NysReg-Gradient: Regularized Nyström-Gradient for Large-Scale Unconstrained Optimization and Its Applications

Hardik Tankaria*, Makoto Yamada†, Dinesh Singh*
* School of Computing and Electrical Engineering, Indian Institute of Technology Mandi, India
hardiktankaria1406@gmail.com, dineshsingh@iitmandi.ac.in
† Machine Learning and Data Science Unit, Okinawa Institute of Science and Technology, Japan

***Abstract*—We develop a regularized Nyström method for solving large-scale unconstrained optimization problems that arise in high-dimensional machine learning. In such settings, classical second-order methods are often impractical due to prohibitive computational and memory costs. While quasi-Newton methods rely exclusively on first-order information and Newton-sketch approaches require large embedding matrices, the proposed method achieves scalability by efficiently exploiting partial curvature information. Specifically, we construct a regularized Nyström approximation of the Hessian using a small subset of columns, yielding an accurate and computationally efficient second-order approximation. The method is compatible with both deterministic and stochastic optimization frameworks, including gradient descent and stochastic gradient descent. To further reduce per-iteration complexity, the regularized-Nyström inverse–gradient product is computed directly without explicit matrix inversion. We provide theoretical convergence guarantees and validate the proposed approach through extensive experiments on benchmark datasets. The results demonstrate clear computational advantages over randomized subspace Newton and Newton-sketch methods in high-dimensional regimes. We further illustrate the practical effectiveness of the method on a brain tumor detection task, highlighting its potential for real-world learning applications.**

***Index Terms*—Nyström method, Large-scale Optimization, Big Data Analytics, High-dimensional Machine Learning, Computational Efficiency**

I. Introduction

HIGH-dimensional learning presents fundamental challenges for numerical optimization, particularly in modern machine learning applications where the number of features can be very large. Domains such as bioinformatics, text analysis, and medical imaging frequently involve tens of thousands of dimensions, which makes many classical optimization algorithms computationally inefficient or even impractical. Consequently, developing optimization methods that scale effectively with dimensionality while maintaining reliable convergence and solution quality remains a central problem in large-scale and high-dimensional learning.

In this work, we consider the large-scale unconstrained optimization problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}), \tag{1}$$

where f is twice continuously differentiable and convex, and the dimension d is potentially very large, as is typical in big data settings.

Second-order methods, most notably Newton's method, are attractive due to their fast local convergence and robustness to ill-conditioning. However, their practical applicability to high-dimensional problems is severely limited by the $O(d^3)$ computational cost and $O(d^2)$ memory requirements associated with forming and inverting the Hessian matrix.

Existing approaches address this challenge through various approximations. Quasi-Newton methods replace the Hessian matrix with its approximation through a secant equation, that improves the scalability. Newton-sketch methods employ random projections to reduce dimensionality, but often require large sketch sizes, undermining their computational benefits. Randomized subspace Newton methods further reduce complexity by projecting onto low-dimensional subspaces, though they may discard informative curvature directions in high-dimensional regimes.

To bridge this gap, we propose a regularized Nyström-based approach that strikes a favorable balance between computational efficiency and optimization effectiveness. By constructing a low-rank Hessian approximation via carefully selected column subsets, the proposed method preserves critical second-order structure while remaining scalable to high-dimensional problems.

A. Background and Contributions

First-order optimization methods, including stochastic gradient descent (SGD) [1], AdaGrad, stochastic variance-reduced gradient (SVRG) [2], Adam [3], and stochastic recursive gradient algorithms such as SARAH, are widely used for large-scale problems due to their low per-iteration cost of $O(nd)$, where n denotes the number of samples. Despite their scalability, these methods often suffer from slow convergence, sensitivity to hyperparameter choices, and degraded performance on ill-conditioned problems.

Newton's method avoids many of these drawbacks and requires minimal tuning for self-concordant objectives, such as ℓ_2-regularized logistic regression. However, its per-iteration complexity of $\Omega(nd^2 + d^{2.37})$ [4] makes it impractical for large-scale applications. To mitigate this, subsampled Newton methods and randomized sketching techniques have been proposed. Subsampled Newton methods compute Hessians on small data subsets and are effective when d is moderate,

but become inefficient in high-dimensional settings. Newton sketch methods [5] approximate the Hessian using random embeddings, yet typically require sketch sizes proportional to d, limiting their efficiency and low-rank appeal.

Several refinements have been introduced to improve sketch-based Newton methods. The Newton-LESS method [6] employs sparse embeddings to reduce computational cost while maintaining convergence guarantees. The randomized subspace Newton (RSN) method [7] approximates the Hessian by projecting it onto a randomly sampled subspace, though it may utilize only partial curvature information.

Nyström-based techniques offer an alternative by constructing low-rank approximations through column sampling. Talwalkar [8] proposed the Nyström logistic regression algorithm, leveraging the Nyström method to approximate the Hessian of the regularized logistic regression.

Quasi-Newton methods, including LBFGS [9] and its stochastic variants [10, 11, 12, 13], further reduce complexity by approximating the Hessian inverse using gradient differences or Hessian-vector products. While effective in many settings, these methods rely primarily on first-order information and may fail to capture rich curvature structure in high-dimensional problems.

Contributions: The main contributions of this work are as follows. We propose a deterministic and stochastic regularized Nyström Hessian approximation method for large-scale unconstrained optimization. The Nyström approximation is regularized using information derived from the gradient, and its inverse is efficiently computed via the Woodbury identity. We provide rigorous convergence analysis and theoretical guarantees. Extensive numerical experiments on benchmark datasets demonstrate the effectiveness of the proposed method compared with existing approaches. Finally, we illustrate its practical utility through a tumor classification task on brain MRI data.

II. Nyström approximation

Over the years, researchers have focused on low-rank approximations by selectively extracting key components of the original matrix. One such popular method in this context is the Nyström approximation [14], which is a low-rank approximation of a positive semidefinite matrix. Derezinski et al. [15] proposed improvements in the approximation guarantees of column subset selection and the Nyström method using spectral properties.

Definition 1 (Nyström approximation)**.** Let $\boldsymbol{H} \in \mathbb{R}^{d\times d}$ be a symmetric positive semi-definite matrix. Then, choose m columns of $\boldsymbol{H}$ randomly to form a $d \times m$ matrix $\boldsymbol{C}$. Let $m\times m$ be a matrix $\boldsymbol{M}$ such that it is formed by the intersection of those m columns and corresponding m rows of $\boldsymbol{H}$. $\boldsymbol{M}_k$ is the best k-rank approximation of $\boldsymbol{M}$. A k-rank Nyström approximation $\boldsymbol{N}_k$ of $\boldsymbol{H}$ can be defined as

$$\boldsymbol{N}_k = \boldsymbol{C}\boldsymbol{M}_k^{\dagger}\boldsymbol{C}^{\top}. \tag{2}$$

where $\boldsymbol{M}_k^{\dagger}$ is a pseudo-inverse of $\boldsymbol{M}_k$. Letting $\boldsymbol{H} = \nabla^2 f(\boldsymbol{w})$ to be a Hessian matrix of the objective function (1), following theorem shows the distance between the Hessian $\boldsymbol{H}$ and the Nyström approximation $\boldsymbol{N}$ of $\boldsymbol{H}$.

Next, we propose the regularized Nyström algorithm for the unconstrained optimization problem. Let $\boldsymbol{H} = \nabla^2 f(\boldsymbol{w})$ be a Hessian of the objective function, and we pick $\boldsymbol{\Omega} \subseteq \{1, \ldots, d\}$ indices uniformly at random such that $m = |\boldsymbol{\Omega}|$ and compute the Nyström approximation as

$$\boldsymbol{N}_k = \boldsymbol{C}\boldsymbol{M}_k^{\dagger}\boldsymbol{C}^{\top} = \boldsymbol{Z}\boldsymbol{Z}^{\top}, \tag{3}$$

where $\boldsymbol{C} \in \mathbb{R}^{d\times m}$ is a matrix consisting of m columns ($m \ll d$) of $\boldsymbol{H}$, M is $m \times m$ intersection matrix of m columns and m rows with same indices, and the rank of $\boldsymbol{M}$ is $k \leq m$. We first obtain the best k rank approximation using the singular value decomposition (SVD) of $\boldsymbol{M}_k$ as $\boldsymbol{M}_k = \boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{U}_k^{\top}$, $U_k \in \mathbb{R}^{m\times k}$ are singular vectors and $\boldsymbol{\Sigma}_k \in \mathbb{R}^{k\times k}$ consisting of k singular values. Then we compute the pseudo-inverse $\boldsymbol{M}_k^{\dagger} = U_k\boldsymbol{\Sigma}_k^{-1}U_k^{\top}$. Using $\boldsymbol{U}\boldsymbol{\Sigma}_t^{-1/2}$ from the pseudo-inverse of $\boldsymbol{M}$ and $\boldsymbol{C}$, we compute $\boldsymbol{Z} = \boldsymbol{C}\boldsymbol{U}_k\boldsymbol{\Sigma}_k^{-1/2} \in \mathbb{R}^{d\times k}$, and we get $\boldsymbol{N}_k = \boldsymbol{Z}\boldsymbol{Z}^{\top}$. Note that the number of columns m is a hyperparameter.

Example 1. ℓ_2-regularization: In this example, we present the elation between ℓ_2 regularization and fixed rank Nyström approximation. Consider ℓ_2 regularized objective function

$$\min_{\boldsymbol{w}\in\mathbb{R}^d}\left\{f(\boldsymbol{w}) := \sum_{i=1}^{n} f_i(\boldsymbol{w}) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2\right\}, \tag{4}$$

where each f_i are convex, twice continuously differentiable, and $\lambda \geq 0$, and hence f is strongly convex function. Then the Hessian of ℓ_2-regularized function can be given as $\boldsymbol{H} = \sum_{i=1}^{n}\nabla^2 f_i(\boldsymbol{w}) + \lambda\boldsymbol{I}$ and $\lambda_{\min}(\nabla^2 f(\boldsymbol{w})) \geq \lambda$. The formulation of column matrix $\boldsymbol{C} = \boldsymbol{H}\boldsymbol{S} = \left(\sum_{i=1}^{n}\nabla^2 f_i(\boldsymbol{w})\right)\boldsymbol{S} + \lambda\boldsymbol{S}^{\top}\boldsymbol{I}$ and matrix $\boldsymbol{M} = \boldsymbol{S}^{\top}\boldsymbol{H}\boldsymbol{S} = \boldsymbol{S}^{\top}\left(\sum_{i=1}^{n}\nabla^2 f_i(\boldsymbol{w})\right)\boldsymbol{S} + \lambda\boldsymbol{S}^{\top}\boldsymbol{S} \in \mathbb{R}^{m\times m}$. Since $m \geq 1$ and entries of $\boldsymbol{S}$ are sampled independently with $\text{rank}(\boldsymbol{S}) = m$, matrix M becomes positive definite and hence it becomes the fixed ranked Nyström approximation, which also helps in the convergence to get minimum eigenvalue of M^{-1}. Hence, we can obtain Nyström approximation using $\boldsymbol{S}$,

$$\boldsymbol{N} = (\boldsymbol{H}\boldsymbol{S})(\boldsymbol{S}^{\top}\boldsymbol{H}\boldsymbol{S})^{-1}(\boldsymbol{H}\boldsymbol{S})^{\top} = \boldsymbol{C}\boldsymbol{M}^{-1}\boldsymbol{C}^{\top}$$

for fixed rank Nyström approximation.

III. Methodology

The second-order optimization methods often utilize the regularized approximated Hessian. Regularized parameters can be obtained through approaches such as the trust-region method or by adaptively determining the regularization parameter based on the gradient information [16, 17].

To ensure the non-singularity and obtain a descent direction, we compute a regularized Nyström approximation. An iterate of the regularized Nyström approximation is given by

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t(\boldsymbol{N}_t + \rho_t\mathbf{I})^{-1}\nabla f(\boldsymbol{w}_t). \tag{5}$$

We call our novel method "NysReg-gradient: Regularized Nyström gradient method (NGD)". The regularized parameter

$\rho_t > 0$ is determined based on the gradient information. Specifically, we set $\rho_t = c_1\|\nabla f(\boldsymbol{w}_t)\|^\gamma$ as similar to [16], where $c_1 > 0$. We consider ρ_t to be either $c_1\sqrt{\|\nabla f(\boldsymbol{w}_t)\|}$ for $\gamma = 1/2$, $c_1\|\nabla f(\boldsymbol{w}_t)\|$ for $\gamma = 1$, or $c_1\|\nabla f(\boldsymbol{w}_t)\|^2$ for $\gamma = 2$ as shown in Table I. We denote $\nabla f(\boldsymbol{w}_t) = \boldsymbol{g}_t$ for the rest of paper.

TABLE I: Relation between proposed methods and value of γ

Method	Value of γ	Regularizer ρ_t	Regularized Nyström
NGD	$\gamma = 1/2$	$\rho_t = c_1\|\boldsymbol{g}_t\|^{1/2}$	$\boldsymbol{N}_t + c_1\|\boldsymbol{g}_t\|^{1/2}$
NGD1	$\gamma = 1$	$\rho_t = c_1\|\boldsymbol{g}_t\|^{1}$	$\boldsymbol{N}_t + c_1\|\boldsymbol{g}_t\|^{1}$
NGD2	$\gamma = 2$	$\rho_t = c_1\|\boldsymbol{g}_t\|^{2}$	$\boldsymbol{N}_t + c_1\|\boldsymbol{g}_t\|^{2}$

To efficiently compute the inverse of $(\boldsymbol{N}_t+\rho_t)$ given in (5), we use the Sherman–Morrison–Woodbury identity as

$$\boldsymbol{p}_t = (\boldsymbol{N}_t+\rho_t\boldsymbol{I})^{-1}\boldsymbol{g}_t = \frac{1}{\rho_t}\boldsymbol{g}_t - \boldsymbol{Q}_t\boldsymbol{Z}_t^\top\boldsymbol{g}_t, \tag{6}$$

where $\boldsymbol{p}_t$ is search direction at tth iteration, $\boldsymbol{N}_t$ is Nyström approximation computed at $\boldsymbol{w}_t$, $\boldsymbol{g}_t$ is a gradient computed at $\boldsymbol{w}_t$ and $\boldsymbol{Q}_t = \frac{1}{\rho_t^2}\boldsymbol{Z}_t(\boldsymbol{I}_k + \frac{1}{\rho_t}\boldsymbol{Z}_t^\top\boldsymbol{Z}_t)^{-1}$. Here, $(\boldsymbol{I}_k + \frac{1}{\rho_t}\boldsymbol{Z}_t\boldsymbol{Z}_t^\top) \in \mathbb{R}^{m\times m}$, and its inverse can be computed much more quickly than the inverse of $(\boldsymbol{N}_t + \rho_t\boldsymbol{I})$ directly. We use the backtracking line search with Armijo's line search rule that finds a step size $\eta_t = \alpha^{(\ell)} = \tau\alpha^{(\ell-1)}$, starting from $\ell = 0$, the initial step size $\eta_0 = \alpha^{(0)}$, and finds the least positive integer $\ell \geq 0$ and increased ℓ by $\ell + 1$ until the

$$f(\boldsymbol{w}_t + \alpha^{(\ell)}\boldsymbol{p}_t) \leq f(\boldsymbol{w}_t) + \alpha^{(\ell)}\beta\boldsymbol{g}_t^\top\boldsymbol{p}_t, \tag{7}$$

holds, where $\alpha, \beta \in (0, 1)$. Next, we introduce the main algorithm.

Algorithm 1 NysReg-gradient: Regularized Nyström-Gradient Algorithm

1: **Initialize** Initial parameters $\boldsymbol{w}_0$, desired rank $|\boldsymbol{\Omega}| = m$, $\alpha, \beta \in (0, 1)$, and maximum iterations $t_{\max}$
2: $t \leftarrow 0$
3: **repeat**
4: $\quad\boldsymbol{g}_t = \nabla f(\boldsymbol{w}_t)$
5: $\quad$randomly pick indices set $\boldsymbol{\Omega} \subseteq \{1, 2, \ldots, d\}$ such that $m = |\boldsymbol{\Omega}|$
6: $\quad$compute $\boldsymbol{C}_t$ ($\boldsymbol{\Omega}$ columns of the Hessian)
7: $\quad$compute $\boldsymbol{Z}_t$ using (3) and compute ρ_t
8: $\quad\boldsymbol{Q}_t = \frac{1}{\rho_t^2}\boldsymbol{Z}_t(\boldsymbol{I}_k + \frac{1}{\rho_t}\boldsymbol{Z}_t^T\boldsymbol{Z}_t)^{-1}$
9: $\quad$Compute $\boldsymbol{p}_t = (\boldsymbol{N}_t + \rho_t\boldsymbol{I})^{-1}\boldsymbol{g}_t$ using (6)
10: $\quad$Use backtracking line search with Armijo's rule to find η_t using (7)
11: $\quad\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t\boldsymbol{p}_t$
12: $\quad t = t + 1$
13: **until** $t = t_{\max}$ or some termination criteria is satisfied
14: **return** $\boldsymbol{w}_t$

A. Computational Complexity

Consider an ℓ_2-regularized logistic regression objective function as given in equation (16). The proposed approach incurs a cost of $O(ndm)$ to compute C_t. The cost of computing the SVD of $\boldsymbol{M}_t = \boldsymbol{U}_t\boldsymbol{\Sigma}_t\boldsymbol{U}_t^\top$ is $O(m^3)$. Since $\boldsymbol{\Sigma}_t$ is diagonal, the pseudo-inverse can be computed simply by inverting diagonal elements, which is negligible as $m \ll d$ and $O(dmk + m^3)$ for the computation of $\boldsymbol{Z}_t$. Lastly, the cost of $\boldsymbol{Q}_t$ is $O(dm^2 + m^3)$ for the Sherman-Morrison update $(\boldsymbol{Z}_t\boldsymbol{Z}_t^\top + \rho_t I)^{-1}\boldsymbol{g}_t$. Thus, the overall computational cost per iteration stands at $O(ndm + dmk + dm^2 + m^3)$, which is directly proportional to that of the first-order method and significantly lower than $O(d^3 + nd^2)$ that of Newton's method and $O(ndm + d^3)$ that of Newton-sketch.

B. Stochastic Variant

In this subsection, we discuss the stochastic variant of the NGD, we compute the mini-batch stochastic gradient at every iteration. We compute the regularized Nyström $\boldsymbol{N}_\tau + \rho_\tau\boldsymbol{I}$, once per epoch with regularized parameter[1] $\rho_\tau = c_1$ and kept fix throughout the run. It is important to note that fix regularized parameter employed to maintain the stability of the iterates as using stochastic gradient $\rho_\tau = c_1\|\nabla f_i\|^\gamma$ cause instability and oscillations. The search direction is given by $(\boldsymbol{N}_\tau + \rho_\tau\boldsymbol{I})^{-1}\boldsymbol{g}_{t-1,\tau}$. We call this variant NSGD. Furthermore, we employ a diminishing step size η_t for the stochastic variant NSGD.

IV. Convergence analysis

In this section, we provide analysis that offers insights into the overall convergence behavior of the algorithm. It is important to note that our convergence analysis is based on the objective function defined in equation (4).

Assumption 1. *i)* The objective function (4) is twice continuously differentiable and f is L_g-smooth, *i.e.,*

$$\|\nabla^2 f(\boldsymbol{w}_t)\| \leq L_g, \quad \forall\ \boldsymbol{w}_t \in \mathbb{R}^d. \tag{8}$$

ii) The objective function (4) is strongly convex. *iii)* The Hessian of the objective function is $L_{\boldsymbol{H}}$-Lipschitz continuous.

Assumption 2. For all $\boldsymbol{x}, \boldsymbol{y}$, the gradient is Lipschitz, *i.e.,*

$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq L_g\|\boldsymbol{x} - \boldsymbol{y}\|.$$

Lemma 1. *Under Assumptions 1-2, with step size η_t obtained by Armijo line search and $m \geq 4/\varepsilon^2$ sampled columns, then expected error satisfies:*

$$\mathbb{E}[\|\boldsymbol{w}_{t+1} - \boldsymbol{w}^*\|] \leq \sigma_t\|\boldsymbol{w}_t - \boldsymbol{w}^*\| + \zeta_t\|\boldsymbol{w}_t - \boldsymbol{w}^*\|^2$$

where

$$\sigma_t = \left(1 - \frac{\eta_t\lambda}{\lambda + \rho_t} + \frac{\eta_t L\mathcal{E}}{\rho_t(\lambda + \rho_t)}\right) \text{ and, } \zeta_t = \frac{\eta_t L_H}{2(\lambda + \rho_t)}.$$

[1]Note that the regularization parameter ρ_τ is a fixed scalar selected via grid search; it is neither derived from full-batch gradients nor from stochastic gradients.

Proof. Let the NGD iterates given as follows:

$$\begin{aligned}
&\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t(\boldsymbol{N}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{g}_t \\
&\boldsymbol{w}_{t+1} - \boldsymbol{w}_* = \boldsymbol{w}_t - \boldsymbol{w}_* - \eta_t(\boldsymbol{N}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{g}_t \\
&\qquad + \eta_t(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{g}_t - \eta_t(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{g}_t \\
&= \underbrace{\boldsymbol{w}_t - \boldsymbol{w}_* - \eta_t(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{g}_t}_{\textbf{Part I}} \\
&+ \underbrace{\eta_t[(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1} - (\boldsymbol{N}_t + \rho_t \boldsymbol{I})^{-1}]\boldsymbol{g}_t}_{\textbf{Part II}} \qquad (9)
\end{aligned}$$

Using Taylor's expansion and Lipschitz continuity of Hessian, we get $\boldsymbol{g}_t = \boldsymbol{H}_t(\boldsymbol{w}_t - \boldsymbol{w}_*) + \boldsymbol{r}_t$ (where $r_t = \int_0^1 \left(\nabla^2 f(\boldsymbol{w}^* + t(\boldsymbol{w}_t - \boldsymbol{w}_*) - \nabla^2 f(\boldsymbol{w}_t)\right)(\boldsymbol{w}_t - \boldsymbol{w}_*)\, dt = O(\|\boldsymbol{w}_t - \boldsymbol{w}_*\|^2)$ as $\|\boldsymbol{r}_t\| \le L_{\boldsymbol{H}}\|\boldsymbol{w}_t - \boldsymbol{w}_*\|^2/2$) and substituting in **Part I** and we obtain,

$$\begin{aligned}
&\boldsymbol{w}_t - \boldsymbol{w}_* - \eta_t(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{g}_t \\
&= \boldsymbol{w}_t - \boldsymbol{w}_* - \eta_t(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}(\boldsymbol{H}_t(\boldsymbol{w}_t - \boldsymbol{w}_*) + r_t) \\
&= \boldsymbol{w}_t - \boldsymbol{w}_*(\boldsymbol{I} - \eta_t(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{H}_t) - \eta_t(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{r}_t
\end{aligned}$$

Taking norm, we get

$$\begin{aligned}
&\|\boldsymbol{I} - \eta_t(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\boldsymbol{H}_t)\|\|\boldsymbol{w}_t - \boldsymbol{w}_*\| \\
&\qquad \le \left(1 - \frac{\eta_t \lambda}{\lambda + \rho_t}\right)\|\boldsymbol{w}_t - \boldsymbol{w}_*\|. \qquad (10)
\end{aligned}$$

The second inequality comes from spectral contraction as $h(x) = 1 - \frac{\eta_t x}{x+a}$ is strictly decreasing for $x > 0$ and attains maximum at $x = \lambda_{\min}(\boldsymbol{H}) = \lambda$ along with $\eta_t \le 1$.

$$\eta_t\|(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\|\|\boldsymbol{r}_t\| \le \frac{\eta_t L_{\boldsymbol{H}}}{2(\lambda + \rho_t)}\|\boldsymbol{w}_t - \boldsymbol{w}_*\|^2, \qquad (11)$$

Note that, f is convex with λ being a ℓ_2-regularizer, making it λ-strongly convex, and hence $\boldsymbol{H}_t$ positive definite with $\lambda_{\min}(\boldsymbol{H}_t) = \lambda$. Next, taking norm on **Part II**, we obtain

$$\begin{aligned}
&\eta_t\|(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1} - (\boldsymbol{N}_t + \rho_t \boldsymbol{I})^{-1}\|\|\boldsymbol{g}_t\| \\
&\qquad \le \frac{\eta_t L\|\boldsymbol{H}_t - \boldsymbol{N}_t\|}{\rho_t(\lambda + \rho_t)}\|\boldsymbol{w}_t - \boldsymbol{w}_*\|,
\end{aligned}$$

which directly follows from upper bounds of $\|(\boldsymbol{H}_t + \rho_t \boldsymbol{I})^{-1}\|$, $\|(\boldsymbol{N}_t + \rho_t \boldsymbol{I})^{-1}\|$ and Lipschitz continuity of the gradient. Next, we take expectation of Part I and Part II inequalities, and using $\mathbb{E}\|\boldsymbol{H}_t - \boldsymbol{N}_t\| \le \lambda_{k+1}(\boldsymbol{H}_t) + \epsilon\sum_{i=1}^d \boldsymbol{H}_{ii,t}^2 =: \mathcal{E}$ from [14][Theorem 3], we obtain

$$\frac{\eta_t L\mathbb{E}\|\boldsymbol{H}_t - \boldsymbol{N}_t\|}{\rho_t(\lambda + \rho_t)}\|\boldsymbol{w}_t - \boldsymbol{w}_*\| \le \frac{\eta_t L\mathcal{E}}{\rho_t(\lambda + \rho_t)}\|\boldsymbol{w}_t - \boldsymbol{w}_*\|.$$

Finally, summing up the inequalities of **Part I** and **Part II**, we obtain the desired upper-bound.

□

Lemma 2. *Under Assumptions 1-2, let $\{\boldsymbol{w}_t\}$ be the iterates generated by NGD1. If the regularized parameter is chosen as $\rho_t = c_1\|\boldsymbol{g}_t\|$, with some $c_1 > 0$ in NGD1, then*

$$c_1\lambda\|\boldsymbol{w}_t - \boldsymbol{w}_*\| \le \rho_t \le c_1 L\|\boldsymbol{w}_t - \boldsymbol{w}_*\|. \qquad (12)$$

Since the regularized parameter of NGD1 has $\gamma = 1$ in good agreement with the strongly convex and Lipschitz gradient inequality, we prove the convergence of NGD1[2].

Theorem 1 (Local linear convergence to a neighborhood)**.** *Under Assumptions 1–2, define*

$$R := \min\left\{\frac{\lambda}{c_1 L}, \frac{\eta_{\min}\lambda}{4L_H}\right\}, \qquad \delta_{\mathcal{E}} := \frac{8L\mathcal{E}}{\eta_{\min}c_1\lambda^2}. \qquad (13)$$

Assume that the Nyström approximation error satisfies $\delta_{\mathcal{E}} \le R$. Then, for any iterate $\boldsymbol{w}_t$ of NGD1 such that

$$\delta_{\mathcal{E}} \le \|\boldsymbol{w}_t - \boldsymbol{w}_*\| \le R, \qquad (14)$$

the iterates satisfy

$$\mathbb{E}\|\boldsymbol{w}_{t+1} - \boldsymbol{w}_*\| \le \left(1 - \frac{\eta_{\min}}{4}\right)\|\boldsymbol{w}_t - \boldsymbol{w}_*\|. \qquad (15)$$

V. Numerical experiments

We begin by describing the experiment setup for numerical experiments. We performed Figure experiments on MATLAB R2018a on Intel(R) Xeon(R) CPU E7-8890 v4 @ 2.20GHz with 96 cores and Figure on MATLAB R2019a on Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz with 32 cores. We implemented the existing and proposed methods in MATLAB using the SGDLibrary [18]. We solve on standard learning problems, that is, the ℓ_2-logistic regression:

$$\min_w F(w) = \frac{1}{n}\sum_{i=1}^n \log\left[1 + \exp(-b_i a_i^T w)\right] + \frac{\lambda}{2}\|w\|^2, \qquad (16)$$

where $a_i \in \mathbb{R}^d$ is feature vector and $b_i \in \{\pm 1\}$ is target label of the i-th sample, and λ is a ℓ_2 regularizer. We evaluated the numerical experiments on benchmark datasets given in Table II. The datasets are binary classification problems and all datasets are available on LIBSVM [19]. We demonstrate the performance of the proposed and existing methods on the ℓ_2-regularized logistic regression problem. We optimize the constant c_1 in regularizer $\rho_t = c_1\|\boldsymbol{g}_t\|^\gamma$ using a grid search $c_1 \in \{10^0, 10^{-1}, 10^{-2}, 10^{-3}\}$. For each method, the best-performing model was selected based on the minimum cost error on the training.

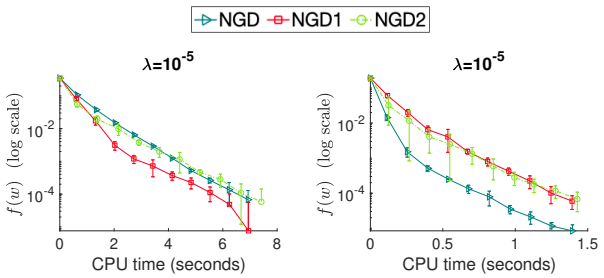
TABLE II: Details of the datasets used in the experiments

Dataset	Dim	Train	Test	Density
adult[3]	$123 + 1$	32,561	16,281	0.1128
gisette[3]	$5,000 + 1$	6,000	1,000	0.9910
epsilon[3]	$2,000 + 1$	50,000	50,000	1
real-sim[3]	$20,958 + 1$	57,909	14,400	0.0024
w8a[3]	$300 + 1$	49,749	14,951	0.0388

First, we study the performance of NGD, NGD1 and NGD2 to see the behavior of different $\rho = c_1\|\boldsymbol{g}_t\|^\gamma$, where $\gamma = 1/2$ for NGD, $\gamma = 1$ for NGD1 and $\gamma = 2$ for NGD2. We computed the ℓ_2-regularized *logistic regression* with $\lambda = 10^{-5}$ on the *ijcnn1* and *adult* datasets. To avoid numerical instability when $\|\boldsymbol{g}_t\|$ becomes very small, we use a safeguarded regularization parameter of the form $\rho_t = c_1\max\{c_0, |\boldsymbol{g}_t|^\gamma\}$, where $c_0 > 0$ is a very small constant.

[2]One can easily obtain the proof of main Theorem for NGD and NGD2 by subtituting $\gamma = 1/2$ and $\gamma = 2$ in Lemma 2.

[3]Available at LIBSVM [19] https://www.csie.ntu.edu.tw/cjlin/libsvm/

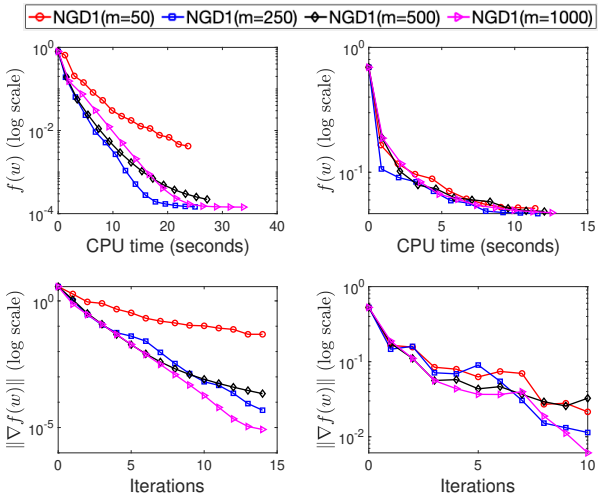


(a) *adult* (b) *ijcnn1*

Fig. 1: Left figure shows the experiments on *adult* for $m = 25$ and right figure shows the experiments on *ijcnn1* for $m = 5$.

Figure 1 shows that NGD1 outperforms NGD and NGD2 for the *adult* dataset and NGD outperforms NGD1 and NGD2 for the *ijcnn1* dataset. Therefore, in the next subsection, we consider NGD and NGD1 to compare the behavior with varying numbers of selected columns.

First, we demonstrate the comparison of various sketch sizes (no. of selected columns) for high-density data *gisette* and sparse data *w8a* on *logistic regression* with $\lambda = 10^{-5}$ to observe the robustness of the proposed methods to eshtablish the efficiency of the proposed method. We keep the same c_1 in ρ_t for each dataset to compare the different numbers of selected columns. Figure 2(a) shows the numerical performance for the *gisette* dataset, due to the high density only $m = 250(5\%)$ columns are sufficient to obtain the minimum value of the objective function within the comparative CPU time along with norm of gradient.



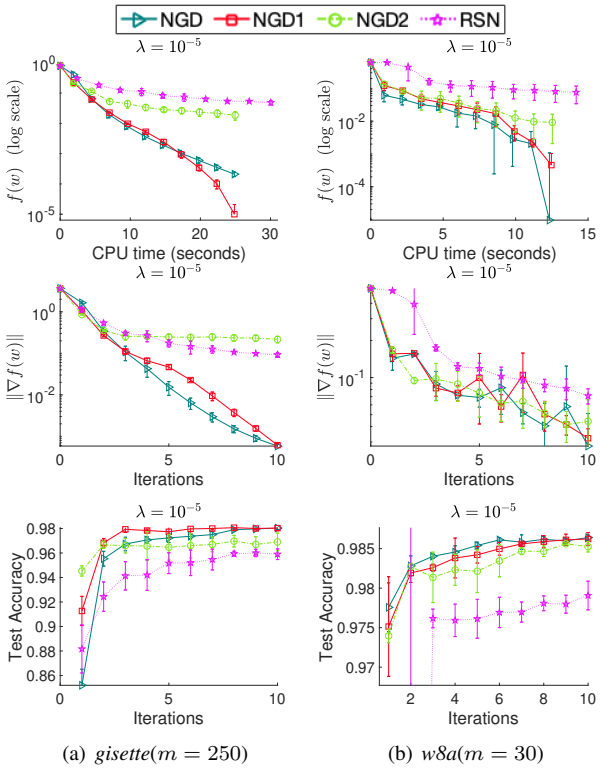
(a) *gisette* (b) *w8a*

Fig. 2: Column comparison: Left column(a) shows experiments on *gisette* dataset and right column(b) shows experiments on *w8a*

Similarly, for the *w8a* dataset we computed NGD and Figure 2(b) obtain the similar minimum value of the objective function in the comparative CPU time and norm of the gradient.

We next compare the NGDs with the randomized subspace Newton (RSN) [7]. To have a fair comparison of the subspace Newton, we compute the RSN with the Armijo's rule with backtracking line search (instead of $1/L$) and compute RSN exactly as given in [7, definition 4] for generalized linear models. We compute the *logistic regression* with $\lambda = 10^{-5}$ and compare NGDs and RSN in Figure 3.

For the *gisette* data, In Figure 3, NGD1 outperforms all methods in terms of achieving minimum cost and in reducing the norm of gradient along with test accuracy. However, NGD2 and RSN performs similar in all expects. As for the *w8a* dataset, Figure 3, NGD outperforms all methods in terms of achieving minimum cost. However, RSN and NGD2 works similar in reducing cost of objective function and RSN reduces the norm of gradient as similar to other NGDs.



(a) *gisette*($m = 250$) (b) *w8a*($m = 30$)

Fig. 3: Comparison with RSN for *gisette* dataset with $m = 250$

Subsequently, we compare the NGDs with the Newton sketch(NS) [5]. We compare the raw Nyström with the Hessian approximation of NS in terms of closeness with the Hessian. NS computes the Hessian approximation as $(\nabla^2 f(\boldsymbol{w})^{1/2})^\top \boldsymbol{P}^\top \boldsymbol{P}(\nabla^2 f(\boldsymbol{w})^{1/2})$, and it is important to note that the $\boldsymbol{P} \in \mathbb{R}^{m\times n}$, where n is the number of samples and m is the sketch size. In Figure 4 we conduct numerical experiments on *w8a, realsim* and *gisette* datasets. We provide the comparison of norm difference with Hessian and its CPU time as m increases. We conduct these numerical experiments on the *logistic regression*. It is important to note that we use the *logistic regression* for *gisette* without ℓ_2 regularization. For the *w8a* dataset, we keep the ℓ_2-regularized *logistic regression* with $\lambda = 10^{-5}$. Hence the rank of $\boldsymbol{H}$ for *w8a* data is full.

As shown in Figure 4 (a) and (b), Nyström approximation

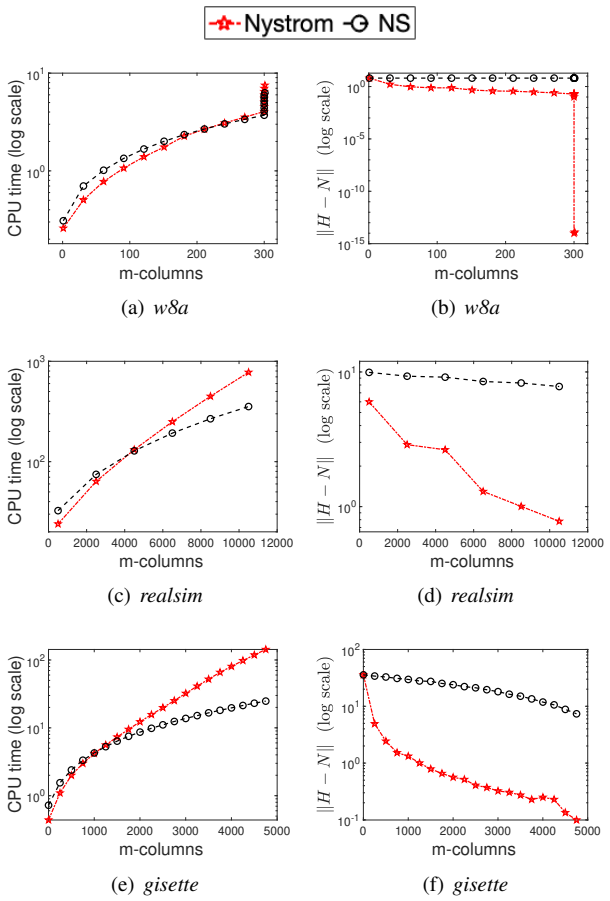


Fig. 4: Comparison between Nyström and Newton sketch where red shows difference between Hessian and Nyström and black shows difference between Hessian and Newton-sketch

outperforms the Newton sketch as m increases with the less CPU time for *w8a* in lesser CPU time compared to NS. Whereas in Figure 4 (e) and (f), the distance norm between Hessian and Nyström decreases significantly as m increases and takes more CPU time after $m \approx 1000$ compared to NS. However, we do not need to compute Nyström for large number of m, as we have seen in Figure 2, only 5% to 15% of d can give sufficient decrease in the objective function.

In the next step, we compared the proposed methods NGD, NGD1, and NGD2 with the existing classical first-order gradient descent and the *state of the art* second-order Hessian approximation method L-BFGS method [9]. The memory used in the L-BFGS method was set to 20.We show the norm of the gradient with respect to iterations. igure 5 shows the performance of experiments on *logistic regression* with $\lambda = 10^{-5}$. As shown in Figure 5(a), NGD1 outperforms all other methods in terms of both CPU time and iterations in terms of both training cost and the norm of the gradient. L-BFGS takes more CPU time compared to all variants of NGDs till the cost of 10^{-5}. Also, GD shows improvements after the 20th iteration and outperforms in terms of the test error, and NGD2 shows some increment in the test accuracy after the 30th iteration. Figure 5(b) shows that the NGDs are performing almost similarly and outperform the L-BFGS and GD in terms of the training cost, testing error, and test accuracy. NGD and NG1 outperform all of the methods in terms of the norm of the gradient.

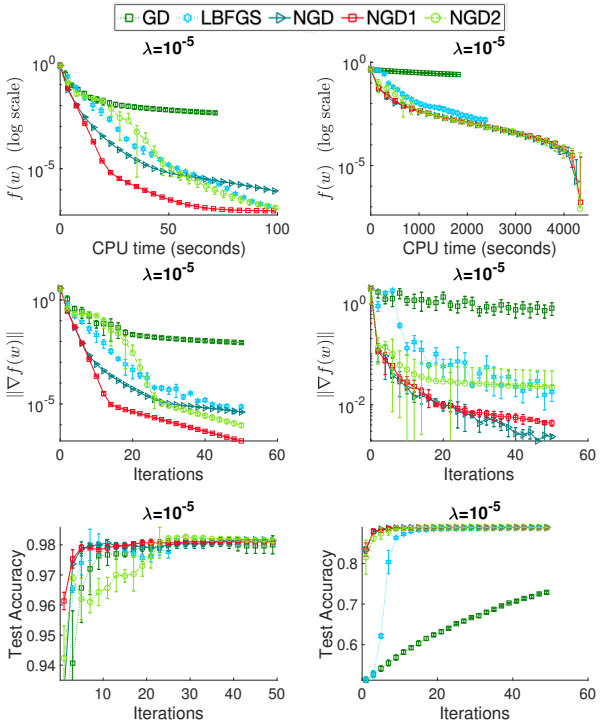


Fig. 5: Experiments on the deterministic methods: Left column (a) shows experiments on *gisette* dataset with $m = 500$ and right column(b) shows experiment on *epsilon* with $m = 200$.

Following this, we evaluate the performance of NSGD on the well-known deep models on the Imagenet dataset.

Experimental Setup: We compared our method with the first-order methods SGD and the well-known approximate second-order method KFAC [20] on ResNet152 [21] and EfficientNet models.

For NSGD, we used $\rho = 0.1$ and fixed the $m = \log_2 |\boldsymbol{w}|$, where $|\boldsymbol{w}|$ is the number of parameters in the respective model. We used a batch size of 128. We used a random sample of size of $\min\{6400, n \times 0.01\}$ to compute the partial Hessian $\boldsymbol{C}$ for Nyström SGD. The update frequency used to re-estimate the preconditioner in KFAC and its variants is set to 200, as used in their experiments. The ImageNet results were computed on a Quadro RTX 8000 GPU.

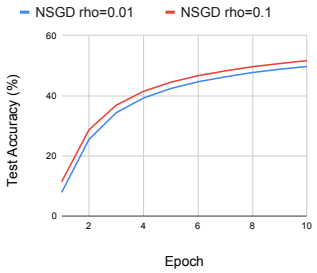


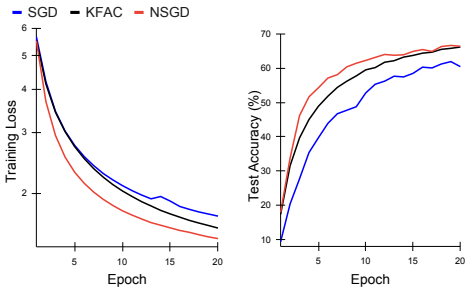
Fig. 6: Effect of the ρ on the test accuracy for ResNet18 on imagenet dataset.

Figure 7 presents the results of the ResNet152 and Ef-

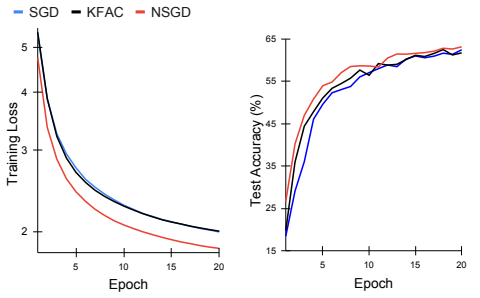
ficientNet on the ImageNet dataset. The proposed method outperformed both the SGD and KFAC for both the models in terms of training loss as well as test accuracy, showing the better optimization and generalization ability of the trained models. Table III shows the computational time comparison of methods. The per update computational time of the Nyström SGD on ResNet152 is 1.703 seconds which is slightly slower than SGD and KFAC. To further speed up the Nyström SGD is an interesting future work. Figure 6 shows the effect of the ρ parameter for ResNet18. As can be seen, the ρ parameter affects the model performance. We found setting $\rho = 0.1$ performs well in practice.

TABLE III: Per iteration computational time (seconds). [*For KFAC on EfficientNet with batch size 128 could not fit into memory]

Model	Method	Batch	Update Time	Hessian
ResNet152	SGD	128	1.006	-
	KFAC	128	1.064	-
	NSGD	128	1.060	0.643
EfficientNet	SGD	128	0.341	-
	KFAC	64*	1.173	-
	NSGD	128	0.347	2.620



(a) ResNet152 (Train loss) (b) ResNet152 (Test Acc.)



(c) EfficientNet(Train loss) (d) EfficientNet(Test Acc.)

Fig. 7: Results on Imagenet using ResNet152 and EfficientNet, respectively.

The codes of the proposed method is available at: https://github.com/hardy-opt/Nys_Reg.

VI. Application: Tumor Detection

Brain MRI is a standard imaging modality for the diagnosis of brain diseases, including tumor detection. Due to the complexity of this task, deep neural networks have become increasingly popular. These models are typically trained using first-order optimization methods. However, when the number of training samples is limited, such methods often struggle to produce stable and well-generalized models in the presence of a large number of parameters. In this work, we study brain MRI images for tumor detection using a dataset of 253 images, consisting of 155 tumor cases and 98 healthy cases, with a feature dimension of 1000.

We adopt a transfer learning framework, which is well suited for brain MRI and biological applications with limited data. In deep architectures, lower layers capture generic representations, while higher layers are task specific. As a result, fine-tuning is commonly restricted to the top layers. The learning problem is formulated as $\min f(w)$ where f is ℓ_2-regularized logistic regression given in (16). To fine-tune the top layers of the pre-trained network, we propose an NGD algorithm that constructs a partial Hessian of size $d \times m$ using uniformly sampled parameters with $m \ll d$, and applies the Nyström method to approximate the full Hessian.

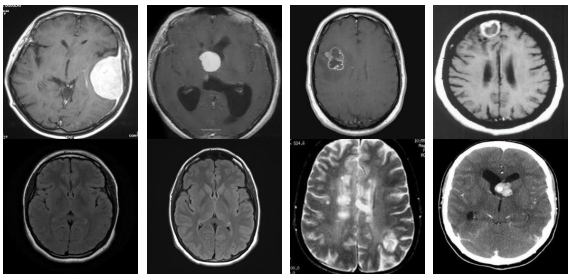


Fig. 8: Sample Images from MRI dataset [22], Top row: Tumor, Bottom row: Healthy

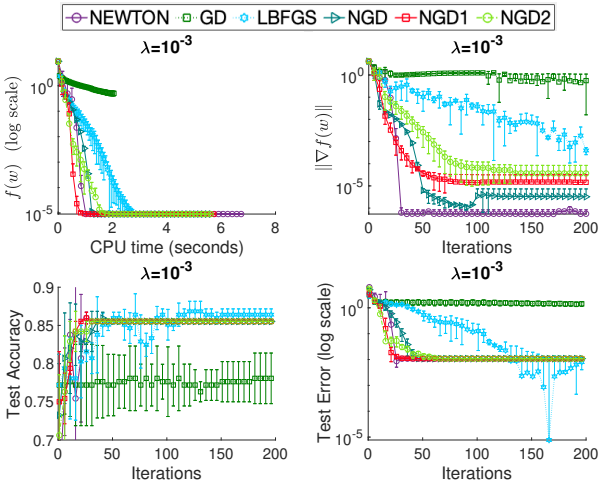


Fig. 9: Comparison of NGDs with existing methods on MRI dataset ($m = 200$)

Figure 9 shows that NGD1 outperforms other methods in terms of training cost in the least CPU time. Newton's method outperforms in terms of decreasing the norm of the gradient. Additionally, all NGDs are giving competitive behavior to each other in terms of the norm of gradient. GD and L-BFGS are not able to give competitive results in terms of test accuracy and test error. Also, all NGDs and Newton's methods have the

upper hand in achieving better test accuracy and test error.

VII. Conclusion

In this paper, we introduce a regularized Nyström approach for Hessian approximation and develop both deterministic and stochastic optimization methods for solving the resulting objective. We provide a comprehensive convergence analysis, including theoretical results based on the distance between the exact Hessian and its Nyström approximation. Extensive numerical experiments are conducted to evaluate the proposed methods against randomized subspace Newton, Newton sketch, and several first-order and quasi-Newton algorithms. The results demonstrate that the proposed approach is robust and achieves accurate Hessian approximation using only $\tilde{c}\log(d)$, where $\tilde{c} \in \{1, 2, 3, 4\}$ corresponding to approximately 5–15% of the dimension for the benchmark datasets considered in our experiments. We further extend our evaluation to a deep learning application on brain tumor detection showing strong practical performance, highlighting its potential for real-world learning tasks.

Acknowledgment

This work was supported by the Japan Society for the Promotion of Science (JSPS) through KAKENHI Grant Number 21H04874. This work was performed when the authors were affiliated with Kyoto University, Japan, and RIKEN, Japan. The authors thank Prof. Nobuo Yamashita for valuable discussions.

References

[1] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, sep 1951.

[2] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," *Advances in neural information processing systems*, vol. 26, pp. 315–323, 2013.

[3] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations, ICLR*, Y. Bengio and Y. LeCun, Eds., 2015.

[4] N. Agarwal, B. Bullins, and E. Hazan, "Second-order stochastic optimization for machine learning in linear time," *Journal of Machine Learning Research, JMLR*, vol. 18, pp. 116:1–116:40, 2017.

[5] M. Pilanci and M. J. Wainwright, "Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence," *SIAM Journal on Optimization*, vol. 27, no. 1, pp. 205–245, 2017.

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

[7] R. Gower, D. Kovalev, F. Lieder, and P. Richtárik, "RSN: Randomized subspace Newton," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[8] A. Talwalkar, "Matrix approximation for large-scale learning," Ph.D. dissertation, New York University, 2010.

[9] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1, pp. 503–528, 1989.

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

[11] R. Kolte, M. Erdogdu, and A. Ozgur, "Accelerating svrg via second-order information," in *NIPS workshop on optimization for machine learning*, 2015.

[12] R. H. Byrd, S. L. Hansen, J. Nocedal, and Y. Singer, "A stochastic quasi-Newton method for large-scale optimization," *SIAM Journal on Optimization*, vol. 26, no. 2, pp. 1008–1031, 2016.

[13] P. Moritz, R. Nishihara, and M. Jordan, "A linearly-convergent stochastic L-BFGS algorithm," in *Proceedings of the International Conference on Artificial Intelligence and Statistics, AISTATS*, 2016, pp. 249–258.

[14] P. Drineas and M. W. Mahoney, "On the Nyström method for approximating a gram matrix for improved kernel-based learning," *Journal of Machine Learning Research, JMLR*, vol. 6, pp. 2153–2175, 2005.

[15] M. Derezinski, R. Khanna, and M. W. Mahoney, "Improved guarantees and a multiple-descent curve for column subset selection and the Nyström method," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4953–4964, 2020.

[16] K. Ueda and N. Yamashita, "Convergence properties of the regularized Newton method for the unconstrained nonconvex optimization," *Applied Mathematics and Optimization*, vol. 62, pp. 27–46, 2010.

[17] H. Tankaria, S. Sugimoto, and N. Yamashita, "A regularized limited memory BFGS method for large-scale unconstrained optimization and its efficient implementations," *Computational Optimization and Applications*, vol. 82, no. 1, pp. 61–88, 2022.

[18] H. Kasai, "SGDLibrary: A MATLAB library for stochastic optimization algorithms," *Journal of Machine Learning Research, JMLR*, vol. 18, pp. 215:1–215:5, 2017.

[19] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

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

[21] 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.

[22] "Dataset : brain mri images for brain tumor detection," *https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection*, 2020.