabla L(\tilde{w})$

  Set $w_0 = \tilde{w}$

  for $k = 0, 1, \ldots m-1$ **do**

    if $ms + k \in \mathcal{U}$ **then**

      Sample batch S_h to obtain indices for $\widehat{\nabla}^2 L(w_k^{(s)})$

      Compute preconditioner $P_k^{(s)}$: SSN (3.1) or NySSN (3.3) with $\widehat{\nabla}^2 L(w_k^{(s)})$

    end if

    Sample stochastic gradient batch S_g

    Compute estimator $\widehat{\nabla} L(w_k^{(s)}) = \frac{1}{b_g} \sum_{i \in S_g} \nabla \ell_i(w_k^{(s)})$ and $\widehat{\nabla} L(\tilde{w}) = \frac{1}{b_g} \sum_{i \in S_g} \nabla \ell_i(\tilde{w})$

    Compute $v_k^{(s)} = \widehat{\nabla} L(w_k^{(s)}) - \widehat{\nabla} L(\tilde{w}) + \bar{g}$

    $w_{k+1}^{(s)} = \mathbf{prox}_{\eta r}^{P_k^{(s)}}(w_k^{(s)} - \eta^{(s)} (P_k^{(s)})^{-1} v_k^{(s)})$ ▷ Apply Algorithm 3.2

    Optional:

    Update learning rate via stochastic linesearch ▷ Apply Algorithm SM2.1

$$\eta^{(s+1)} = SLS(\eta^{(s)})$$

  end for

  Option 1: $\tilde{w} = \frac{1}{m} \sum_{k=1}^{m} w_k^{(s)}$ ▷ Update snapshot as average of inner iterates

  Option 2: $\tilde{w} = w_m^{(s)}$ ▷ Update snapshot as last iterate

end for

part of the objective in (rERM) using a subset $S_h \subset \{1, \ldots, n\}$ of the data with batch size $b_h = |S_h|$. The preconditioner is constructed as

$$P = \frac{1}{b_h} \sum_{i \in S_h} \nabla^2 \ell_i(w) + \rho I, \tag{3.1}$$

where $\rho > 0$ is a regularization parameter that mitigates noise in the smaller eigenvalues of this preconditioner.

By using only a subset of the data, this approach significantly reduces computational cost compared to a full computation of the Hessian (as in Newton's method), yet still identifies essential information about the local curvature. To understand the approximation qualities of the SSN preconditioner, we first recall the notion of ρ-Hessian dissimilarity from [19].

Definition 3.1. *Let $L(w)$ be as in (rERM), where each $\ell_i : \mathbb{R}^p \mapsto \mathbb{R}$ is a smooth convex function. Let $\rho \geq 0$ and $w \in \mathbb{R}^p$, then for ρ-Hessian dissimilarity at w is given by*

$$\tau^{\rho}(\nabla^2 L(w)) = \max_{i \in [n]} \lambda_{\max} \left((\nabla^2 L(w) + \rho I)^{-1/2} (\nabla^2 \ell_i(w) + \rho I)(\nabla^2 L(w) + \rho I)^{-1/2} \right).$$

Moreover, given a subset $\mathcal{S}$ of $\mathbb{R}^p$, we define the ρ-maximal Hessian dissimilarity over $\mathcal{S}$ by:

$$\tau_\star^\rho(\mathcal{S}) = \sup_{w\in\mathcal{S}} \tau^\rho(\nabla^2 L(w)).$$

Remark 3.2. When $\mathcal{S} = \mathbb{R}^p$, we will write $\tau_\star^\rho$ for shorthand.

ρ-Hessian dissimilarity measures how much an individual Hessian $\nabla^2\ell_i(w)$ deviates from the average Hessian $\nabla^2 L(w)$. When the $\nabla^2\ell_i(w)$ are relatively similar to each other, the smaller $\tau^\rho(\nabla^2 L(w))$ is—in the extreme case all the $\nabla^2\ell_i(w)$ are the same, $\tau^\rho(\nabla^2 L(w)) = 1$. Conversely, when an outlier $\nabla^2\ell_i(w)$ exists, the dissimilarity can be as large as n. The following lemma from [19] summarizes these facts.

Lemma 3.3. *For any $\rho \geq 0$ and $w \in \mathbb{R}^p$, the following inequalities holds*

$$\tau^\rho(w) \leq \min\left\{n, \frac{M(w)+\rho}{\mu+\rho}\right\},$$

where $M(w) \coloneqq \lambda_{\max}(\nabla^2\ell_i(w))$.

$$\tau_\star^\rho \leq \min\left\{n, \frac{L_{\max}+\rho}{\mu+\rho}\right\}.$$

The ρ-Hessian dissimilarity can be far smaller than the upper bound in Lemma 3.3 suggests. See [19] for more details. This is significant as $\tau^\rho(\nabla^2 L(w))$ controls the sample size required to obtain a non-trivial approximation to the Hessian.

Lemma 3.4. *Let $w \in \mathbb{R}^p$, $\zeta \in (0,1)$ and $\rho > 0$. Construct $\widehat{\nabla}^2 L(w)$ with*

$$b_H = \mathcal{O}\left(\frac{\tau^\rho(\nabla^2 L(w))}{\zeta^2}\log\left(\frac{d_{\text{eff}}^\rho(\nabla^2 L(w))}{\delta}\right)\right).$$

Then, with probability at least $1-\delta$,

$$(1-\zeta)P_{\text{SSN}} \preceq \nabla^2 L(w) + \rho I \preceq (1+\zeta)P_{\text{SSN}}.$$

3.2.2. Nyström Subsampled Newton Preconditioner. `SAPPHIRE` achieves superior performance using a different preconditioner: the Nystrom Subsampled Newton (NySSN) preconditioner introduced in [19, 18]. The Nyström preconditioner computes a low-rank approximation of the Hessian matrix by projecting the subsampled Hessian onto a low-rank subspace in the span of Ω. The Nyström Subsampled Newton preconditioner is given by

$$P = (\widehat{\nabla}^2 L(w)\Omega)(\Omega^\top\widehat{\nabla}^2 L(w)\Omega)^{-1}(\Omega^\top\widehat{\nabla}^2 L(w)) + \rho I \tag{3.2}$$

where $\Omega \in \mathbb{R}^{p\times r}$ is a random test matrix. Typical choices for Ω include standard normal random matrices, randomized trigonometric transforms, and sparse-sign matrices [49, 20].

Constructing the NySSN preconditioner via (3.2) is numerically unreliable due to the presence of the pseudoinverse. Instead we apply the numerically stable procedure from [49] to compute the Nyström approximation: $(\widehat{\nabla}^2 L(w)\Omega)(\Omega^\top\widehat{\nabla}^2 L(w)\Omega)^{-1}(\Omega^\top\widehat{\nabla}^2 L(w))$. The numerically stable procedure is presented in Algorithm SM1.1 in Section SM1. It provides an

approximate low-rank eigendecomposition of $\widehat{\nabla}^2 L(w)$: $\hat{V}\hat{\Lambda}\hat{V}^\top$. Using the stable procedure, the NySSN preconditioner is given by

$$P = \hat{V}\hat{\Lambda}\hat{V}^\top + \rho I. \tag{3.3}$$

The preconditioner and its inverse can be applied to vectors in $\mathcal{O}(pr)$ time and requires $\mathcal{O}(pr)$ storage [19, 18].

This low-rank preconditioner is faster to invert for large-scale problems compared to the SSN preconditioner, especially when b_H is large or the data is dense, and significantly reduces the computational cost of preconditioning. Like the SSN preconditioner, the NySSN preconditioner admits strong theoretical guarantees. We have the following result from [19].

Theoretical guarantees.

Lemma 3.5. *Let $w \in \mathbb{R}^p$, $\rho > 0$, and $\gamma \geq 1$. Construct $\widehat{\nabla}^2 L(w)$ with*

$$b_h = \mathcal{O}\left(\tau^\rho(\nabla^2 L(w)) \log\left(\frac{d_{\text{eff}}^\rho(\nabla^2 L(w))}{\delta}\right)\right)$$

samples and the Nyström approximation with rank $r = \mathcal{O}\left(d_{\text{eff}}^{\gamma\rho}(\widehat{\nabla}^2 L(w)) + \log\left(\frac{1}{\delta}\right)\right)$. Then with probability at least $1 - \delta$,

$$\frac{1}{2\gamma} P_{\text{NySSN}} \preceq \nabla^2 L(w) + \rho I \preceq \frac{3}{2} P_{\text{NySSN}}.$$

3.2.3. Choosing a preconditioner. It is natural to wonder when the SSN preconditioner is preferable to the NySSN preconditioner, and vice versa. A naive first appeal to the theory would suggest that the SSN preconditioner should exhibit superior performance (but perhaps is more expensive to apply), as the NySSN preconditioner truncates the subsampled Hessian, and hence loses information. However, the situation turns out to be much more nuanced in practice. Prior studies [18, 19] have shown that the NySSN preconditioner and SSN preconditioner often perform comparably to each other.

A general comparison of the preconditioners is given in Table 3. In terms of computation cost, the NySSN preconditioner is less expensive to apply and store when the Hessian is dense. Conversely, when the Hessian is sparse, the SSN preconditioner is less expensive to store and can also be faster to apply, however the latter advantage may vanish in highly parallel computing environments.

Table 3
Comparison of Preconditioners

	Construction Cost	Computation Cost	Memory Requirement
SSN	NA	$\mathcal{O}(b_h p)$	$\mathcal{O}(b_h p)$
NySSN	$\mathcal{O}(b_h r p)$	$\mathcal{O}(rp)$	$\mathcal{O}(rp)$

While prior studies have been unable to demonstrate a concrete advantage of one preconditioner over the other, in this paper we observe that the NySSN preconditioner generally outperforms the SSN preconditioner across a wide testbed of problems (see Section 5)—consisting of datasets that range from very dense to very sparse, and vary in size from small and to large. Given these results, and prior findings, we recommend using the NySSN preconditioner.

3.3. Scaled Proximal Mapping. In contrast to `ProxSVRG`, to update the parameters, `SAPPHIRE` must evaluate the scaled proximal mapping:

$$w_{k+1} = \mathbf{prox}^P_{\eta r}(w_k - \eta P^{-1} v_k) := \operatorname*{argmin}_{w \in \mathbb{R}^p} \left\{ r(w) + \frac{1}{2\eta} \|w - P^{-1}\left(w_k - \eta v_k\right)\|_P^2 \right\}$$

$$= \operatorname*{argmin}_{w \in \mathbb{R}^p} \left\{ \eta r(w) + \langle \eta v_k, w - w_k \rangle + \frac{1}{2} \|w - w_k\|_P^2 \right\}. \tag{3.4}$$

Unlike the traditional proximal operator, which often has a closed-form solution, (3.4) must be solved iteratively. For `SAPPHIRE` to be practical, it is essential that (3.4) be solved efficiently. `SAPPHIRE` uses the Accelerated Proximal Gradient (APG) algorithm [5, 37] to solve (3.4), motivated by three factors. The first is that it is easy to apply the preconditioner P to vectors, so computing the gradient of the smooth part of (3.4) is cheap. The second is that we can easily set the learning rate without resorting to line search—the smoothness constant is $\lambda_1(P)+\rho$, which is easy to compute for our preconditioners. The third is that (3.4) is $\lambda_1(P)+\rho$-smooth and ρ-strongly convex and APG converges at the optimal rate of $\tilde{\mathcal{O}}\left(\sqrt{\lambda_1(P)/\rho}\right)$. We present pseudocode for APG applied to (3.4) in Algorithm 3.2.

Algorithm 3.2 Accelerated Proximal Gradient (APG) for solving (3.4).

Input: starting point x_0, preconditioner P, and regularization function r
Initialize: $y_0 = x_0, s_0 = 1$
Set $\alpha = (\lambda_1(P) + \rho)^{-1}$
for $t = 0, 1, \ldots T$ **do**
  Calculate $x_{t+1} = \mathbf{prox}_{\alpha\eta r}\left(y_t - \alpha(\eta v_t + P(x_t - w_k))\right)$
  Set $s_{t+1} = \frac{1}{2}(1 + \sqrt{1 + 4s_t^2})$
  Update $y_{t+1} = x_{t+1} + \frac{s_t - 1}{s_{t+1}}(x_{t+1} - x_t)$
end for

In practice, we find running just twenty iterations of Algorithm 3.2 allows `SAPPHIRE` to achieve fast convergence.

3.4. Hyperparameter recommendations. For the Hessian batchsize and rank, we recommend the values of $b_h = 256$, $r = 10$. We recommend updating the preconditioner every 5 epochs for non-quadratic objectives. For quadratic objectives, the preconditioner update frequency should be infinite, as the Hessian is constant. We recommend using 20 APG iterations for evaluating the scaled proximal mapping in Algorithm 3.1. For the learning rate η, we recommend a default value of $1/4$. This recommendation is inspired by Theorem 4.8 with the additional assumption that $\mathcal{L}_P = 1$, which would be the case if we had the perfect preconditioner. This theory-inspired heuristic is used in all experiments in Section 5, and leads to excellent performance. As an alternative strategy, we present a stochastic linesearch heuristic in Section SM2, which also works very well in practice.

4. Theory. In this section, we provide a convergence analysis for `SAPPHIRE`. Our analysis shows `SAPPHIRE` converges to the global optimum linearly when $L(w)$ is smooth and $\mathcal{R}(w)$ is strongly convex, and sublinearly when $L(w)$ is smooth and $\mathcal{R}(w)$ is convex. We then provide

concrete examples that illustrate when preconditioning improves convergence. In particular, when $L(w)$ is smooth and $\mathcal{R}(w)$ is strongly convex, we establish that SAPPHIRE enjoys local condition-number free convergence.

4.1. Quadratic Regularity.

We begin by defining an important regularity condition [18].

Definition 4.1 (Quadratic Regularity). *Let $f : \mathcal{C} \mapsto \mathbb{R}$ be a smooth convex function, where $\mathcal{C}$ is a closed convex subset of $\mathbb{R}^p$. The function f is* quadratically regular *if there exist constants $0 < \gamma_\ell \leq \gamma_u < \infty$ such that for all $w_0, w_1, w_2 \in \mathbb{R}^p$,*

$$\frac{\gamma_l(\mathcal{C})}{2}\|w_2 - w_1\|^2_{\nabla^2 f(w_0)} \leq f(w_2) - f(w_1) - \langle \nabla f(w_1), w_2 - w_1 \rangle \leq \frac{\gamma_u(\mathcal{C})}{2}\|w_2 - w_1\|^2_{\nabla^2 f(w_0)}. \tag{4.1}$$

Here, $\gamma_u(\mathcal{C})$ and $\gamma_l(\mathcal{C})$ are called the upper and lower quadratic regularity constants, *respectively. Moreover, if $f(w) = \frac{1}{n}\sum_{i=1}^n f_i(w)$ and each f_i are $(\gamma_{u_i}, \gamma_{l_i})$-quadratically regular, we define*

$$\gamma_{u_{\max}}(\mathcal{C}) = \max_{i\in[n]} \gamma_{u_i}(\mathcal{C}), \quad \gamma_{l_{\min}}(\mathcal{C}) = \min_{i\in[n]} \gamma_{l_i}(\mathcal{C}).$$

We also define the quadratic regularity ratio *and the* maximal quadratic regularity ratio *as*

$$\mathfrak{q}(\mathcal{C}) := \frac{\gamma_u(\mathcal{C})}{\gamma_l(\mathcal{C})}, \quad \mathfrak{q}_{\max} := \frac{\gamma_{u_{\max}}(\mathcal{C})}{\gamma_l(\mathcal{C})}.$$

Remark 4.2. If $\mathcal{C} = \mathbb{R}^p$, we will omit explicitly writing $\mathcal{C}$ when presenting the quadratic regularity constants/ratios.

Quadratic regularity generalizes the traditional assumptions of smoothness and strong convexity to the Hessian norm. This assumption is critical to show convergence under infrequent preconditioner updates, as it allows f to be upper and lower bounded in terms of the Hessian evaluated at where the preconditioner was constructed. Most importantly, quadratic regularity holds whenever the function in question is smooth and strongly convex.

Lemma 4.3 (Smoothness and strong-convexity imply quadratic regularity). *Let $f : \mathcal{C} \mapsto \mathbb{R}$ be a β-smooth μ-strongly convex function, where $\mathcal{C}$ is a closed convex subset of $\mathbb{R}^p$. Then f is quadratically regular.*

Unfortunately, when f is only smooth and convex, quadratic regularity fails: the Hessian is only guaranteed to be psd, and where it has a nullspace, it cannot define a norm. Instead, in this case, our convergence analysis rests on the weaker notion of *ρ-weak quadratic regularity*.

Definition 4.4 (ρ-weak quadratic regularity). *Let $f : \mathcal{C} \mapsto \mathbb{R}$ be a smooth convex function, where $\mathcal{C}$ is a closed convex subset of $\mathbb{R}^p$. Then f is ρ-weakly quadratically regular if the regularized function*

$$f_\rho(w) = f(w) + \frac{\rho}{2}\|w\|^2 \text{ is quadratically regular.}$$

We denote the corresponding quadratic regularity constants by: γ_u^ρ, γ_l^ρ, $\gamma_{u_{\max}}^\rho$, and $\gamma_{l_{\min}}^\rho$.

We immediately conclude the following result from this definition and Lemma 4.3.

Lemma 4.5 (Smoothness and convexity imply ρ-weak quadratic regularity). *If f is β-smooth and convex, then it is ρ-weakly quadratically regular for any $\rho > 0$.*

Three different scenarios.. When analyzing (rERM) under the hypothesis of convexity, the standard regularity assumptions are: 1. The $\ell_i(w)$ are smooth and strongly convex for all $i \in [n]$, 2. The ℓ_i are smooth for all $i \in [n]$ and $L(w)$ is strongly convex, and 3. The $\ell_i(w)$ are smooth for all $i \in [n]$. Lemma 4.3 and Lemma 4.5 show these assumptions can be expressed in the language of quadratic regularity:

1) $\ell_i(w)$ is β_i-smooth and strongly convex for all $i \in [n] \implies \ell_i(w)$ is quadratically regular for all $\imath \in [n]$ and $L(w)$ is quadratically regular.
2) $\ell_i(w)$ is β_i-smooth and convex for all $i \in [n]$ and $L(w)$ is strongly convex $\implies \ell_i(w)$ is ρ-weakly quadratically regular for all $\imath \in [n]$ and $L(w)$ is quadratically regular.
3) $\ell_i(w)$ is β_i-smooth and convex for all $i \in [n] \implies \ell_i(w)$ is ρ-weakly quadratically regular for all $\imath \in [n]$ and $L(w)$ is ρ-weakly quadratically regular.

Our analysis focuses on settings 1) and 3), as setting 2) is identical to setting 1) except for a change in one constant. We will elaborate on this point more below.

4.1.1. When quadratic regularity improves over the condition number. In this subsection, we provide intuition for the quadratic regularity ratio through examples that contrast it with the condition number, the quantity that typically appears in the analysis of optimization algorithms. This discussion expands on that of [18]. As our analysis depends on the quadratic regularity ratio and not the condition number, our upper bounds are correspondingly tighter when the quadratic regularity ratio is smaller than the condition number.

Least-squares loss. Let $L(w) = \frac{1}{2n}\|Xw - y\|^2 + \frac{\nu\|w\|_2^2}{2}$, where $X \in \mathbb{R}^{n\times p}$ and $\nu \geq 0$. Since L is a sum of quadratic functions, it has a constant Hessian and equals its own Taylor expansion. It immediately follows that $\gamma_{l_i} = \gamma_{u_i} = 1$. Hence, $\mathfrak{q} = \mathfrak{q}_{\max} = 1$. This ratio is much smaller than the condition number $\frac{\sigma_{\max}(X)^2+n\nu}{\sigma_{\min}(X)^2+n\nu}$ when the data matrix A is ill-conditioned.

GLM on a bounded domain. A function f is *M-quasi-self concordant* (M-qsc) over $\mathcal{C}$ if

$$D^3 f(x)[u,u,v] \leq M\|u\|^2_{\nabla^2 f(x)}\|v\| \quad \forall x \in \mathcal{C} \text{ and } \forall u, v \in \mathbb{R}^p,$$

where $D^3 f(x)$ is the trilinear form representing the third derivative of f [38]. Let $R > 0$ and suppose that $D = \operatorname{diam}(\mathcal{C}) \leq \log(R)/M$. Then the arguments of [18] show that

$$\mathfrak{q}(\mathcal{C}) \leq R^2, \quad \mathfrak{q}_{\max}(\mathcal{C}) \leq R^2.$$

Any GLM (which includes non-quadratic problems like logistic and Poisson regression) with a data matrix X whose rows satisfy $\|x_i\| \leq 1$[2] for all $i \in [n]$ is 1-quasi-self-concordant [27, 15]. Thus, for $R = \mathrm{e}$, we have $\mathfrak{q}(\mathcal{C}) \leq 8$. In contrast, the condition number of L over $\mathcal{C}$ behaves like: $\kappa_L(\mathcal{C}) = \Theta\left(\frac{\sigma^2_{\max}(X)+n\nu}{\sigma^2_{\min}(X)+n\nu}\right)$, which is large when the data matrix A is ill-conditioned. This analysis shows that for objectives of interest, the quadratic regularity ratio may be a constant independent of the condition number even when the function is not well approximated by a quadratic.

[2]This is a standard normalization step employed in packages like `scikit-learn` for stochastic optimizers like SAGA.

4.2. Assumptions. This subsection introduces assumptions needed for our analysis.

Assumption 1 (Convexity and smoothness). *The non-smooth function $r(w)$ is lower semi-continuous and convex, and its effective domain $dom(r) = \{w \in \mathbb{R}^d \mid r(w) < +\infty\}$ is closed.*

Assumption 1 is standard and holds for all practical convex regularizers of interest.

Assumption 2 (ζ-spectral approximation). *There exists $\zeta \in (0, 1)$ such that for each $j \in \mathcal{U}$, the preconditioner P_j constructed at w_j satisfies*

$$\begin{cases} (1-\zeta)P_j \preceq \nabla^2 L(w_j) \preceq (1+\zeta)P_j, & \text{if } L(w) \text{ is quadratically regular,} \\ \nabla^2 L(w_j) \leq (1+\zeta)P_j & \text{if } L(w) \text{ is } \rho\text{-weakly quadratically regular.} \end{cases}$$

Lemma 3.4 and Lemma 3.5 show that the SSN and NySSN preconditioners, when constructed properly, satisfy the conditions of Assumption 2 with high probability. Thus, Assumption 2 can be viewed as conditioning on the good event that the appropriate approximation bound holds. A similar assumption was made in [18]. All our theorems can be shown to hold so long as Assumption 2 holds with high probability: when the failure probability is sufficiently small, we can apply the law of total expectation to obtain the same rate with a slightly worse constant factor. We rely instead on Assumption 2 as it leads to simpler proofs and allows us to establish the convergence of `SAPPHIRE` with any preconditioner that satisfies Assumption 2, rather than only for the SSN and NySSN preconditioners.

4.3. Convergence of `SAPPHIRE`. To establish convergence of `SAPPHIRE`, we must control the smoothness parameter of the stochastic gradient in the preconditioned norm in expectation. A constant $\mathcal{L}_P$ that provides an upper bound on this parameter is known as the *preconditioned expected smoothness constant* [18, 19]. The preconditioned expected smoothness generalizes the Euclidean norm-based expected smoothness constant from [23] to preconditioned space. In the case when $r(w) = 0$ in (rERM), [18, 19] have established bounds on the preconditioned expected smoothness constant. The following lemma provides an explicit expression for $\mathcal{L}_P$ in the general composite case.

Lemma 4.6 (Preconditioned Expected Smoothness). *Instate Assumption 1 and let each $\ell_i(w)$ in (rERM) be convex and twice-continuously differentiable. Let $\rho > 0$ and P be a preconditioner constructed at $w_P \in \mathbb{R}^p$ satisfying*

$$\nabla^2 L(w_P) \preceq (1+\zeta)P.$$

Then for any $w \in \mathbb{R}^p$, if each $\ell_i(w)$ in (rERM) is quadratically regular, then

$$\mathbb{E}\|\widehat{\nabla} L(w) - \widehat{\nabla} L(w_\star)\|^2_{P^{-1}} \leq 2\mathcal{L}_{\mathcal{P}}[\mathcal{R}(w) - \mathcal{R}(w_\star)],$$

where

$$\mathcal{L}_P = \left(\frac{n(b_g - 1)}{b_g(n-1)} \gamma_u + \tau_\star^\rho \frac{n - b_g}{b_g(n-1)} \gamma_{u_{\max}} \right) (1+\zeta).$$

The proof is provided in section SM3.

Lemma 4.6 extends the classical smoothness condition in deterministic optimization to the stochastic and preconditioned setting and establishes a direct relationship between the

preconditioned gradient norm variance and the suboptimality of $\mathcal{R}(w)-\mathcal{R}(w^\star)$. It generalizes the results of [18, 19] to the convex composite setting. If the individual ℓ_i's are ρ-weakly quadratically regular, then $\mathcal{L}_\mathcal{P}$ in Lemma 4.6 will be constructed by $\gamma_u^\rho, \tau_\star^\rho$, and $\gamma_{u_{\max}}^\rho$.

Lemma 4.7 (Preconditioned Stochastic Variance). *Instate Assumption 1 and Assumption 2, and define the variance-reduced stochastic gradient at inner iteration k in outer iteration s, $v_k^{(s)} = \widehat{\nabla} L(w_k^{(s)}) - \widehat{\nabla} L(\hat{w}^{(s)}) + \nabla L(\hat{w}^{(s)})$. The variance of this stochastic gradient is bounded in the preconditioned norm as*

$$\mathbb{E}\|v_k^{(s)} - \nabla L(w_k^{(s)})\|^2_{(P_k^{(s)})^{-1}} \leq 4\mathcal{L}_\mathcal{P}[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w_\star) + \mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w_\star)].$$

The proof is provided in section SM4.

Lemma 4.7 shows that by employing the variance-reduced stochastic gradient $v_k^{(s)}$, we are guaranteed that the variance of the stochastic gradient goes to zero as we approach the optimum. This property is essential to establishing convergence. If the gradient variance does not go to zero as we approach the optimum, we can only reach a neighborhood of the optimum with a fixed stepsize.

4.3.1. Convergence for quadratically regular L. Here, we establish global convergence of `SAPPHIRE` under quadratic regularity of L. For brevity, we only consider the case when each $\ell_i(w)$ is quadratically regular. The argument and resulting statements for the case when the $\ell_i(w)$ are only ρ-weakly quadratically regular are identical, except that we replace $\mathcal{L}_P$ by $\mathcal{L}_{P_\rho}$.

Theorem 4.8 (Global Linear Convergence). *Instate Assumption 1 and Assumption 2. Suppose each $\ell_i(w)$ is quadratically regular. Run Algorithm 3.1 with learning rate $0 < \eta < \frac{1}{4\mathcal{L}_P}$. Then the output of Algorithm 3.1 satisfies*

$$\mathbb{E}[\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w_\star)] \leq \left(\frac{1}{(1-\zeta)\gamma_\ell \eta(1-4\eta\mathcal{L}_P)m} + \frac{4\eta\mathcal{L}_P(m+1)}{(1-4\eta\mathcal{L}_P)m}\right)^s (\mathcal{R}(w_0) - \mathcal{R}(w_\star)).$$

Thus, setting $\eta = \mathcal{O}(1/\mathcal{L}_P)$ and $m = \mathcal{O}(\frac{\mathcal{L}_P}{(1-\zeta)\gamma_\ell})$, we have

$$\mathbb{E}[\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w_\star)] \leq \left(\frac{2}{3}\right)^s (\mathcal{R}(w_0) - \mathcal{R}(w_\star)).$$

Hence, the error falls below $\epsilon > 0$ after $s \geq 3\log\left(\frac{\mathcal{R}(\hat{w}^{(0)})-\mathcal{R}(w_\star)}{\epsilon}\right)$ outer iterations and the total number of stochastic gradient queries needed to reach an ϵ-suboptimal point is bounded by

$$\mathcal{O}\left(\left(n + \frac{n}{1-\zeta}\left(\frac{b_g-1}{n-1}\mathfrak{q} + \frac{\tau_\rho^\star}{n}\frac{n-b_g}{n-1}\mathfrak{q}_{\max}\right)\right)\log\left(\frac{1}{\epsilon}\right)\right). \tag{4.2}$$

The proof of Theorem 4.8 is provided in Appendix A.1.

Theorem 4.8 establishes global linear convergence of `SAPPHIRE` when L is quadratically regular and each ℓ_i is quadratically regular. It substantially generalizes Theorem 17 in [18], which only establishes convergence in the special case $r(w) = \nu/2\|w\|_2^2$. In the preconditioned

setting, the role of the condition numbers κ and $\kappa_{\max}$ are played by the quadratic regularity ratios $\mathfrak{q}$ and $\mathfrak{q}_{\max}$. The convergence rate is controlled by a convex combination of $\mathfrak{q}$ and $\mathfrak{q}_{\max}$, which captures the benefits of minibatching. As b_g increases from 1 to n, the weight on the smaller ratio $\mathfrak{q}$ approaches unity, while the weight on $\mathfrak{q}_{\max}$ approaches 0. When $\mathfrak{q}, \mathfrak{q}_{\max} = \mathcal{O}(1)$, which corresponds to the setting when preconditioning helps globally, the total number of gradient queries scales as

$$\mathcal{O}\left(\left(n + \frac{n}{1-\zeta}\right)\log\left(\frac{1}{\epsilon}\right)\right).$$

Thus, `SAPPHIRE's` convergence rate is completely determined by the quality of the preconditioner, whose impact on the convergence rate comes through the $(1-\zeta)^{-1}$ factor. In the case when $1-\zeta = \Omega(1)$, `SAPPHIRE` exhibits the optimal number of queries $\mathcal{O}(n\log(1/\epsilon))$.

Remark 4.9. If the regularizer corresponds to a projection onto a closed convex set $\mathcal{C}$, then $\mathfrak{q}$ and $\mathfrak{q}_{\max}$ in Theorem 4.8 should be replaced by $\mathfrak{q}(\mathcal{C})$ and $\mathfrak{q}_{\max}(\mathcal{C})$.

Theorem 4.8 along with our discussion in Subsection 4.1.1 immediately yields the following corollary, which provides two concrete settings where `SAPPHIRE` exhibits an optimal convergence rate.

Corollary 4.10. *Under the hypotheses of Theorem 4.8 with the additional assumption that $1-\zeta = \Omega(1)$, the following statements hold:*

1. *Suppose $L(w) = \frac{1}{2n}\|Xw-b\|^2 + \frac{\nu\|w\|^2}{2}$ and $r(w) = \mu\|w\|_1$. Run Algorithm 3.1 with $\mathcal{U} = \{0\}$, $\eta = \mathcal{O}(1)$, $m = \mathcal{O}(1)$ inner iterations, and $s = \mathcal{O}\left(\log\left(\frac{1}{\epsilon}\right)\right)$ outer iterations. Then Algorithm 3.1 converges to expected loss ϵ with the total number of full gradient queries bounded as $\mathcal{O}(n\log(1/\epsilon))$.*
2. *Suppose $L(w) = \frac{1}{n}\sum_{i=1}^n \ell(x_i^T w) + \frac{\nu\|w\|}{2}^2$, with $\|x_i\| \le 1$ for all $i \in [n]$ and $r(w) = 1_{\mathcal{C}}$, where $\mathcal{C}$ is a closed convex set with $\operatorname{diam}(\mathcal{C}) \le 2$. Run Algorithm 3.1 with $\mathcal{U} = \{0\}$, $\eta = \mathcal{O}(1)$, $m = \mathcal{O}(1)$ inner iterations, and $s = \mathcal{O}\left(\log\left(\frac{1}{\epsilon}\right)\right)$ outer iterations. Then converges to expected loss ϵ with the total number of full gradient queries bounded as $\mathcal{O}(n\log(1/\epsilon))$.*

4.3.2. Convergence for convex ρ-weak quadratically regular L. When $L(w)$ is only convex and smooth, a common setting in large-scale machine learning problems, i.e., Lasso, `SAPPHIRE` admits the following ergodic convergence guarantee.

Theorem 4.11 (`SAPPHIRE`: Convex ρ-Weak Quadratically Regular Convergence). *Instate Assumption 1 and Assumption 2. Fix $m > 0$. Suppose each $\ell_i(w)$ is convex and ρ-weakly quadratically regular. Run Algorithm 3.1 with Option 2 and learning rate $\eta = \min\{\frac{1}{4\mathcal{L}_P(m+2)}, \frac{1}{8(m+2)}\}$. Define the sample average as $\bar{w} = \frac{1}{Sm}\sum_{s=0}^{S-1}\sum_{k=1}^{m}\hat{w}_k^{(s)}$, then after S outer iterations,*

$$\mathbb{E}\left[\mathcal{R}(\bar{w}) - \mathcal{R}(w_\star)\right] \le \frac{48(\mathcal{L}_P^2+4)(m+2)}{S}\|w_0 - w_\star\|_{P_0^{(0)}}^2 + \frac{12(\mathcal{L}_P+2)}{S}\left(\mathcal{R}(w_0) - \mathcal{R}(w_\star)\right).$$

Thus, after $S = \mathcal{O}\left(\frac{m\mathcal{L}_P^2}{\epsilon}\right)$ outer iterations,

$$\mathbb{E}\left[\mathcal{R}(\bar{w}) - \mathcal{R}(w_\star)\right] \le \epsilon\left[\|w_0 - w_\star\|_{P_0^{(0)}}^2 + \mathcal{R}(w_0) - \mathcal{R}(w_\star)\right].$$

The proof of Theorem 4.11 is provided in Section SM7.

Theorem 4.11 establishes that SAPPHIRE converges ergodically at an $\mathcal{O}(1/\epsilon)$ rate, matching the rate of gradient descent in the smooth convex case and ProxSVRG without preconditioning [42]. Unfortunately, the dependence of S on m in the theorem implies the total gradient queries scale as $\mathcal{O}(\frac{n+m^2\mathcal{L}_P^2}{\epsilon})$, rather than the expected $\mathcal{O}(n+\mathcal{L}_P/\epsilon)$. This coupling also appears in analysis without preconditioning [42], with a rate of $\mathcal{O}(\frac{n+m^2\mathcal{L}}{\epsilon})$, so this issue does not stem from SAPPHIRE employing preconditioning. The issue could be avoided by combining SAPPHIRE with a black-box reduction such as AdaptReg [3], which is based upon approximately minimizing a sequence of strongly convex surrogates. However, we have not found this to be necessary in practice. The suboptimal dependence on m arises because Theorem 4.11 assumes the very conservative hyperparameter setting: $\eta = \mathcal{O}(1/(\mathcal{L}_P m))$. In practice, we run SAPPHIRE with $\eta = \mathcal{O}(1/\mathcal{L}_P)$, which corresponds to the setting in Theorem 4.8 when $L(w)$ is quadratically regular. While this more aggressive hyperparameter setting is not supported by Theorem 4.11, it yields excellent empirical performance in practice (section 5). The theory-practice gap in the setting of η shows Theorem 4.11 is overly conservative in the requirements it stipulates for SAPPHIRE to converge.

When global convergence rates are pessimistic. Theorem 4.11 can overestimate the time needed to solve (rERM) when the regularizer is structured. Consider the Lasso problem where $L(w) = \frac{1}{2n}\|Xw - y\|^2$, $X \in \mathbb{R}^{n\times p}$ with $p > n$, and $r(w) = \lambda\|w\|_1$. When $p > n$, the covariance matrix $\frac{1}{n}X^TX$ is degenerate, so $L(w)$ is convex but not strongly convex. However, the defining property of the Lasso model is that the solution vector $w_\star$ is sparse. When restricted to the support set of the solution $w_\star$, the covariance matrix is often no longer degenerate, so strong convexity holds as long as the iterates stay on the support set, which implies a linear convergence rate. Optimization algorithms that identify the low-dimensional manifold on which the solution lives within a finite number of iterations and remain there are said to possess the *manifold identification property* [30, 31, 47]. Variance-reduced stochastic gradient methods like ProxSVRG, SAGA, and SAPPHIRE possess this property [42]. Hence, for problems like the Lasso, SAPPHIRE will exhibit an initial sublinear convergence phase, followed by a linearly convergent phase once it has identified the manifold on which the solution lives. For some problem instances, this identification occurs rapidly so that the linearly convergent phase dominates—in which case the rate predicted by Theorem 4.11 is highly pessimistic. The manifold identification property can still be beneficial even when the objective is globally strongly convex, as with the elastic net. On the low-dimensional manifold, $L(w)$ can be better conditioned than it is globally, so the preconditioner does not have to be as good to ensure the preconditioned condition number is close to unity.

4.4. Local convergence of SAPPHIRE.

In this subsection, we establish the local condition number free convergence of SAPPHIRE. We focus on the case that each $\ell_i(w)$ is ν-strongly convex and has an M-Lipschitz Hessian. Local convergence is established within the following neighborhood of the optimum $w_\star$:

$$\mathcal{N}_{\varepsilon_0}(w_\star) := \left\{ \|w - w_\star\|^2_{\nabla^2 L(w_\star)} \le \frac{\nu^{3/2}}{2M} \right\}.$$

The key to achieving fast local convergence is that within $\mathcal{N}_{\varepsilon_0}(w_\star)$, the quadratic regularity constants are guaranteed to be very close to unity, enabling us to establish the following result.

Theorem 4.12. *Let $\varepsilon_0 \in (0, 1/6]$. Suppose that each ℓ_i is ν-strongly convex, and has an M-Lipschitz Hessian, and that $w_0 \in \mathcal{N}_{\varepsilon_0}(w_\star)$. Instate Assumption 1 and Assumption 2 with $\zeta = \varepsilon_0$. Run Algorithm 3.1 using Option 2 with $\mathcal{U} = \{0\}$, $m = 10$ inner iterations, $s = 2\log\left(\frac{1}{\epsilon}\right)$ outer iterations, $\eta = 1$, and $b_g = \tilde{\mathcal{O}}\left(\tau^\rho(\mathcal{N}_{\varepsilon_0}(w_\star))\log(\frac{1}{\delta})\right)$. Then, with probability at least $1-\delta$,*

$$\|\hat{w}^{(s)} - w_\star\|_{\nabla^2 L(w_\star)} \leq \epsilon.$$

Hence, the total number of stochastic gradient queries within ϵ distance of the optimum is bounded by

$$\tilde{\mathcal{O}}\left(n\log\left(\frac{1}{\epsilon}\right)\right).$$

The proof of Theorem 4.12 is provided in Section SM8.

Theorem 4.12 shows that within in $\mathcal{N}_{\varepsilon_0}(w_\star)$, SAPPHIRE enjoys linear convergence independent of the condition number. It provides a generalization of Theorem 19 in [18] to the strongly convex composite setting. As in [18], the required gradient batchsize only scales as $\tilde{\mathcal{O}}\left(\tau^\nu(\mathcal{N}_{\varepsilon_0}(w_\star))\right)$, which is never larger than the condition number κ or n and is often significantly smaller, as we shall see shortly below when we specialize to GLMs. Having a gradient batchsize requirement independent of κ is crucial in the ill-conditioned setting common in large-scale machine learning, where we can easily have $\kappa > n$.

To make Theorem 4.12 more concrete, we present the following corollary, which specializes to the case when $L(w)$ corresponds to a GLM.

Corollary 4.13. *Let $X \in \mathbb{R}^{n\times p}$, and let $X_i \in \mathbb{R}^p$ denote the ith row of X. Under the hypotheses of Theorem 4.12, suppose that $\ell_i(w) = \ell(x_i^\top w) + \frac{\nu\|w\|^2}{2}$, $\frac{1}{n}\lambda_j(X^\top X) \leq Cj^{-2\beta}$ for $\beta > 1$, and $\nabla^2 L(w_\star)$ is ridge-leverage incoherent. Then if $b_g = \mathcal{O}\left(\sqrt{n}\log\left(\frac{1}{\delta}\right)\right)$, it holds with probability at least $1-\delta$ that only*

$$\tilde{\mathcal{O}}\left(n\log\left(\frac{1}{\epsilon}\right)\right)$$

stochastic gradient evaluations are required to ensure the output of Algorithm 3.1 satisfies

$$\|\hat{w}^{(s)} - w_\star\|_{\nabla^2 L(w_\star)} \leq \epsilon.$$

The proof is provided in Section SM9.

Corollary 4.13 shows that under a spectral decay condition on X that commonly arises in machine learning problems, SAPPHIRE only needs to use a batchsize of $\tilde{\mathcal{O}}\left(\sqrt{n}\right)$ to ensure a condition number-free convergence with high probability. Thus, we can set b_g to be far smaller than n, while ensuring a fast convergence rate. This concrete example shows that the dependence upon $\tau^\rho_\star(\mathcal{N}_{\varepsilon_0}(w_\star))$ yields real improvements over results where the batch size depends upon κ.

5. Experiments.

In this section, we verify the effectiveness of SAPPHIRE (Algorithm 3.1) with experiments on real-world data on a variety of machine learning tasks from LIBSVM [11], OpenML [50], and torchvision [34]. Our experiments utilize a diverse collection of datasets,

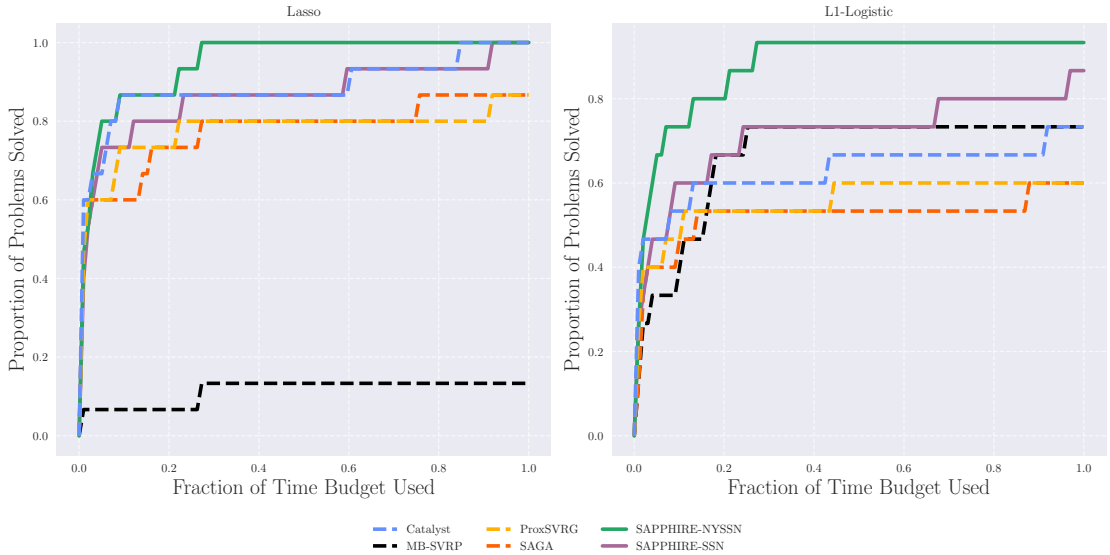

Figure 2. *Performance Plot with Small Regularization*

which capture a variety of settings: (big-data) $n \gg p$, wide-data ($p \gg n$), and big and high-dimensional ($n \sim p$). Moreover, we consider datasets of varying degrees of sparsity, ranging from extremely sparse to completely dense. Please see Table SM1 for details.

We organize the experiments as follows:

- Performance comparisons (Subsection 5.1): We show the effectiveness of `SAPPHIRE` for solving (rERM). We compare it with existing stochastic first-order optimizers `Catalyst` [32], `ProxSVRG` [52], and `SAGA` [13], and a stochastic second-order method `MB-SVRP` [51].
- Showcase on large-scale applications (Subsection 5.2): We demonstrate `SAPPHIRE` exhibits superior performance on real world large-scale learning tasks: click prediction, malicious link detection, and phenotype prediction from genetic data.
- Verification of `SAPPHIRE` convergence (Subsection 5.3): We provide experiments verifying that `SAPPHIRE` satisfies the convergence guarantees presented in Section 4.

`SAPPHIRE` is ran using the hyperparameter settings presented in Section 3, and competing algorithms are run according to standard recommendations in the literature. See Section SM10 for a detailed overview. Code to reproduce the experiments may be found at the GitHub Repository https://github.com/udellgroup/sapphire.

5.1. Performance experiments. For the performance experiments, we consider 14 regression and classification tasks. We train a lasso model for regression tasks and l_1-logistic regression for classification tasks. The regularization parameter is fixed at $10^{-2}\|X^T y\|_\infty/n$, corresponding to a small value of regularization that leads to a harder optimization problem. As an ablation, we also consider larger values of regularization; see Section SM11 for these results. For each task, the optimizer is given 120 seconds to solve the problem. We terminate an optimizer early if the norm of the gradient mapping falls below 10^{-4}.

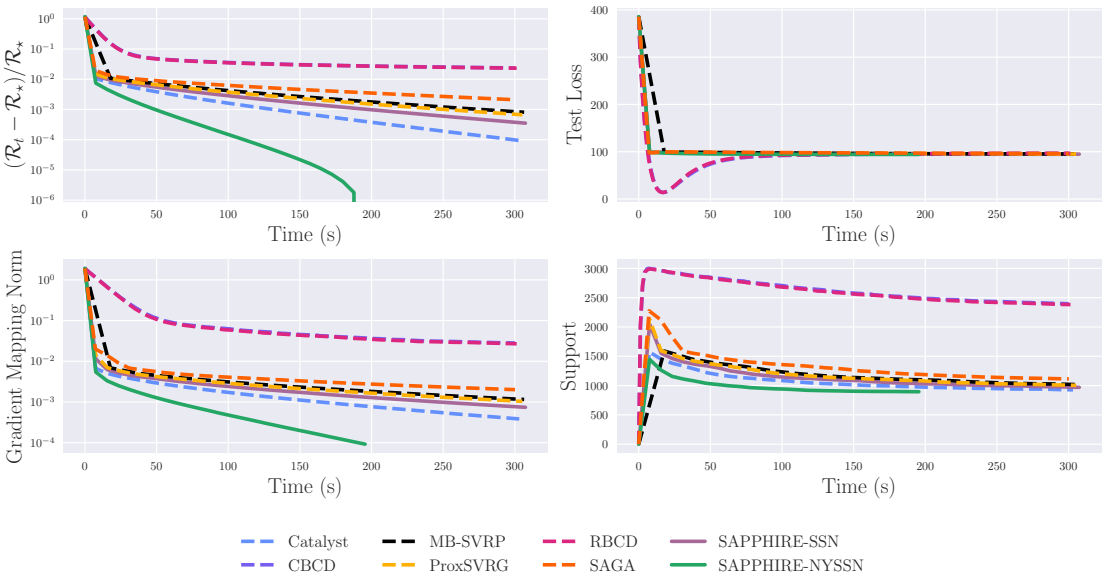

Figure 3. *Showcase Experiment on Gene Selection*

Figure 2 shows that both SAPPHIRE variants outperform other methods on these tasks. Notably, SAPPHIRE with NySSN preconditioner finishes all tasks in only 25% of the time budget. In contrast, Catalyst requires 80% of the time budget on regression tasks, and no other baseline method is able to complete all classification tasks within the time budget.

5.2. Showcase experiments. First, we evaluate SAPPHIRE on a click-through rate prediction task using 2014 Avazu-Kaggle competition data. This dataset is large-scale with $10^7 \times 10^6$ size and highly sparse with only 0.0001% non-zero entries. We train it using logistic regressions with elastic-net regularization. As shown in Figure 1, SAPPHIRE achieves fast convergence in less than 60 seconds and yields more compact feature selections compared to baselines.

Second, we evaluate SAPPHIRE selecting genes to predict phenotypes using UK Biobank data. This dataset is large-scale, with size $2.63 \cdot 10^5 \times 10^3$, and dense, with 99.6% non-zero entries. We train it using least-squares regression with elastic-net regularization. Figure 3 shows SAPPHIRE yields the most compact gene selections in 50 seconds and converges fastest.

5.3. Convergence experiments. In this subsection, we empirically verify the convergence theory developed in Section 4. We consider four datasets: covtype, ova_lung, rcv1, and yearmsd. These four datasets cover the data regimes: $n \gg p, p \gg n$, and $n \sim p$. For simplicity, we only consider SAPPHIRE with the NySSN preconditioner. For covtype and rcv1, we train an l_1-logistic regression model with penalty strength $\mu = 10^{-1}\|X^T y\|_\infty / n$. For yearmsd, we train a lasso model with the same regularization strength, while for ova_lung, we train an elastic-net regression model with $\mu = 10^{-1}\|X^T y\|_\infty / n$, $\nu = 10^{-1}/n$. For each problem, the reference point used for the optimum $\mathcal{R}_\star$ was found by running SAPPHIRE until the norm of the gradient mapping fell below 10^{-12}.

Figure 4 presents the results. SAPPHIRE exhibits linear convergence on each of the three

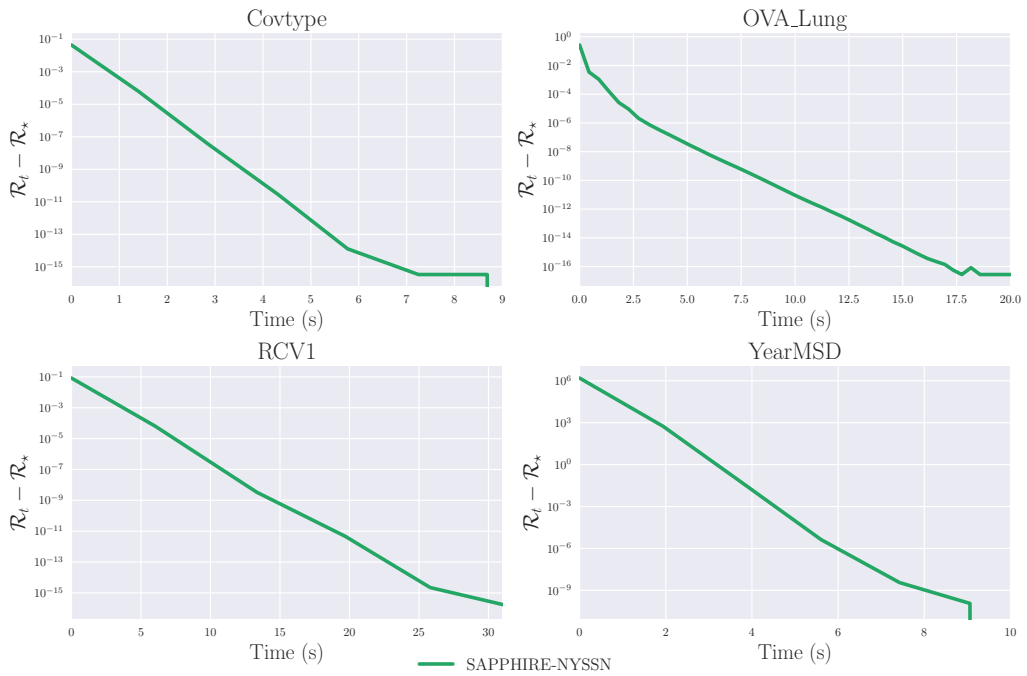

Figure 4. *Verification of convergence of* `SAPPHIRE`*.* `SAPPHIRE` *exhibits linear convergence on all four datasets, consistent with the theory and discussion presented in Section 4.*

problems, agreeing with the theory developed in Section 4. In the case of covtype, the data matrix A is numerically rank deficient, but `SAPPHIRE` still exhibits linear convergence. The rapid convergence despite the lack of strong convexity in the problem is consistent with the discussion in Subsection 4.3, where the manifold identification property leads to a much faster rate of convergence than the worst-case rate predicted by Theorem 4.11.

6. Conclusion.

We propose `SAPPHIRE`, an optimization algorithm to accelerate large-scale statistical learning for ill-conditioned and non-smooth regularized empirical risk minimization problems.

We provide a rigorous theoretical analysis for the convergence of the `SAPPHIRE` algorithm, demonstrating global and local linear convergence under quadratic regularity and sublinear convergence under general convex and weak quadratic regular conditions. Empirical results across diverse datasets validate the superior performance of our algorithm in both convergence speed and computational efficiency compared to baseline methods like `Prox-SVRG` and `SAGA`.

Therefore, we introduce a robust and efficient framework to address the challenges of ill-conditioned, composite, large-scale optimization problems arising in machine learning. By integrating variance reduction techniques with preconditioned proximal mappings, the `SAPPHIRE` algorithm not only improves optimization performance but also offers a scalable and versatile solution for modern data-driven applications.

Appendix A. Proofs for global convergence of `SAPPHIRE`.

In this section, we provide proofs for all results related to the global convergence of `SAPPHIRE`.

A.1. SAPPHIRE: Global Linear Convergence. The proof is based on a sequence of lemmata. We begin with the following result, which provides a bound for SAPPHIRE after one inner iteration.

Lemma A.1 (Bound for One Inner Iteration). *Suppose we are in outer iteration s at inner iteration k. Then the following inequality holds*

$$\mathbb{E}\left[\|w_{k+1}^{(s)} - w^\star\|^2_{P_k^{(s)}}\right] + 2\eta\mathbb{E}\left[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w^\star)\right]$$
$$\leq \|w_k^{(s)} - w^\star\|^2_{P_k^{(s)}} + 8\eta^2\mathcal{L}_P[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w^\star) + \mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)].$$

The proof is given in Section SM6.

Lemma A.1 establishes a bound for one inner iteration, which we use to establish the following contraction relation for one outer iteration.

Lemma A.2 (Bound for One Outer Iteration). *Suppose we are in outer iteration s. Then the output of this outer iteration $\hat{w}^{(s+1)}$ satisfies*

$$\mathbb{E}[\mathcal{R}(\hat{w}^{(s+1)})] - \mathcal{R}(w^\star) \leq \left(\frac{1}{(1-\zeta)\gamma_\ell\eta(1-4\mathcal{L}_P\eta)m} + \frac{4\mathcal{L}_P\eta(m+1)}{(1-4\mathcal{L}_P\eta)m}\right)[\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)].$$

Proof. Applying Lemma A.1 for $k = 0, ..., m-1$, and summing yields

$$\sum_{k=0}^{m-1}\mathbb{E}[\|w_{k+1}^{(s)} - w^\star\|_{P_k^{(s)}}] + 2\eta\sum_{k=0}^{m-1}\mathbb{E}\left[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w^\star)\right]$$
$$\leq \sum_{k=0}^{m-1}\|w_k^{(s)} - w^\star\|^2_{P_k^{(s)}} + 4\eta\mathcal{L}_P\sum_{k=0}^{m-1}[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w^\star) + \mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)]$$

Taking the total expectation over the inner iterations and rearranging yields

$$\mathbb{E}[\|w_k^{(m)} - w_\star\|^2_{P_k^{(s)}}] + 2\eta\mathbb{E}[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w_\star)] + 2\eta(1-4\eta\mathcal{L}_P)\sum_{k=1}^{m-1}\mathbb{E}[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w_\star)]$$
$$\leq \|\hat{w}^{(s)} - w_\star\|^2_{P_k^{(s)}} + 8(m+1)\eta^2\mathcal{L}_P(\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)).$$

Our choice of η implies $2\eta \geq 2\eta(1-4\eta\mathcal{L}_P)$, yielding

$$\mathbb{E}[\|w_k^{(m)} - w_\star\|^2_{P_k^{(s)}}] + 2\eta(1-4\eta\mathcal{L}_P)\sum_{k=1}^{m}\mathbb{E}[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w_\star)]$$
$$\leq \|\hat{w}^{(s)} - w_\star\|^2_{P_k^{(s)}} + 8(m+1)\eta^2\mathcal{L}_P(\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)).$$

Rearranging, using the definition of $\hat{w}^{(s+1)}$ and convexity of $\mathcal{R}$ yields

$$\mathbb{E}\left[\mathcal{R}(\hat{w}^{(s+1)}) - \mathcal{R}(w^\star)\right] \leq \frac{1}{2\eta m\,(1-4\eta\mathcal{L}_P)}\|\hat{w}^{(s)} - w^\star\|^2_{P_k^{(s)}}$$
$$+ \frac{4\eta\mathcal{L}_P(m+1)}{m(1-4\eta\mathcal{L}_P)}\left(\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)\right).$$

Now, by lower quadratic regularity of L and optimality of $w^\star$, we have

$$\begin{aligned}\|\hat{w}^{(s)} - w^\star\|^2_{P_0^{(s)}} &\leq \frac{2}{(1-\zeta)\gamma_\ell}[L(\hat{w}^{(s)}) - L(w^\star)] \\ &\leq \frac{2}{(1-\zeta)\gamma_\ell}[L(\hat{w}^{(s)}) - L(w^\star) + r(\hat{w}^{(s)}) - r(w^\star)] \\ &= \frac{2}{(1-\zeta)\gamma_\ell}[\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)].\end{aligned}$$

Here, the second inequality follows from the fact that $r(\hat{w}^{(s)}) - r(w^\star) \geq 0$ as $w^\star$ is optimal.

Combining this with our previous bound, we conclude

$$\mathbb{E}[\mathcal{R}(\hat{w}^{(s+1)}) - \mathcal{R}(w^\star)] \leq \left(\frac{1}{(1-\zeta)\gamma_\ell\eta(1-4\eta\mathcal{L}_P)m} + \frac{4\eta\mathcal{L}_P(m+1)}{(1-4\eta\mathcal{L}_P)m}\right)[\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)].$$

The contraction relation in Lemma A.2 gives us everything we need to prove Theorem 4.8.

A.2. Proof for Theorem 4.8.

Proof. Set $\eta = \frac{1}{16\mathcal{L}_P}$ and $m = \frac{100\mathcal{L}_P}{(1-\zeta)\gamma_\ell}$. By Lemma A.2, we perform the recursion and obtain

$$\mathbb{E}\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star) \leq \left(\frac{2}{3}\right)^s (\mathcal{R}(\hat{w}^{(0)}) - \mathcal{R}(w^\star)).$$

Therefore, if the number of stages satisfies

$$s \geq 3\log\left(\frac{\mathcal{R}(\hat{w}^{(0)}) - \mathcal{R}(w^\star)}{\epsilon}\right),$$

then we achieve

$$\mathbb{E}\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star) \leq \epsilon.$$

Observing that each stage requires $n + 2mb_g$ component gradient evaluations, we immediately conclude that the total number stochastic gradient evaluations is given by

$$\mathcal{O}\left(\left[n + \frac{\mathcal{L}_P b_g}{(1-\zeta)\gamma_\ell}\right]\log\left(\frac{1}{\epsilon}\right)\right).$$

The rest of the claim follows by substituting in the expression for $\mathcal{L}_\mathcal{P}$ in Lemma 4.6. ∎

Acknowledgments. We would like to thank Manuel Rivas for helpful discussions and providing access to the UKBiobank data.

REFERENCES

[1] Z. Allen-Zhu, *Katyusha: The first direct acceleration of stochastic gradient methods*, Journal of Machine Learning Research, 18 (2018), pp. 1–51.

[2] Z. Allen-Zhu, *Natasha 2: Faster non-convex optimization than sgd*, Advances in neural information processing systems, 31 (2018).
[3] Z. Allen-Zhu and E. Hazan, *Optimal black-box reductions between optimization objectives*, Advances in Neural Information Processing Systems, 29 (2016).
[4] Z. Allen-Zhu and E. Hazan, *Variance reduction for faster non-convex optimization*, in International conference on machine learning, PMLR, 2016, pp. 699–707.
[5] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 183–202.
[6] A. S. Berahas, J. Nocedal, and M. Takác, *A multi-batch l-bfgs method for machine learning*, Advances in Neural Information Processing Systems, 29 (2016).
[7] J. Blanchet, C. Cartis, M. Menickelly, and K. Scheinberg, *Convergence rate analysis of a stochastic trust-region method via supermartingales*, INFORMS Journal on Optimization, 1 (2019), pp. 92–119.
[8] 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.
[9] 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.
[10] 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.
[11] C.-C. Chang and C.-J. Lin, *Libsvm: a library for support vector machines*, ACM Transactions on Intelligent Systems and Technology, 2 (2011), pp. 1–27.
[12] J. Chen, R. Yuan, G. Garrigos, and R. M. Gower, *SAN: stochastic average Newton algorithm for minimizing finite sums*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2022, pp. 279–318.
[13] A. Defazio, F. Bach, and S. Lacoste-Julien, *SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives*, Advances in Neural Information Processing Systems, 27 (2014).
[14] M. Derezinski, *Stochastic variance-reduced newton: Accelerating finite-sum minimization with large batches*, in OPT 2023: Optimization for Machine Learning.
[15] N. Doikov, *Minimizing quasi-self-concordant functions by gradient regularization of newton method*, arXiv preprint arXiv:2308.14742, (2023).
[16] M. A. Erdogdu and A. Montanari, *Convergence rates of sub-sampled Newton methods*, Advances in Neural Information Processing Systems, 28 (2015).
[17] M. Feng, Z. Frangella, and M. Pilanci, *Cronos: Enhancing deep learning with scalable gpu accelerated convex neural networks*, arXiv preprint arXiv:2411.01088, (2024).
[18] Z. Frangella, P. Rathore, S. Zhao, and M. Udell, *PROMISE: Preconditioned stochastic optimization methods by incorporating scalable curvature estimates*, Journal of Machine Learning Research, 25 (2024), pp. 1–57, http://jmlr.org/papers/v25/23-1187.html.
[19] Z. Frangella, P. Rathore, S. Zhao, and M. Udell, *Sketchysgd: reliable stochastic optimization via randomized curvature estimates*, SIAM Journal on Mathematics of Data Science, 6 (2024), pp. 1173–1204.
[20] Z. Frangella, J. A. Tropp, and M. Udell, *Randomized nyström preconditioning*, SIAM Journal on Matrix Analysis and Applications, 44 (2023), pp. 718–752.
[21] S. Garg, A. S. Berahas, and M. Dereziński, *Second-order information promotes mini-batch robustness in variance-reduced gradients*, arXiv preprint arXiv:2404.14758, (2024).
[22] R. Gower, D. Kovalev, F. Lieder, and P. Richtárik, *RSN: Randomized Subspace Newton*, Advances in Neural Information Processing Systems, 32 (2019).
[23] R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik, *SGD: General analysis and improved rates*, in International Conference on Machine Learning, PMLR, 2019, pp. 5200–5209.
[24] A. Ilyas, S. M. Park, L. Engstrom, G. Leclerc, and A. Madry, *Datamodels: Predicting predictions from training data*, in Proceedings of the 39th International Conference on Machine Learning, 2022.
[25] S. J Reddi, S. Sra, B. Poczos, and A. J. Smola, *Proximal stochastic methods for nonsmooth non-*

convex finite-sum optimization, Advances in Neural Information Processing Systems, 29 (2016).
[26] R. Johnson and T. Zhang, *Accelerating stochastic gradient descent using predictive variance reduction*, Advances in Neural Information Processing Systems, 26 (2013).
[27] S. P. Karimireddy, S. U. Stich, and M. Jaggi, *Global linear convergence of Newton's method without strong-convexity or Lipschitz gradients*, arXiv preprint arXiv:1806.00413, (2018).
[28] J. M. Kohler and A. Lucchi, *Sub-sampled cubic regularization for non-convex optimization*, in International Conference on Machine Learning, PMLR, 2017, pp. 1895–1904.
[29] X. Li, S. Wang, and Z. Zhang, *Do subsampled newton methods work for high-dimensional data?*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 4723–4730.
[30] J. Liang, J. Fadili, and G. Peyré, *Activity identification and local linear convergence of forward–backward-type methods*, SIAM Journal on Optimization, 27 (2017), pp. 408–437.
[31] J. Liang, J. Fadili, and G. Peyré, *Local convergence properties of douglas–rachford and alternating direction method of multipliers*, Journal of Optimization Theory and Applications, 172 (2017), pp. 874–913.
[32] H. Lin, J. Mairal, and Z. Harchaoui, *Catalyst acceleration for first-order convex optimization: from theory to practice*, Journal of Machine Learning Research, 18 (2018), pp. 1–54.
[33] B. Liu, M. Xie, and M. Udell, *Controlburn: Feature selection by sparse forests*, in Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, 2021, pp. 1045–1054.
[34] S. Marcel and Y. Rodriguez, *Torchvision the machine-vision package of torch*, in Proceedings of the 18th ACM international conference on Multimedia, 2010, pp. 1485–1488.
[35] S. Y. Meng, S. Vaswani, I. H. Laradji, M. Schmidt, and S. Lacoste-Julien, *Fast and furious convergence: Stochastic second order methods under interpolation*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 1375–1386.
[36] P. Moritz, R. Nishihara, and M. Jordan, *A linearly-convergent stochastic L-BFGS algorithm*, in Artificial Intelligence and Statistics, PMLR, 2016, pp. 249–258.
[37] Y. Nesterov, *Gradient methods for minimizing composite functions*, Mathematical programming, 140 (2013), pp. 125–161.
[38] Y. Nesterov, *Lectures on Convex Optimization*, Springer, 2018.
[39] C. Paquette, H. Lin, D. Drusvyatskiy, J. Mairal, and Z. Harchaoui, *Catalyst for gradient-based nonconvex optimization*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2018, pp. 613–622.
[40] M. Pilanci and T. Ergen, *Neural networks are convex regularizers: Exact polynomial-time convex optimization formulations for two-layer networks*, in International Conference on Machine Learning, PMLR, 2020, pp. 7695–7705.
[41] 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.
[42] C. Poon, J. Liang, and C. Schoenlieb, *Local convergence properties of saga/prox-svrg and acceleration*, in International Conference on Machine Learning, PMLR, 2018, pp. 4124–4132.
[43] P. Rathore, W. Lei, Z. Frangella, L. Lu, and M. Udell, *Challenges in training pinns: A loss landscape perspective*, arXiv preprint arXiv:2402.01868, (2024).
[44] S. J. Reddi, A. Hefny, S. Sra, B. Poczos, and A. Smola, *Stochastic variance reduction for nonconvex optimization*, in International conference on machine learning, PMLR, 2016, pp. 314–323.
[45] F. Roosta, Y. Liu, P. Xu, and M. W. Mahoney, *Newton-mr: Inexact Newton method with minimum residual sub-problem solver*, EURO Journal on Computational Optimization, 10 (2022), p. 100035.
[46] F. Roosta-Khorasani and M. W. Mahoney, *Sub-sampled Newton methods*, Mathematical Programming, 174 (2019), pp. 293–326.
[47] Y. Sun, H. Jeong, J. Nutini, and M. Schmidt, *Are we there yet? manifold identification of gradient-related proximal methods*, in The 22nd International Conference on Artificial Intelligence and Statistics, PMLR, 2019, pp. 1110–1119.
[48] N. Tripuraneni, M. Stern, C. Jin, J. Regier, and M. I. Jordan, *Stochastic cubic regularization for fast nonconvex optimization*, Advances in Neural Information Processing Systems, 31 (2018).
[49] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher, *Fixed-rank approximation of a positive-semidefinite matrix from streaming data*, Advances in Neural Information Processing Systems, 30 (2017).

[50] J. Vanschoren, J. N. Van Rijn, B. Bischl, and L. Torgo, *Openml: networked science in machine learning*, ACM SIGKDD Explorations Newsletter, 15 (2014), pp. 49–60.
[51] 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.
[52] L. Xiao and T. Zhang, *A proximal stochastic gradient method with progressive variance reduction*, SIAM Journal on Optimization, 24 (2014), pp. 2057–2075.
[53] 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.
[54] Z. Yao, P. Xu, F. Roosta, and M. W. Mahoney, *Inexact nonconvex Newton-type methods*, INFORMS Journal on Optimization, 3 (2021), pp. 154–182.
[55] Z. Yao, P. Xu, F. Roosta, S. J. Wright, and M. W. Mahoney, *Inexact newton-cg algorithms with complexity guarantees*, IMA Journal of Numerical Analysis, 43 (2023), pp. 1855–1897.
[56] H. Ye, L. Luo, and Z. Zhang, *Approximate Newton methods*, Journal of Machine Learning Research, 22 (2021), pp. 1–41.
[57] R. Yuan, A. Lazaric, and R. M. Gower, *Sketched Newton–Raphson*, SIAM Journal on Optimization, 32 (2022), pp. 1555–1583.

SUPPLEMENTARY MATERIALS: SAPPHIRE: Preconditioned Stochastic Variance Reduction for Faster Large-Scale Statistical Learning*

Jingruo Sun†, Zachary Frangella*, and Madeleine Udell*

SM1. Computing randomized Nyström approximation. We propose the following algorithm of randomized low-rank approximation to assist the construction of Nyström preconditioner in Section 3.

Algorithm SM1.1 RandNysApprox

Input: Orthogonalized test matrix $\Omega \in \mathbb{R}^{p\times r_H}$, $r_H = \mathrm{rank}(H_{S_H})$,
Sketch matrix $M = \widehat{\nabla}^2 L(w)\Omega \in \mathbb{R}^{p\times r_H}$
Compute shift $\nu = \sqrt{p}\cdot \mathrm{eps}(\sigma_{\max}(M))$
$M_\nu = M + \nu\Omega$
Cholesky decomposition $C = \mathrm{chol}(\Omega^\top M_\nu)$
Thin SVD $[\widehat{V}, \Sigma, \sim] = \mathrm{svd}(MC^{-1}, 0)$
$\widehat{\Lambda} = \max\{0, \Sigma^2 - \nu I\}$
return $\widehat{V}, \widehat{\Lambda}$

Algorithm SM1.1 provides the Hessian approximation and construct the Nyström preconditioner in (3.3) as $P = \widehat{V}\widehat{\Lambda}\widehat{V}^\top$. Here the function $\mathrm{eps}(\cdot)$ represents the positive distance to the next largest floating point number of the same precision. All eigenvalues of the approximation are non-negative. We apply it in conjunction with a regularizer to ensure positive definiteness.

SM2. Stochastic linesearch. Recently, [SM9] developed a version of Armijo line search for the stochastic proximal gradient method. Inspired by this work, we propose a stochastic version of Armijo line search (SLS) [SM9] to update the learning rate in the composite optimization problem, as shown in Algorithm SM2.1. However, there are two important differences from the method in [SM9]: (i) Algorithm SM2.1 only evaluates the minibatch loss instead of the full loss and (ii) Algorithm SM2.1 uses the preconditioned norm rather than the Euclidean norm to determine the stepsize. Algorithm SM2.1 also includes adds a learning rate ceiling $\eta_{\max}$ and a learning rate floor $\eta_{\min}$, this ensures the learning rate never becomes too large or too small. We recommend using $\eta_{\max} = 1$ and $\eta_{\min} = 0.05$.

Figure SM1 shows the result of applying SLS to the problems in Subsection 4.3 used to verify the convergence of `SAPPHIRE`. Figure SM1 shows that `SAPPHIRE` with SLS exhibits the

*Submitted to the editors June 10th, 2025.
Funding: MU, JS, and ZF gratefully acknowledge support from the National Science Foundation (NSF) Award IIS-2233762, the Office of Naval Research (ONR) Awards N000142212825, N000142412306, and N000142312203, the Alfred P. Sloan Foundation, and from IBM Research as a founding member of Stanford Institute for Human-centered Artificial Intelligence (HAI).
†Department of Management Science and Engineering, Stanford University, CA (jingruo@stanford.edu, zfran@stanford.edu, udell@stanford.edu).

Algorithm SM2.1 Stochastic Line Search (SLS) for Learning Rate

Input: initial learning rate η_0, maximum learning rate $\eta_{\max}$, minimum learning rate $\eta_{\min}$, preconditioner $P_k^{(s)}$, gradient batch S_g with size b_g,
gradient estimate $v_k^{(s)}$, current and previous iterates $w_{k+1}^{(s)}$ and $w_k^{(s)}$,
loss function ℓ, and regularization function r

Initialize: coefficient $\gamma \in (0,1)$

if $\frac{1}{b_g}\sum_{i\in S_g}\ell_i(w_{k+1}^{(s)}) \leq \frac{1}{b_g}\sum_{i\in S_g}\ell_i(w_k^{(s)}) + \langle v_k^{(s)}, w_{k+1}^{(s)} - w_k^{(s)}\rangle + \frac{1}{2\eta_s}\|w_{k+1}^{(s)} - w_k^{(s)}\|^2_{P_k^{(s)}}$ **then**
  Update $\eta^{(s+1)} = \min\left\{\frac{1}{\gamma}\eta^{(s)}, \eta_{\max}\right\}$
else
  Update $\eta^{(s+1)} = \max\left\{\gamma\eta^{(s)}, \eta_{\min}\right\}$
end if

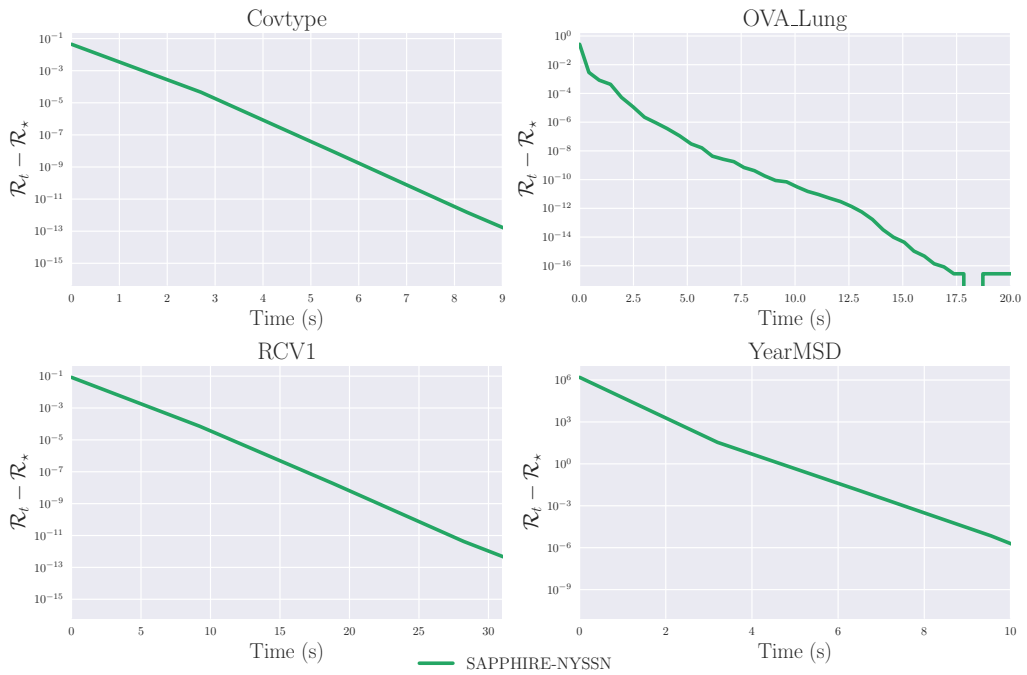

Figure SM1. *Verification of convergence of* `SAPPHIRE`*.* `SAPPHIRE` *exhibits linear convergence on all four datasets, consistent with the theory and discussion presented in Section 4.*

same linear convergence as in Figure 4, indicating that Algorithm SM2.1 provides a reliable strategy for setting the learning rate.

SM3. Proof for Lemma 4.6.

Proof. By Proposition 3.16 in [SM3], it holds that

$$\mathbb{E}\|\hat{\nabla}L(w) - \hat{\nabla}L(w^\star)\|^2_{P^{-1}} \leq 2\mathcal{L}_P\left(L(w) - L(w^\star) - \langle \nabla L(w^\star), w - w^\star\rangle\right).$$

Now, by the optimality of $w^\star = \arg\min_w\{L(w) + r(w)\}$, there exists $\xi^\star \in \partial r(w^\star)$ such

that $\nabla L(w^\star) + \xi^\star = 0$. Thus, by the convexity of $r(w)$, we deduce

$$\begin{aligned} L(w) - L(w^\star) - \langle \nabla L(w^\star), w - w^\star \rangle &= L(w) - L(w^\star) + \langle \xi^\star, w - w^\star \rangle \\ &\le L(w) - L(w^\star) + r(w) - r(w^\star) \\ &= \mathcal{R}(w) - \mathcal{R}(w^\star). \end{aligned}$$

Combining these two results,

$$\mathbb{E}\|\hat{\nabla} L(w) - \hat{\nabla} L(w^\star)\|_{P^{-1}}^2 \le 2\mathcal{L}_P[\mathcal{R}(w) - \mathcal{R}(w^\star)]. \qquad \blacksquare$$

SM4. Proof for Lemma 4.7. First, we calculate the expectation of $v_k^{(s)}$ as

$$\begin{aligned} \mathbb{E}[v_k^{(s)}] &= \mathbb{E}[\hat{\nabla} L(w_k^{(s)})] - \mathbb{E}[\hat{\nabla} L(\hat{w}^{(s)})] + \nabla L(\hat{w}^{(s)}) \\ &= \nabla L(w_k^{(s)}) - \nabla L(\hat{w}^{(s)}) + \nabla L(\hat{w}^{(s)}) \\ &= \nabla L(w_k^{(s)}). \end{aligned}$$

Building on Lemma 4.6, we derive

$$\begin{aligned} \mathbb{E}\|v_k^{(s)} - \nabla L(w_k^{(s)})\|_{(P_k^{(s)})^{-1}}^2 &= \mathbb{E}\|\hat{\nabla} L(w_k^{(s)}) - \hat{\nabla} L(\hat{w}^{(s)}) + \nabla L(\hat{w}^{(s)}) - \nabla L(w_k^{(s)})\|_{(P_k^{(s)})^{-1}}^2 \\ &\le \mathbb{E}\|\hat{\nabla} L(w_k^{(s)}) - \hat{\nabla} L(\hat{w}^{(s)})\|_{(P_k^{(s)})^{-1}}^2 \\ &\quad - \|\nabla L(w_k^{(s)}) - \nabla L(\hat{w}^{(s)})\|_{(P_k^{(s)})^{-1}}^2 \\ &\le \mathbb{E}\|\hat{\nabla} L(w_k^{(s)}) - \hat{\nabla} L(\hat{w}^{(s)})\|_{(P_k^{(s)})^{-1}}^2 \\ &\le 2\mathbb{E}\|\hat{\nabla} L(w_k^{(s)}) - \hat{\nabla} L(w^\star)\|_{(P_k^{(s)})^{-1}}^2 \\ &\quad + 2\mathbb{E}\|\hat{\nabla} L(\hat{w}^{(s)}) - \hat{\nabla} L(w^\star)\|_{(P_k^{(s)})^{-1}}^2 \\ &\le 4\mathcal{L}_P[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w^\star) + \mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)]. \end{aligned}$$

Here, the first inequality uses $\mathbb{E}\|X - \mathbb{E}X\|_A^2 \le \mathbb{E}\|X\|_A^2$, which is valid for any random variable $X \in \mathbb{R}^d$ and symmetric positive definite matrix A. The third inequality uses $\|a+b\|_A^2 \le 2(\|a\|_A^2 + \|b\|_A^2)$. The last inequality applies Lemma 4.6 twice.

SM5. A technical lemma. We need the following technical result to establish global linear convergence of `SAPPHIRE`, which extends [SM13, Lemma 3] to the preconditioned setting.

Lemma SM5.1. *Let $L(w)$ be quadratically regular and $r(w)$ be convex. For any $w \in dom(r)$ and arbitrary $v \in \mathbb{R}^d$, define $\tilde{w} = \mathrm{prox}_{\eta r}^P(w - \eta P^{-1} v)$, $g_P = \frac{1}{\eta} P(w - \tilde{w})$, and $\Delta = v - \nabla L(w)$, where $0 < \eta \le \frac{1}{(1+\zeta)\gamma_u}$. Then we have for any $w' \in \mathbb{R}^p$,*

$$\mathcal{R}(w') \ge \mathcal{R}(\tilde{w}) + \langle g_P, w' - w \rangle + \frac{\eta}{2}\|g_P\|_{P^{-1}}^2 + \frac{(1-\zeta)\gamma_\ell}{2}\|w' - w\|_P^2 + \langle \Delta, \tilde{w} - w' \rangle.$$

Proof. We write the proximal update $\tilde{w}$ explicitly as

$$\begin{aligned}\tilde{w} &= \operatorname{prox}_{\eta r}^{P}(w - \eta P^{-1} v) \\ &= \arg\min_{w'} \left\{ \frac{1}{2}\|w' - (w - \eta P^{-1} v)\|_P^2 + \eta r(w') \right\}.\end{aligned}$$

The associated optimality condition states that there exists a $\xi \in \partial r(\tilde{w})$ such that

$$P\left(\tilde{w} - (w - \eta P^{-1} v)\right) + \eta \xi = 0.$$

and we note that $g_P = P(w - \tilde{w})/\eta$, so we have $\xi = g_P - v$.

Applying quadratic regularity of L, we can lower bound $L(w)$ by

$$\begin{aligned}L(w) &\geq L(\tilde{w}) - \langle \nabla L(w), \tilde{w} - w \rangle - \frac{(1+\zeta)\gamma_u}{2}\|\tilde{w} - w\|_P^2 \\ &\geq L(\tilde{w}) - \langle \nabla L(w), \tilde{w} - w \rangle - \frac{1}{2\eta}\|\tilde{w} - w\|_P^2.\end{aligned}$$

By the lower quadratic regularity of L and convexity of r, we have for any $w \in \operatorname{dom}(\mathrm{r})$ and $w' \in \mathbb{R}^d$,

$$\begin{aligned}\mathcal{R}(w') &= L(w') + r(w') \\ &\geq L(w) + \nabla L(w)^\top (w' - w) + \frac{(1-\zeta)\gamma_\ell}{2}\|w' - w\|_P^2 + R(\tilde{w}) + \xi^\top (w' - \tilde{w}) \\ &\geq L(\tilde{w}) - \nabla L(w)^\top (\tilde{w} - w) - \frac{1}{2\eta}\|\tilde{w} - w\|_P^2 \\ &\quad + \nabla L(w)^\top (w' - w) + \frac{(1-\zeta)\gamma_\ell}{2}\|w' - w\|_P^2 + r(\tilde{w}) + \xi^\top (w' - \tilde{w}) \\ &= \mathcal{R}(\tilde{w}) + \nabla L(w)^\top (w' - \tilde{w}) + \xi^\top (w' - \tilde{w}) - \frac{1}{2\eta}\|\tilde{w} - w\|_P^2 + \frac{(1-\zeta)\gamma_\ell}{2}\|w' - w\|_P^2.\end{aligned}$$

Note that $g_P = \frac{1}{\eta} P(w - \tilde{w})$, so we have

$$\frac{1}{2\eta}\|\tilde{w} - w\|_P^2 = \frac{1}{2\eta} \cdot \eta^2 \langle P^{-1} g_P, P(P^{-1} g_P) \rangle = \frac{\eta}{2}\langle g_P, P^{-1} g_P \rangle = \frac{\eta}{2}\|g_P\|_{P^{-1}}^2.$$

Collect all the inner products on the right-hand-side and denote $\Delta = v - \nabla L(w)$, we have

$$\begin{aligned}&\langle \nabla L(w), w' - \tilde{w} \rangle + \langle \xi, w' - \tilde{w} \rangle \\ &= \langle \nabla L(w), w' - \tilde{w} \rangle + \langle g_P - v, w' - \tilde{w} \rangle \\ &= \langle g_P, w' - \tilde{w} \rangle + \langle v - \nabla L(w), \tilde{w} - w' \rangle \\ &= \langle g_P, w' - w + w - \tilde{w} \rangle + \langle \Delta, \tilde{w} - w' \rangle \\ &= \langle g_P, w' - w \rangle + \langle g_P, \eta P^{-1} g_P) \rangle + \langle \Delta, \tilde{w} - w' \rangle \\ &= \langle g_P, w' - w \rangle + \eta \|g_P\|_{P^{-1}}^2 + \langle \Delta, \tilde{w} - w' \rangle.\end{aligned}$$

Plugging the derivation of $\frac{1}{2\eta}\|\tilde{w}-w\|_P^2$ and $\langle \nabla L(w), w'-\tilde{w}\rangle + \langle \xi, w'-\tilde{w}\rangle$ back for $\mathcal{R}(w')$, we obtain

$$\begin{aligned}\mathcal{R}(w') &\geq \mathcal{R}(\tilde{w}) + \langle \nabla L(w), w'-\tilde{w}\rangle + \langle \xi, w'-\tilde{w}\rangle - \frac{1}{2\eta}\|\tilde{w}-w\|_P^2 + \frac{(1-\zeta)\gamma_\ell}{2}\|w'-w\|_P^2 \\ &\geq \mathcal{R}(\tilde{w}) + \langle g_P, w'-w\rangle + \eta\|g_P\|_{P^{-1}}^2 + \langle \Delta, \tilde{w}-w'\rangle - \frac{\eta}{2}\|g_P\|_{P^{-1}}^2 + \frac{(1-\zeta)\gamma_\ell}{2}\|w'-w\|_P^2 \\ &= \mathcal{R}(\tilde{w}) + \langle g_P, w'-w\rangle + \frac{\eta}{2}\|g_P\|_{P^{-1}}^2 + \frac{(1-\zeta)\gamma_\ell}{2}\|w'-w\|_P^2 + \langle \Delta, \tilde{w}-w'\rangle. \end{aligned}$$

■

SM6. Proof of Lemma A.1.

Proof. Define the stochastic gradient mapping

$$\widehat{G}_k^{(s)} = \frac{1}{\eta}\left(w_k^{(s)} - w_{k+1}^{(s)}\right) = \frac{1}{\eta}\left(w_k^{(s)} - \mathrm{prox}_{\eta r}^{P}\left(w_k^{(s)} - \eta P_k^{(s)^{-1}} v_k^{(s)}\right)\right),$$

so the proximal gradient step can be written as

$$w_{k+1}^{(s)} = w_k^{(s)} - \eta \widehat{G}_k^{(s)}.$$

Moreover, we define

$$\tilde{p}_k^{(s)} := \left(P_k^{(s)}\right)^{-1} v_k^{(s)}, \quad p_k^{(s)} := \left(P_k^{(s)}\right)^{-1} \nabla F(w_k^{(s)}).$$

Applying the previous relation, we deduce that

$$\begin{aligned}\|w_{k+1}^{(s)} - w^\star\|_{P_k^{(s)}}^2 &= \|w_k^{(s)} - \eta\widehat{G}_k^{(s)} - w^\star\|_{P_k^{(s)}}^2 \\ &= \|w_k^{(s)} - w^\star\|_{P_k^{(s)}}^2 - 2\eta\langle \widehat{G}_k^{(s)}, w_k^{(s)} - w^\star\rangle_{P_k^{(s)}} + \eta^2\|\widehat{G}_k^{(s)}\|_{P_k^{(s)}}^2.\end{aligned}$$

Note that our assumptions guarantee $\eta < \frac{1}{4\mathcal{L}_P}$. Applying Lemma SM5.1 with $w = w_k^{(s)}, v = v_k^{(s)}, \tilde{w} = w_{k+1}^{(s)}, g_P = P_k^{(s)}\widehat{G}_k^{(s)}, w' = w^\star$ and $\Delta_k^{(s)} = v_k^{(s)} - \nabla L(w_k^{(s)})$, we have

$$\begin{aligned}&-\langle \widehat{G}_k^{(s)}, w_k^{(s)} - w^\star\rangle_{P_k^{(s)}} + \frac{\eta}{2}\|\widehat{G}_k^{(s)}\|_{P_k^{(s)}}^2 \\ &\leq \mathcal{R}(w^\star) - \mathcal{R}(w_{k+1}^{(s)}) - \frac{(1-\zeta)\gamma_\ell}{2}\|w^\star - w_k^{(s)}\|_{P_k^{(s)}}^2 - \langle \Delta_k^{(s)}, w_{k+1}^{(s)} - w^\star\rangle.\end{aligned}$$

This property of gradient mapping derives the iteration that

$$\begin{aligned}\|w_{k+1}^{(s)} - w^\star\|_{P_k^{(s)}}^2 &\leq \|w_k^{(s)} - w^\star\|_{P_k^{(s)}}^2 - \eta(1-\zeta)\gamma_\ell\|w_k^{(s)} - w^\star\|_{P_k^{(s)}}^2 \\ &\quad - 2\eta[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w^\star)] - 2\eta\langle \Delta_k^{(s)}, w_{k+1}^{(s)} - w^\star\rangle \\ &\leq \|w_k^{(s)} - w^\star\|_{P_k^{(s)}}^2 - 2\eta[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w^\star)] - 2\eta\langle \Delta_k^{(s)}, w_{k+1}^{(s)} - w^\star\rangle.\end{aligned}$$

Next, we bound the quantity $-2\eta\langle\Delta_k^{(s)}, w_{k+1}^{(s)} - w^\star\rangle$. Let $\bar{w}_{k+1}^{(s)}$ denote the result of taking a preconditioned proximal gradient step with the full gradient as

$$\bar{w}_{k+1}^{(s)} := \operatorname{prox}_{\eta r}^{P}\left(w_k^{(s)} - \eta p_k^{(s)}\right).$$

Expanding $w_{k+1}^{(s)} - w^\star$ with $\bar{w}_{k+1}^{(s)}$,

$$\begin{aligned}
-2\eta\langle\Delta_k^{(s)}, w_{k+1}^{(s)} - w^\star\rangle &= -2\eta\langle\Delta_k^{(s)}, w_{k+1}^{(s)} - \bar{w}_{k+1}^{(s)}\rangle - 2\eta\langle\Delta_k^{(s)}, \bar{w}_{k+1}^{(s)} - w^\star\rangle \\
&\le 2\eta\|\Delta_k^{(s)}\|_{P_k^{(s)-1}}\|w_{k+1}^{(s)} - \bar{w}_{k+1}^{(s)}\|_{P_k^{(s)}} - 2\eta\langle\Delta_k^{(s)}, \bar{w}_{k+1}^{(s)} - w^\star\rangle \\
&\le 2\eta\|\Delta_k^{(s)}\|_{P_k^{(s)-1}}\left\|\left(w_k^{(s)} - \eta\tilde{p}_k^{(s)}\right) - \left(w_k^{(s)} - \eta p_k^{(s)}\right)\right\|_{P_k^{(s)}} \\
&\quad - 2\eta\langle\Delta_k^{(s)}, \bar{w}_{k+1}^{(s)} - w^\star\rangle \\
&= 2\eta\|\Delta_k^{(s)}\|_{P_k^{(s)-1}}\|\eta {P_k^{(s)}}^{-1}\Delta_k^{(s)}\|_{P_k^{(s)}} - 2\eta\langle\Delta_k^{(s)}, \bar{w}_{k+1}^{(s)} - w^\star\rangle \\
&= 2\eta^2\|\Delta_k^{(s)}\|_{P_k^{(s)-1}}^2 - 2\eta\langle\Delta_k^{(s)}, \bar{w}_{k+1}^{(s)} - w^\star\rangle
\end{aligned}$$

Here, we use Cauchy-Schwarz inequality for the first inequality and non-expansiveness of proximal mapping for the second inequality.

Combining with the previous result, we have

$$\begin{aligned}
\|w_{k+1}^{(s)} - w^\star\|_{P_k^{(s)}}^2 \le \|w_k^{(s)} - w^\star\|_{P_k^{(s)}}^2 &- 2\eta[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w^\star)] \\
&+ 2\eta^2\|\Delta_k^{(s)}\|_{P_k^{(s)-1}}^2 - 2\eta\langle\Delta_k^{(s)}, \bar{w}_{k+1}^{(s)} - w^\star\rangle.
\end{aligned}$$

Taking the expectation over $v_k^{(s)}$ of both sides of the preceding display and applying Lemma 4.7 obtains

$$\begin{aligned}
\mathbb{E}\left[\|w_{k+1}^{(s)} - w^\star\|_{P_k^{(s)}}^2\right] &= \|w_k^{(s)} - w^\star\|_{P_k^{(s)}}^2 - 2\eta\mathbb{E}[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w^\star)] \\
&\quad + 2\eta^2\mathbb{E}\left[\|v_k^{(s)} - \nabla L(w_k^{(s)})\|_{P_k^{(s)-1}}^2\right] \\
&\le \|w_k^{(s)} - w^\star\|_{P_k^{(s)}}^2 - 2\eta\mathbb{E}[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w^\star)] \\
&\quad + 8\mathcal{L}_P\eta^2[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w^\star) + \mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)].
\end{aligned}$$

Rearranging the last display, we conclude the desired result. ∎

SM7. SAPPHIRE: Sublinear convergence analysis. We now prove Theorem 4.11, which establishes global sublinear convergence of SAPPHIRE under ρ-weak quadratic regularity, which covers the setting when $L(w)$ is only smooth and convex.

Proof. Assume we are in outer iteration s, then summing the bound in Lemma A.1 yields

$$\mathbb{E}[\|w_k^{(m)} - w_\star\|^2_{P_k^{(s)}}] + 2\eta\mathbb{E}[\mathcal{R}(w_{k+1}^{(s)}) - \mathcal{R}(w_\star)] + 2\eta(1 - 4\eta\mathcal{L}_P)\sum_{k=1}^{m-1}\mathbb{E}[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w_\star)]$$
$$\leq \|\hat{w}^{(s)} - w_\star\|^2_{P_k^{(s)}} + 8(m+1)\eta^2\mathcal{L}_P(\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)).$$

As $\eta = \min\{\frac{1}{4\mathcal{L}_P(m+2)}, \frac{1}{8(m+2)}\}$ we have that $2\eta\,(1 - 4\eta\mathcal{L}_P) \geq \eta^2$. Thus,

$$\mathbb{E}[\|\hat{w}^{(s+1)} - w_\star\|^2_{P_k^{(s)}}] + (2\eta - \eta^2)\mathbb{E}[\mathcal{R}(\hat{w}^{(s+1)}) - \mathcal{R}(w_\star)] + \eta^2\sum_{k=1}^{m}\mathbb{E}[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w_\star)]$$
$$\leq \|\hat{w}^{(s)} - w_\star\|^2_{P_k^{(s)}} + 8(m+1)\eta^2\mathcal{L}_P(\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star))$$
$$\leq \|\hat{w}^{(s)} - w_\star\|^2_{P_k^{(s)}} + (2\eta - \eta^2)(\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star)),$$

where in the last inequality, we used that value of η implies that $2\eta - \eta^2 \geq 8(m+1)\eta^2\mathcal{L}_P$. Thus, the preceding display can be rearranged to yield

$$\eta^2\sum_{k=1}^{m}\mathbb{E}[\mathcal{R}(w_k^{(s)}) - \mathcal{R}(w_\star)] \leq \|\hat{w}^{(s)} - w_\star\|^2_{P_k^{(s)}} + (2\eta - \eta^2)(\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star))$$
$$- \mathbb{E}[\|\hat{w}^{(s+1)} - w_\star\|^2_{P_k^{(s)}}] - (2\eta - \eta^2)\mathbb{E}[\mathcal{R}(\hat{w}^{(s+1)}) - \mathcal{R}(w_\star)].$$

Using convexity of $\mathcal{R}$ this becomes

$$m\eta^2\mathbb{E}\left[\mathcal{R}\left(\frac{1}{m}\sum_{k=1}^{m}w_k^{(s)}\right) - \mathcal{R}(w_\star)\right] \leq \|\hat{w}^{(s)} - w_\star\|^2_{P_k^{(s)}} - \mathbb{E}[\|\hat{w}^{(s+1)} - w_\star\|^2_{P_k^{(s)}}]$$
$$+ (2\eta - \eta^2)\left[\mathcal{R}(\hat{w}^{(s)}) - \mathcal{R}(w^\star) - \mathbb{E}[\mathcal{R}(\hat{w}^{(s+1)}) - \mathcal{R}(w_\star)]\right].$$

Taking the total expectation, summing over all S outer iterations, and using convexity of R yields

$$mS\eta^2\mathbb{E}\left[\mathcal{R}\left(\frac{1}{Sm}\sum_{s=0}^{S-1}\sum_{k=1}^{m}\hat{w}_k^{(s)}\right) - \mathcal{R}(w_\star)\right] \leq \|w_0 - w_\star\|^2_{P_0^{(0)}} + (2\eta - \eta^2)\left(\mathcal{R}(w_0) - \mathcal{R}(w_\star)\right).$$

Define $\bar{w}$ as $\frac{1}{Sm}\sum_{s=0}^{S-1}\sum_{k=1}^{m}\hat{w}_k^{(s)}$. Rearranging, we find that

$$\mathbb{E}\left[\mathcal{R}(\bar{w}) - \mathcal{R}(w_\star)\right] \leq \frac{1}{\eta^2 mS}\|w_0 - w_\star\|^2_{P_0^{(0)}} + \frac{1}{mS}\left(\frac{1}{\eta} - 1\right)\left(\mathcal{R}(w_0) - \mathcal{R}(w_\star)\right).$$

Using the identity $\frac{1}{\min\{a,b\}} \leq 1/a + 1/b$ for $a, b > 0$ yields

$$\mathbb{E}\left[\mathcal{R}(\bar{w}) - \mathcal{R}(w_\star)\right] \leq \frac{(16\mathcal{L}_P^2+64)(m+2)^2}{mS}\|w_0 - w_\star\|^2_{P_0^{(0)}} + \frac{(4\mathcal{L}_P+8)(m+2)}{mS}\left(\mathcal{R}(w_0) - \mathcal{R}(w_\star)\right)$$
$$\leq \frac{3(16\mathcal{L}_P^2+64)(m+2)}{S}\|w_0 - w_\star\|^2_{P_0^{(0)}} + \frac{3(4\mathcal{L}_P+8)}{S}\left(\mathcal{R}(w_0) - \mathcal{R}(w_\star)\right).$$

Thus, setting $S = \mathcal{O}\left(\frac{m\mathcal{L}_P^2}{\varepsilon}\right)$ yields

$$\mathbb{E}\left[\mathcal{R}(\bar{w}) - \mathcal{R}(w_\star)\right] \leq \epsilon \left(\|w_0 - w_\star\|_{P_0^{(0)}}^2 + (\mathcal{R}(w_0) - \mathcal{R}(w_\star)) \right). \qquad \blacksquare$$

SM8. SAPPHIRE: Local convergence analysis. In this section, we prove Theorem 4.12, which shows local condition number-free convergence of SAPPHIRE in the neighborhood

$$\mathcal{N}_{\varepsilon_0}(w_\star) = \left\{ w \in \mathbb{R}^p : \|w - w_\star\|_{\nabla^2 F(w_\star)} \leq \frac{\varepsilon_0 \nu^{3/2}}{2M} \right\}.$$

The overall proof strategy is similar to that of other approximate Newton methods. Namely, we first show that the iterates remain within $\mathcal{N}_{\varepsilon_0}(w_\star)$, where the quadratic regularity constants are close to unity. Once this has been established, we argue that the output of each stage of Algorithm 3.1 contracts to the optimum at a condition number-free rate.

SM8.1. Preliminaries. We begin by recalling the following technical lemma from [SM3], which shows the following items hold in $\mathcal{N}_{\varepsilon_0}(w_\star)$: (1) the quadratic regularity constants are close to unity, (2) the Hessians are uniformly close in the Loewner ordering, (3) taking an exact Newton step moves the iterate closer to the optimum in the Hessian norm, (4) $\nabla F_i(w)$, $\nabla F(w)$ are $(1+\varepsilon_0)$ Lipschitz in $\mathcal{N}_{\varepsilon_0}(w_\star)$.

Lemma SM8.1. *Let $w, w' \in \mathcal{N}_{\varepsilon_0}(w_\star)$, and suppose P is a ε_0-spectral approximation constructed at some $w_0 \in \mathcal{N}_{\varepsilon_0}(w_\star)$, then the following items hold.*

1.
$$\frac{1}{1+\varepsilon_0} \leq \gamma_{l_{\min}}(\mathcal{N}_{\varepsilon_0}(w_\star)) \leq \gamma_{u_{\max}}(\mathcal{N}_{\varepsilon_0}(w_\star)) \leq (1+\varepsilon_0).$$

2.
$$(1-\varepsilon_0)\nabla^2 L(w) \preceq \nabla^2 L(w') \preceq (1+\varepsilon_0)\nabla^2 L(w).$$

3.
$$\|w - w_\star - \nabla^2 L(w)^{-1}(\nabla L(w) - \nabla L(w_\star)\|_{\nabla^2 L(w)} \leq \varepsilon_0 \|w - w_\star\|_{\nabla^2 L(w)}.$$

4.
$$\|\nabla L_i(w) - \nabla L_i(w_\star)\|_{\nabla^2 L_i(w')^{-1}} \leq (1+\varepsilon_0)\|w - w_\star\|_{\nabla^2 L_i(w')}, \quad \textit{for all } i \in [n],$$
$$\|\nabla L(w) - \nabla L(w_\star)\|_{\nabla^2 L(w')^{-1}} \leq (1+\varepsilon_0)\|w - w_\star\|_{\nabla^2 F(w')}.$$

SM8.2. Controlling the error in the stochastic gradient. Similar to the global convergence analysis, it is essential that the deviation of the variance-reduced gradient from the exact gradient goes to zero as we approach $w_\star$. Thus, our analysis begins with the following lemma, which gives a high probability bound for the preconditioned gradient error. It provides a local analog of Lemma 4.7.

Lemma SM8.2. *Let $\beta_g \in (0,1)$. If $w_k^{(s)} \in \mathcal{N}_{\varepsilon_0}(w_\star)$ and $v_k^{(s)}$ is constructed with batchsize $b_g = \mathcal{O}\left(\frac{\tau_\star^\nu(\mathcal{N}_{\varepsilon_0}(w_\star))\log(\frac{1}{\delta})}{\beta_g^2}\right)$, then with probability at least $1-\delta$*

$$\|v_k^{(s)} - \nabla L(w_k^{(s)})\|_{P^{-1}} \leq \beta_g \left(\|w_k^{(s)} - w_\star\|_P + \|\hat{w}^{(s)} - w_\star\|_P\right).$$

Proof. Let $X_i = \nabla^2 L(w_\star)^{-1/2}\left(\nabla L_i(w_k^{(s)}) - \nabla L_i(\hat{w}^{(s)}) - \left(\nabla L(w_k^{(s)}) - \nabla L(\hat{w}^{(s)})\right)\right)$. By definition of X_i,

$$\nabla^2 L(w_\star)^{-1/2}\left(v_k^{(s)} - \nabla L(w_k^{(s)})\right) = \frac{1}{b_g}\sum_{i\in\mathcal{B}} X_i := X.$$

Observe that $\|X\| = \|v_k^{(s)} - \nabla L(w_k^{(s)})\|_{\nabla^2 L(w_\star)^{-1}}$, and $\mathbb{E}[X] = 0$ by definition of the variance-reduced gradient. Therefore, we can control $\|v_k^{(s)} - \nabla L(w_k^{(s)})\|_{\nabla^2 L(w_\star)^{-1}}$ by a concentration argument similar to [SM3]. We can then convert the result to the (P^{-1}, P)-dual norm pair by applying Lemma SM8.1.

We shall use Bernstein's inequality for vectors to bound $\|X\|$ with high probability. In order to apply this variant of Bernstein's inequality, we must establish bounds on $\|X_i\|$ and $\mathbb{E}\|X_i\|^2$. We begin by bounding $\|X_i\|$. To this end, observe that,

$$\begin{aligned}
\|X_i\|^2 &\overset{(1)}{\leq} 2\|\nabla L_i(w_k^{(s)}) - \nabla L_i(\hat{w}^{(s)})\|^2_{\nabla^2 L(w_\star)^{-1}} + 2\|\nabla L(w_k^{(s)}) - \nabla L(\hat{w}^{(s)})\|^2_{\nabla^2 L(w_\star)^{-1}} \\
&\overset{(2)}{\leq} 4\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))^2(1+\varepsilon_0)^2\|w_k^{(s)} - \hat{w}^{(s)}\|^2_{\nabla^2 L(w_\star)} \\
&\leq 8\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))^2(1+\varepsilon_0)^2\left(\|w_k^{(s)} - w_\star\|^2_{\nabla^2 L(w_\star)} + \|\hat{w}^{(s)} - w_\star\|^2_{\nabla^2 L(w_\star)}\right).
\end{aligned}$$

Here (1) uses $\|x+y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$, and (2) uses Lemma 3.3 and item 4 of Lemma SM8.1. Taking the square root on both sides yields

$$\|X_i\| \leq 2\sqrt{2}\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))(1+\varepsilon_0)\left(\|w_k^{(s)} - w_\star\|_{\nabla^2 L(w_\star)} + \|\hat{w}^{(s)} - w_\star\|_{\nabla^2 L(w_\star)}\right).$$

This establishes the required bound on $\|X_i\|$. We now turn to bounding $\mathbb{E}\|X_i\|^2$. To begin, observe that an argument similar to the one in Lemma 4.7 yields

$$\mathbb{E}\|X_i\|^2 \leq 2\mathbb{E}\|\nabla L_i(w_k^{(s)}) - \nabla L_i(w_\star)\|_{\nabla^2 L(w_\star)^{-1}} + 2\mathbb{E}\|\nabla L_i(\hat{w}^{(s)}) - \nabla L_i(w_\star)\|_{\nabla^2 L(w_\star)^{-1}}.$$

Again using Lemma 3.3 and Lemma SM8.1, we obtain

$$\begin{aligned}
&2\mathbb{E}\|\nabla L_i(w_k^{(s)}) - \nabla L_i(w_\star)\|_{\nabla^2 L(w_\star)^{-1}} + 2\mathbb{E}\|\nabla L_i(\hat{w}^{(s)}) - \nabla L_i(w_\star)\|_{\nabla^2 L(w_\star)^{-1}} \\
&\le 2\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))\mathbb{E}\|\nabla L_i(w_k^{(s)}) - \nabla L_i(w_\star)\|_{\nabla^2 L_i(w_\star)^{-1}} \\
&\quad + 2\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))\mathbb{E}\|\nabla L_i(\hat{w}^{(s)}) - \nabla L_i(w_\star)\|_{\nabla^2 L_i(w_\star)^{-1}} \\
&\le 2\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))(1+\varepsilon_0)\mathbb{E}\left(L_i(w_k^{(s)}) - L_i(w_\star) - \langle \nabla L_i(w_\star), w_k^{(s)} - w_\star\rangle\right) \\
&\quad + 2\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))(1+\varepsilon_0)\mathbb{E}\left(L_i(\hat{w}^{(s)}) - L_i(w_\star) - \langle \nabla L_i(w_\star), \hat{w}^{(s)} - w_\star\rangle\right) \\
&= 2\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))(1+\varepsilon_0)\left(L(w_k^{(s)}) - L(w_\star) - \langle \nabla L(w_\star), w_k^{(s)} - w_\star\rangle\right) \\
&\quad + 2\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))(1+\varepsilon_0)\left(L(\hat{w}^{(s)}) - L(w_\star) - \langle \nabla L(w_\star), \hat{w}^{(s)} - w_\star\rangle\right) \\
&\le 2\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))(1+\varepsilon_0)^2\left(\|w_k^{(s)} - w_\star\|_{\nabla^2 L(w_\star)} + \|\hat{w}^{(s)} - w_\star\|_{\nabla^2 L(w_\star)}\right).
\end{aligned}$$

Hence, the scaled gradient residual X_i satisfies

$$\mathbb{E}\|X_i\|^2 \le 2\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))(1+\varepsilon_0)^2\left(\|w_k^{(s)} - w_\star\|_{\nabla^2 L(w_\star)} + \|\hat{w}^{(s)} - w_\star\|_{\nabla^2 L(w_\star)}\right).$$

After giving the bound of $\|X_i\|$ and $\mathbb{E}\|X_i\|^2$, we can apply Lemma 27 from [SM3] with $b_g = \mathcal{O}\left(\frac{\tau_\star(\mathcal{N}_{\varepsilon_0}(w_\star))\log\left(\frac{1}{\delta}\right)}{\beta_g^2}\right)$ to reach

$$\|v_k^{(s)} - \nabla L(w_k^{(s)})\|_{\nabla^2 F(w_\star)^{-1}} \le \frac{\beta_g}{4}\left(\|w_k^{(s)} - w_\star\|_{\nabla^2 F(w_\star)} + \|\hat{w}^{(s)} - w_\star\|_{\nabla^2 F(w_\star)}\right).$$

Converting to preconditioned norms via Lemma SM8.1, this becomes

$$\|v_k^{(s)} - \nabla L(w_k^{(s)})\|_{P^{-1}} \le \beta_g\left(\|w_k^{(s)} - w_\star\|_P + \|\hat{w}^{(s)} - w_\star\|_P\right).$$

■

SM8.3. Establishing a one iteration contraction. With Lemma SM8.2 in hand, we now establish a contraction relation for iterates in any outer iteration s. This lemma guarantees the SAPPHIRE iterates remain in $\mathcal{N}_{\varepsilon_0}(w_\star)$, essential for showing condition number-free local convergence.

Lemma SM8.3. *Let $w_k^{(s)} \in \mathcal{N}_{\varepsilon_0}(w_\star)$, and $\beta_g \in (0,1)$. Suppose the gradient batchsize satisfies $b_g = \mathcal{O}\left(\frac{\tau_\star^\nu(\mathcal{N}_{\varepsilon_0}(w_\star))\log\left(\frac{k+1}{\delta}\right)}{\beta_g^2}\right)$. Then with probability at least $1 - \frac{\delta}{(k+1)^2}$*

1. $\|\Delta_{k+1}^{(s)}\|_{\nabla^2 F(w_\star)} \le \frac{3}{4}\|\Delta_k^{(s)}\|_{\nabla^2 F(w_\star)} + \frac{7}{48}\|\Delta_0^{(s)}\|_{\nabla^2 F(w_\star)}$
2. $w_{k+1}^{(s)} \in \mathcal{N}_{\varepsilon_0}(w_\star)$.

Proof. Let $\Delta_{k+1}^{(s)} = \operatorname{prox}_r^P\left(w_k^{(s)} - P^{-1}\nabla L(w_k^{(s)})\right) - w_\star$. We begin with the following inequality,

$$\begin{aligned}
\left\|\Delta_{k+1}^{(s)}\right\|_P &= \left\|\operatorname{prox}_r^P\left(w_k - P^{-1}v_k^{(s)}\right) - w_\star\right\|_P \\
&= \left\|\operatorname{prox}_r^P\left(w_k - P^{-1}v_k^{(s)}\right) - \operatorname{prox}_r^P\left(w_\star - P^{-1}\nabla L(w_\star)\right)\right\|_P \\
&\leq \left\|\left(w_k - P^{-1}v_k^{(s)}\right) - \left(w_\star - P^{-1}\nabla L(w_\star)\right)\right\|_P \\
&= \left\|P(w_k - w_\star) - (\nabla L(w_k) - \nabla L(w_\star)) + \nabla L(w_k) - v_k^{(s)}\right\|_{P^{-1}} \\
&\leq \left\|P(w_k^{(s)} - w_\star) - (\nabla L(w_k^{(s)}) - \nabla L(w_\star))\right\|_{P^{-1}} + \left\|v_k^{(s)} - \nabla L(w_k^{(s)})\right\|_{P^{-1}}.
\end{aligned}$$

In the second inequality, we used the non-expansiveness of the scaled proximal mapping. The preceding display consists of two terms. The first term represents the error in the approximate Taylor expansion

$$\nabla L(w_k^{(s)}) - \nabla L(w_\star) \approx P(w_k^{(s)} - w_\star).$$

The second term measures the deviation of the stochastic gradient from the exact gradient. Using Lemma SM8.2, the second term can be bounded as,

$$\beta_g\left(\|\Delta_k^{(s)}\|_P + \|\Delta_0^{(s)}\|_P\right).$$

Thus, we now turn to bounding the Taylor error term. To this end, observe that the triangle inequality yields

$$\begin{aligned}
&\left\|P(w_k^{(s)} - w_\star) - (\nabla L(w_k^{(s)}) - \nabla L(w_\star))\right\|_{P^{-1}} \\
\leq\ &\left\|\nabla^2 L(w_k^{(s)})(w_k^{(s)} - w_\star) - (\nabla L(w_k^{(s)}) - \nabla L(w_\star))\right\|_{P^{-1}} + \left\|(P - \nabla^2 L(w_k^{(s)}))(w_k^{(s)} - w_\star)\right\|_{P^{-1}}.
\end{aligned}$$

The first term in this inequality is the exact Taylor expansion error, while the second term represents the error in approximating the Hessian. We can bound the first term using Lemma SM8.1 as follows,

$$\begin{aligned}
&\left\|\nabla^2 L(w_k^{(s)})(w_k^{(s)} - w_\star) - (\nabla L(w_k^{(s)}) - \nabla L(w_\star))\right\|_{P^{-1}} \\
\overset{(1)}{\leq}\ &\frac{1}{\sqrt{1-\varepsilon_0}}\left\|\nabla^2 L(w_k^{(s)})(w_k^{(s)} - w_\star) - (\nabla L(w_k^{(s)}) - \nabla L(w_\star))\right\|_{\nabla^2 L(w_k^{(s)})^{-1}} \\
=\ &\frac{1}{\sqrt{1-\varepsilon_0}}\|w_k^{(s)} - w_\star - \nabla^2 L(w_k^{(s)})^{-1}(\nabla L(w_k^{(s)}) - \nabla L(w_\star))\|_{\nabla^2 L(w_k^{(s)})} \\
\overset{(2)}{\leq}\ &\frac{\varepsilon_0}{\sqrt{1-\varepsilon_0}}\|\Delta_k^{(s)}\|_{\nabla^2 L(w_k^{(s)})} \\
\overset{(3)}{\leq}\ &\varepsilon_0\sqrt{\frac{1+\varepsilon_0}{1-\varepsilon_0}}\|\Delta_k^{(s)}\|_P \\
\overset{(4)}{\leq}\ &2\varepsilon_0\|\Delta_k^{(s)}\|_P.
\end{aligned}$$

Here (1) uses item 1 of Lemma SM8.1, (2) uses item 2 of Lemma SM8.1, (3) uses item of Lemma SM8.1 again, and (4) uses $\varepsilon_0 \le \frac{1}{6}$.

We can also bound the Hessian approximation error term via Lemma SM8.1. Indeed,

$$\begin{aligned}\left\|\left(P - \nabla^2 L(w_k^{(s)})\right)(w_k^{(s)} - w_\star)\right\|_{P^{-1}} &= \left\|P^{1/2}(I - P^{-1/2}\nabla^2 F(w_k^{(s)})P^{-1/2})P^{1/2}(w_k^{(s)} - w_\star)\right\|_{P^{-1}} \\ &= \left\|(I - P^{-1/2}\nabla^2 F(w_k^{(s)})P^{-1/2})P^{1/2}(w_k^{(s)} - w_\star)\right\| \\ &\le \left\|I - P^{-1/2}\nabla^2 F(w_k^{(s)})P^{-1/2}\right\| \left\|w_k^{(s)} - w_\star\right\|_P \\ &\le \varepsilon_0 \|w_k^{(s)} - w_\star\|_P = \varepsilon_0 \|\Delta_k^{(s)}\|_P,\end{aligned}$$

where the last inequality uses item 2 of Lemma SM8.1. Putting together the two bounds, we find the approximate Taylor error term satisfies

$$\left\|P(w_k^{(s)} - w_\star) - (\nabla L(w_k^{(s)}) - \nabla L(w_\star))\right\|_{P^{-1}} \le 3\varepsilon_0 \|\Delta_k^{(s)}\|_P.$$

Combining the bounds on the approximate Taylor error and the error in the stochastic gradient, we deduce

$$\left\|\Delta_{k+1}^{(s)}\right\|_P \le (\beta_g + 3\varepsilon_0)\|\Delta_k^{(s)}\|_P + \beta_g \|\Delta_0^{(s)}\|_P.$$

Now, converting norms yields

$$\begin{aligned}\left\|\Delta_{k+1}^{(s)}\right\|_{\nabla^2 L(w_\star)} &\le (1+\varepsilon_0)(\beta_g + 3\varepsilon_0)\|\Delta_k^{(s)}\|_{\nabla^2 L(w_\star)} + \beta_g(1+\varepsilon_0)\|\Delta_0^{(s)}\|_{\nabla^2 L(w_\star)} \\ &\le \frac{3}{4}\|\Delta_k^{(s)}\|_{\nabla^2 L(w_\star)} + \frac{7}{48}\|\Delta_0^{(s)}\|_{\nabla^2 L(w_\star)}.\end{aligned}$$

■

SM8.4. Showing convergence for one stage. Now that we have established the iterates produced by `SAPPHIRE` remain in $\mathcal{N}_{\varepsilon_0}(w_\star)$, we can establish the convergence rate for one stage.

Lemma SM8.4 (One-stage analysis). *Let $\hat{w}^{(s)} \in \mathcal{N}_{\varepsilon_0}(w_\star)$. Run Algorithm 3.1 with $m = 10$ inner iterations and gradient batchsize satisfies $b_g = \mathcal{O}\left(\tau_\star^\nu(\mathcal{N}_{\varepsilon_0}(w_\star)) \log\left(\frac{m+1}{\delta}\right)\right)$. Then with probability at least $1 - \delta$,*

1. $\hat{w}^{(s+1)} \in \mathcal{N}_{\frac{2}{3}\varepsilon_0}(w_\star)$.
2. $\|\hat{w}^{(s+1)} - w_\star\|_{\nabla^2 L(w_\star)} \le \frac{2}{3}\|\hat{w}^{(s)} - w_\star\|_{\nabla^2 L(w_\star)}$.

Proof. As $\hat{w}^{(s)} \in \mathcal{N}_{\varepsilon_0}(w_\star)$, it follows by union bound that the conclusions of Lemma SM8.3 hold for all $w_k^{(s)}$, where $k \in \{0, \dots, m-1\}$, with probability at least

$$1 - \sum_{k=0}^{m-1} \frac{\delta}{(m+1)^2} = 1 - \frac{m}{(m+1)^2}\delta \ge 1 - \delta.$$

Consequently, applying Lemma SM8.3,

$$\|\Delta_m^{(s)}\|_{\nabla^2 L(w_\star)} \le \frac{3}{4}\|\Delta_{m-1}^{(s)}\|_{\nabla^2 L(w_\star)} + \frac{7}{48}\|\Delta_0^{(s)}\|_{\nabla^2 L(w_\star)}.$$

Now recursively applying the relation in the previous display, and using $m = 10 > \frac{\log(1/15)}{\log(3/4)}$, we reach

$$\begin{aligned}\|\Delta_m^{(s)}\|_{\nabla^2 L(w_\star)} &\leq \left(\frac{3}{4}\right)^m \|\Delta_0^{(s)}\|_{\nabla^2 L(w_\star)} + \left(\sum_{k=0}^{m-1}\left(\frac{3}{4}\right)^k\right)\frac{7}{48}\|\Delta_0^{(s)}\|_{\nabla^2 F(w_\star)} \\ &\leq \frac{1}{15}\|\Delta_0^{(s)}\|_{\nabla^2 L(w_\star)} + \frac{7}{48(1-\frac{3}{4})}\|\Delta_0^{(s)}\|_{\nabla^2 L(w_\star)} \\ &= \left(\frac{1}{15}+\frac{7}{12}\right)\|\Delta_0^{(s)}\|_{\nabla^2 L(w_\star)} \leq \frac{2}{3}\|\Delta_0^{(s)}\|_{\nabla^2 L(w_\star)}.\end{aligned}$$

Hence $\hat{w}^{(s+1)} = w_m^{(s)} \in \mathcal{N}_{\frac{2}{3}\varepsilon_0}(w_\star)$. ∎

We now have everything we need to prove Theorem 4.12.

SM8.5. Proof for Theorem 4.12. By Lemma SM8.4, we perform the recursion and obtain

$$\|\hat{w}^{(s)} - w_\star\|_{\nabla^2 L(w_\star)} \leq \left(\frac{2}{3}\right)^s \|\hat{w}^{(0)} - w_\star\|_{\nabla^2 L(w_\star)}.$$

Therefore, with $\varepsilon_0 \in (0, 1/6]$, if the number of stages satisfies

$$s \geq 3\log\left(\frac{\|\hat{w}^{(0)} - w_\star\|_{\nabla^2 L(w_\star)}}{\epsilon}\right),$$

then we achieve

$$\|\hat{w}^{(s)} - w_\star\|_{\nabla^2 L(w_\star)} \leq \epsilon.$$

Observing that each stage requires $n + 2mb_g$ component gradient evaluations, and that $\tau^\rho(\mathcal{N}_{\varepsilon_0}(w_\star)) \leq n$ (recall Lemma 3.3), we immediately conclude that the total number stochastic gradient evaluations is given by

$$\mathcal{O}\left(\left[n + \tilde{\mathcal{O}}\left(\tau^\rho(\mathcal{N}_{\varepsilon_0}(w_\star))\log\left(\frac{1}{\delta}\right)\right)\right]\log\left(\frac{1}{\epsilon}\right)\right) = \mathcal{O}\left(n\log\left(\frac{1}{\epsilon}\right)\right).$$

This completes the proof.

SM9. Proof of Corollary 4.13.

Proof. The hypotheses on the spectrum of $\frac{1}{n}X^TX$ and the assumption on the ridge-leverage coherence of $\nabla^2 L(w_\star)$, allow us to apply Lemma 7 and Proposition 15 of [SM3] to conclude that $\tau^\rho(\mathcal{N}_{\varepsilon_0}(w_\star)) = \mathcal{O}(\sqrt{n})$. The corollary now follows by invoking Theorem 4.12. ∎

SM10. Additional experimental details.

In this section, we provide additional details for the experiments performed in section 5.

SM10.1. Algorithmic hyperparameters.

In this subsection, we detail how the hyperparameter settings for the algorithms used in section 5.

SM10.1.1. Gradient batchsize and Coordinate blocksize. For the performance experiments, we used a gradient batchsize of $b_g = 256$ for datasets with $n_{\text{tr}} < 10^5$, and $b_g = 2048$ for datasets with $n_{\text{tr}} \geq 10^5$. For the showcase experiments, we use a gradient batchsize of $b_g = \lfloor 0.01 n_{\text{tr}} \rfloor$. For the block coordinate methods, we use a blocksize of $\lfloor 0.01 n_{\text{tr}} \rfloor$.

SM10.1.2. Learning rate. We set the learning rate for `SAGA` and `SVRG` according to the recommendations in [SM4, SM11]. Note, these papers set the learning rate based on the expected smoothness constant $\mathcal{L}$ [SM5], which accounts for minibatching, and enables the use of larger learning rate than the classical recommendations in [SM7, SM2], which assume $b_g = 1$. For `Catalyst`, we follow the recommendations in [SM8]. The learning rate for `MBSVRP` is set as $\eta = \min\{1/(4\mathcal{L}), 1\}$. This setting was found after considerable experimentation, as we found the recommended learning in [SM12] often lead to divergence. The learning rate for the block coordinate methods was set as the reciprocal of the block smoothness constant of the sampled block, as is standard practice in the literature [SM1, SM10].

SM10.1.3. Other hyperparameter settings. `Catalyst` and `MB-SVRP` have additional hyperparameters, for these we follow the recommendations in the original papers [SM8, SM12].

SM10.2. Datasets used in the experiments. Table SM1 presents the details for all the datasets used in the main paper. The condition number κ is computed as $\kappa(X^T X)$ if $n > p$ and $\kappa(XX^T)$ if $p > n$. The largest and smallest eigenvalue are estimated using `scipy`'s svds function with the solver set to LOBPCG.

SM10.2.1. Preprocessing details. The rows of all data matrices are scaled to have unit-norm to ameiliorate ill-conditioning from poorly scaled data. Note, the condition number estimate in Table SM1 is for the datasets after their rows have been scaled to have unit norm.

For the torchvision datasets, classification is not performed on the original datasets. Instead, we a perform a feature transformation by passing through the data matrices through the first 49 layers of a pre-trained ResNet50 model [SM6] available in torchvision.

SM11. Performance plots for medium strength regularization. In this section, we run the same performance experiment as in section 5, only with a larger value of the regularization: $\mu = 10^{-1}\|X^T b\|_\infty / n$. `SAPPHIRE` still yields the best performance, but its advantage has narrowed somewhat, as it is now comparable to `Catalyst` on the Lasso testbed, however it still maintains its advantage on the Logistic regression testbed. The improved performance of the first-order methods is unsurprising, as larger regularization leads to a better conditioned problem, which implies faster convergence of first-order methods.

Table SM1
Datasets Summary

Dataset	Task	n_{tr}	n_{tst}	p	κ	Non-zeros (%)	Source
a9a	Classification	32561	16281	122	5.45e+39	100	LIBSVM
abalone	Regression	3341	836	8	1.73e+03	100	LIBSVM
avazu	Classification	12642186	1719304	999975	1.10e+08	0.0001	LIBSVM
cadata	Regression	16512	4128	8	5.89e+05	100	LIBSVM
covtype	Classification	464809	116203	54	1.28e+05	100	LIBSVM
e2006	Regression	16087	3308	150358	3.81e+08	0.83	LIBSVM
epsilon	Classification	400000	100000	2000	3.21e+10	100	LIBSVM
gisette	Classification	6000	1000	5000	3.71e+06	100	LIBSVM
housing	Regression	404	102	13	5.95e+07	100	LIBSVM
ledgar	Classification	70000	10000	19986	8.62e+05	0.29	LIBSVM
mg	Regression	1108	277	6	1.02e+01	100	LIBSVM
mushrooms	Classification	6499	1625	112	4.76e+45	100	LIBSVM
phishing	Classification	8844	2211	68	2.08e+40	100	LIBSVM
rcv1	Classification	677399	20242	47236	2.53e+05	0.15	LIBSVM
realsim	Classification	57847	14462	20958	9.62e+04	0.25	LIBSVM
scotus	Classification	6400	1400	126397	2.95e+05	1.03	LIBSVM
space_ga	Regression	2485	622	6	5.14e+02	100	LIBSVM
url	Classification	1916904	479226	3231961	4.29e+07	0.0035	LIBSVM
w8a	Classification	39799	9950	300	5.31e+83	100	LIBSVM
yearmsd	Regression	463715	51630	90	6.60e+05	100	LIBSVM
ct_scan	Regression	42800	10700	384	2.15e+40	100	OpenML
dorothea	Classification	920	230	100000	4.08e+01	0.91	OpenML
imdb_drama	Classification	96735	24184	1001	4.26e+02	1.94	OpenML
ova_colon	Regression	1236	309	10935	5.37e+05	100	OpenML
ova_lung	Classification	1236	309	10935	5.56e+05	100	OpenML
ovarian	Regression	202	51	15154	9.94e+04	100	OpenML
prostate	Regression	81	21	12600	9.58e+03	100	OpenML
qsar_tid_11	Regression	4593	1149	1024	1.75e+04	6.34	OpenML
ujiindoorloc_latitude	Regression	16838	4210	525	4.49e+47	100	OpenML
yolanda	Regression	320000	80000	100	3.92e+06	100	OpenML
cifar_10	Classification	50000	10000	2048	1.04e+07	100	torchvision
fashion_mnist	Classification	60000	10000	2048	1.25e+13	100	torchvision
svhn	Regression	73257	26032	2048	5.32e+08	100	torchvision
uk_biobank	Regression	269704	67425	3511	3.84e+16	99.6	UK Biobank

REFERENCES

[1] A. Beck and L. Tetruashvili, *On the convergence of block coordinate descent type methods*, SIAM Journal on Optimization, 23 (2013), pp. 2037–2060.

[2] A. Defazio, F. Bach, and S. Lacoste-Julien, *SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives*, Advances in Neural Information Processing Systems, 27 (2014).

[3] Z. Frangella, P. Rathore, S. Zhao, and M. Udell, *PROMISE: Preconditioned stochastic optimization methods by incorporating scalable curvature estimates*, Journal of Machine Learning Research, 25 (2024), pp. 1–57, http://jmlr.org/papers/v25/23-1187.html.

[4] N. Gazagnadou, R. Gower, and J. Salmon, *Optimal mini-batch and step sizes for SAGA*, in International Conference on Machine Learning, PMLR, 2019, pp. 2142–2150.

[5] R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik, *SGD: General analysis and improved rates*, in International Conference on Machine Learning, PMLR, 2019,

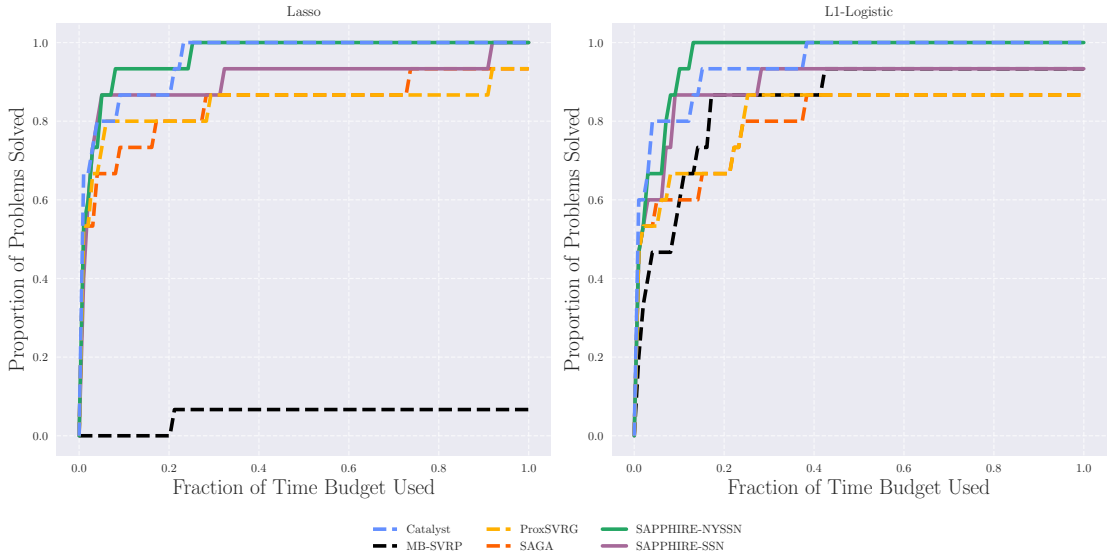

Figure SM2. *Performance plot with medium regularization. Even with a larger value of the regularization,* ***SAPPHIRE*** *still delivers the best performance, though the gap has narrowed compared to Figure 2, as the first-order competitors perform better with larger regularization.*

pp. 5200–5209.

[6] 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.

[7] R. Johnson and T. Zhang, *Accelerating stochastic gradient descent using predictive variance reduction*, Advances in Neural Information Processing Systems, 26 (2013).

[8] H. Lin, J. Mairal, and Z. Harchaoui, *Catalyst acceleration for first-order convex optimization: from theory to practice*, Journal of Machine Learning Research, 18 (2018), pp. 1–54.

[9] L. M. Nguyen, K. Scheinberg, and T. H. Tran, *Stochastic ista/fista adaptive step search algorithms for convex composite optimization*, Journal of Optimization Theory and Applications, 205 (2025), https://doi.org/10.1007/s10957-025-02621-8, https://doi.org/10.1007/s10957-025-02621-8.

[10] P. Richtárik and M. Takáč, *Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function*, Mathematical Programming, 144 (2014), pp. 1–38.

[11] O. Sebbouh, N. Gazagnadou, S. Jelassi, F. Bach, and R. Gower, *Towards closing the gap between the theory and practice of svrg*, Advances in neural information processing systems, 32 (2019).

[12] 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.

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