RANDOMIZED SUB-SAMPLED METHODS FOR MATRIX APPROXIMATION

ANDREW AZZAM*, BENJAMIN ONG†, AND ALLAN STRUTHERS‡

Abstract. We introduce a family of randomized iterative algorithms to approximate matrices by sub-sampling with respect to randomized input and output subspaces. These sub-sampled algorithms have several advantages over existing sample-based Quasi-Newton algorithms: the sub-sampled algorithms use significantly less data per iteration and can be tuned (by selecting input and output dimensions) for modern hardware architectures. We establish convergence rates for the algorithms and demonstrate consistent practical performance on a wide class of test matrices. An accelerated scheme is introduced and compared numerically against existing accelerated Quasi-Newton schemes.

Key words. matrix approximation, randomized algorithms, Quasi-Newton.

AMS subject classifications. 68W20, 68W25, 65F35, 90C53

1. Introduction.

Randomized sampled matrix approximations can be used to accelerate many numerical algorithms; the extensive list of articles citing a comprehensive review article [6] provides a wide range of applications and algorithms. Sampled Quasi-Newton schemes [5] approximate $A \in \mathbb{R}^{m\times n}$ by iteratively evaluating

$$U^T A \in \mathbb{R}^{s_1\times n} \text{ and/or } A\,V \in \mathbb{R}^{m\times s_2},$$

with $U^T \sim \mathcal{N}(0,1)^{s_1\times m}$, $V \sim \mathcal{N}(0,1)^{n\times s_2}$, $s_1 \ll m$, and $s_2 \ll n$. Here and in what follows, $\hat{x} \sim p^{i\times j}$ indicates that $\hat{x} \in \mathbb{R}^{i\times j}$ has entries drawn from the distribution p and $\bar{x} = \mathbf{E}[\hat{x}]$ is the expectation of $\hat{x}$. The primary motivation for existing sampled algorithms is to reduce the data footprint of the algorithms and to provide (partially) tunable algorithms for modern hardware architectures.

In this work, we introduce sub-sampled algorithms which sample row and column spaces simultaneously by evaluating

$$U^T A\,V \in \mathbb{R}^{s_1\times s_2}.$$

The data footprint of the sub-sampled algorithms we develop ($s_1 \times s_2$) is significantly smaller than the data footprint of existing sampled algorithms ($s_1 \times n$ and/or $m \times s_2$), and provides additional tuning parameters for available hardware.

Work Estimates and Relationship to Quasi-Newton Algorithms. We assume 'black-box' matrix access which efficiently computes products $A\,V$, $U^T A$ and/or $U^T A\,V$ for $U^T \in \mathbb{R}^{s_1\times m}$ and $V \in \mathbb{R}^{n\times s_2}$ at a cost proportional to the number of output entries; this will be the primary cost metric for our algorithms. An explicit example of this black-box matrix access is the evaluation of a second derivative using Algorithmic Differentiation in Forward-Forward mode [8]. Our goal is to develop and analyze iterative approximations to A which use sub-samples. The resulting algorithms are strongly connected to and motivated by Quasi-Newton algorithms from nonlinear optimization and sampled Quasi-Newton algorithms [5]. When comparing computational cost of the algorithms, $A\,V$ is a sample of $m\,s_2$ elements, $U^T A$ is a sample of $s_1\,n$ elements, while $U^T A\,V$ is a sub-sample of $s_1\,s_2$ elements. A long-term

*Department of Mathematical Sciences, Michigan Technological University (atazzam@mtu.edu).
†Department of Mathematical Sciences, Michigan Technological University (ongbw@mtu.edu).
‡Department of Mathematical Sciences, Michigan Technological University (struther@mtu.edu).

research objective is to create limited-memory variants (motivated by L-BFGS [9]) of these sub-sampled algorithms.

Various sampled Quasi-Newton methods [4, 5] have been developed based on block updates [1]. The block optimization algorithm [4] takes several Quasi-Newton steps with a fixed Hessian approximation (to reduce linear algebra) before performing a block update, and accelerates terminal convergence with an ingenious heuristic. Several variant block updates (based on traditional minimum change justifications for DFP and BFGS [9]) are developed and used in iterative algorithms for approximate inverses [5]: additional theory and heuristic acceleration techniques have also been explored [7].

In this article, minimum-change motivated arguments are used to develop a family of updates which iteratively incorporate $s_1 \times s_2$ pieces of information from the sub-sample $U^T A V \in \mathbb{R}^{s_1 \times s_2}$ to generate a sequence of approximations to A. The data footprint of each iteration is $s_1 \times s_2$ which is substantially smaller than that of the sampled algorithms [5, 4, 7] and can be fully tuned to the available hardware. Convergence rates are derived which are comparable with existing Quasi-Newton algorithms [5, 4, 7].

1.1. Outline. The paper is organized as follows. Section 2 introduces our randomized sub-sampled methods and their relationship with those in the literature. Section 3 provides convergence rates for the sub-sampled methods. Section 4 compares the results of numerical experiments with those of equivalent sampled methods. Section 5 develops block power iteration accelerated sub-sampled algorithms and numerically compares them to equivalent sampled based algorithms with similar heuristics.

1.2. Notation. Throughout the article: SPD is an acronym for symmetric positive definite and W denotes SPD weight matrices; superscript $^+$ denotes the Moore-Penrose pseudo-inverse; $\langle X, Y \rangle_F = \operatorname{Tr}\left[X^T Y\right]$ and $\|X\|_F^2 = \langle X, X \rangle_F$ denote the Frobenius inner product and norm; residuals are measured using weighted norms,

$$\|X\|^2_{F(W_1^{-1}, W_2^{-1})} = \|W_1^{-1/2} X W_2^{-1/2}\|_F^2 \quad \text{and} \quad \|X\|^2_{F(W^{-1})} = \|W^{-1/2} X W^{-1/2}\|_F^2,$$

with conforming SPD weights W_1, W_2 and W; algorithms are developed using the W-weighted projector, which projects onto the column space of $W U$,

$$\mathcal{P} = P_{W^{-1},U} = W U (U^T W U)^{-1} U^T. \tag{1.1}$$

The weighted projector satisfies

$$\mathcal{P} W = W \mathcal{P}^T = \mathcal{P} W \mathcal{P}^T \quad \text{and} \quad W^{-1} \mathcal{P} = \mathcal{P}^T W^{-1} = \mathcal{P}^T W^{-1} \mathcal{P}. \tag{1.2}$$

2. Randomized Approximation Methods. Numerical optimization texts [9] motivate and derive Quasi-Newton update schemes for SPD matrices A using constrained minimum change criteria (for $B \approx A$ and $H \approx A^{-1}$) in weighted Frobenius norms. Traditional algorithms are derived by selecting different weights. Block update algorithms (sampled algorithms in our terminology) which update multiple directions simultaneously are derived [5, 4] similarly. The KKT equations [9] for the quadratic

programs,

$$B_{k+1} = \arg\min_{B} \left\{ \frac{1}{2}\|B - B_k\|^2_{F(W^{-1})} \mid B\,U = A\,U \text{ and } B = B^T \right\} \tag{2.1}$$

$$H_{k+1} = \arg\min_{H} \left\{ \frac{1}{2}\|H - H_k\|^2_{F(W^{-1})} \mid U = H\,A\,U \text{ and } H = H^T \right\} \tag{2.2}$$

gives two different updates using the same sample $A\,U_k$: the update to B_k produces B_{k+1}, an improved approximation to A; the update to H_k produces H_{k+1}, an improved approximation to A^{-1}. The linear algebraic updates that result are

$$\begin{aligned} B_{k+1} &= B_k + \mathcal{P}_B(A - B_k) + (A - B_k)\mathcal{P}_B^T - \mathcal{P}_B(A - B_k)\mathcal{P}_B^T, \\ H_{k+1} &= H_k + \mathcal{P}_H(A^{-1} - H_k) + (A^{-1} - H_k)\mathcal{P}_H^T - \mathcal{P}_H(A^{-1} - H_k)\mathcal{P}_H^T, \end{aligned} \tag{2.3}$$

where the weighted projectors $\mathcal{P}_B$ and $\mathcal{P}_H$ defined by Eq. (1.1) are

$$\begin{aligned} \mathcal{P}_B &= P_{W^{-1},U} = W\,U(U^T\,W\,U)^{-1}U^T, \\ \mathcal{P}_H &= P_{W^{-1},A\,U} = W\,A\,U(U^T\,A\,W\,A\,U)^{-1}U^T\,A. \end{aligned}$$

Block DFP [10] is the B formulation with $W = A$,

$$B_{k+1} = (I_n - \mathcal{P}_{\text{DFP}})\; B_k(I_n - \mathcal{P}_{\text{DFP}}^T) + \mathcal{P}_{\text{DFP}}\,A. \tag{2.4}$$

where

$$\mathcal{P}_{\text{DFP}} = P_{A,U} = A\,U(U^T\,A\,U)^{-1}U^T.$$

Block BFGS [5, 4] is the H formulation (inverted using the Shermann-Morisson-Woodbury formula) with $W = A^{-1}$,

$$B_{k+1} = B_k - B_kU\left(U^TB_kU\right)^{-1}U^TB_k + AU\left(U^TAU\right)^{-1}U^TA. \tag{2.5}$$

Note, the two other plausible versions ($W = A$ in the H update and $W = A^{-1}$ in the B update) produce updates containing A^{-1}. Such updates are not useful.

Our goal is to use subsamples $U^T\,A\,V$ to construct approximations to A. We only consider updates B satisfying

$$U^T\,B\,V = U^T\,A\,V.$$

There are many such potential updates, for instance, $U\,U^+A\,V\,V^+$ minimizes the unweighted Frobenius norm.

2.1. General Sub-Sampled Update. We define updates using the minimal change criterion;

$$B_{k+1} = \arg\min_{B} \left\{ \frac{1}{2}\|B - B_k\|^2_{F(W_1^{-1},W_2^{-1})} \mid U^T\,B\,V = U^T\,A\,V \right\}, \tag{2.6}$$

which defines the self-correcting update (for details see Appendix B)

$$B_{k+1} = B_k + P_{W_1^{-1},U^k}(A - B_k)P^T_{W_2^{-1},V^k}. \tag{2.7}$$

By construction, Eq. (2.7) simply corrects the sub-sampled mismatch $U^T(A - B_k)V$. It cannot increase the weighted Frobenius norm $\|A - B_k\|^2_{F(W_1^{-1}, W_2^{-1})}$ and, provided the sub-space sequences U_k and V_k eventually exhaust the underlying spaces, the weighted residual must decrease monotonically to zero.

Given $A \in \mathbb{R}^{m\times n}$, an initial estimate $B_0 \in \mathbb{R}^{m\times n}$, sub-sample sizes $\{s_1, s_2\}$, and SPD weights $\{W_1, W_2\}$, Eq. (2.7) generates a sequences $\{B_k\}$ that converges to A monotonically in the appropriate weighted Frobenius norm. The resulting algorithm is summarized in Algorithm 2.1: boxed values show the number of samples of A on a pseudocode line; the return-line double boxed value is the total number of samples.

Algorithm 2.1 NS: Non-Symmetric Sub-Sampled Approximation

Require: $B_0 \in \mathbb{R}^{m\times n}$, SPD $W_1 \in \mathbb{R}^{m\times m}$, $W_2 \in \mathbb{R}^{n\times n}$, $\{s_1, s_2\} \in \mathbb{N}$.

1: **repeat** $\{k = 0, 1, \ldots\}$
2: Sample $U_k \sim N(0,1)^{m\times s_1}$ and $V_k \sim N(0,1)^{n\times s_2}$
3: Compute residual $\Lambda_k = U_k^T A\, V_k - U_k^T B_k V_k \in \mathbb{R}^{s_1\times s_2}$ ..... $\boxed{s_1 s_2}$
4: Update $B_{k+1} = B_k + W_1 U_k (U_k^T W_1 U_k)^{-1} \Lambda_k (V_k^T W_2 V_k)^{-1} V_k^T W_2$
5: **until** convergence
6: **return** B_{k+1} ..... $\boxed{\boxed{(k+1)\,(s_1 s_2)}}$

Algorithm 2.1, does not generate symmetric approximations for symmetric A. The next two sections modify the basic algorithm to preserve symmetry. When discussing symmetric updates we will always use symmetric initializations $B_0 = B_0^T$ and symmetric weights $W = W_1 = W_2$.

2.2. Symmetric Update. Symmetric sampling, $V_k = U_k$, and weighting $W = W_1 = W_2$ in Algorithm 2.1 with symmetric initialization $B_0 = B_0^T$ gives a sequence of symmetric approximations, B_k, to a symmetric $n \times n$ matrix A. The resulting algorithm is summarized in Algorithm 2.2 with sample counts boxed as before.

Algorithm 2.2 SS1: Symmetric Sub-Sampled Approximation

Require: $B_0 \in \mathbb{R}^{n\times n}$ satisfying $B_0^T = B_0$, SPD $W \in \mathbb{R}^{n\times n}$, $s_1 \in \mathbb{N}$.

1: **repeat** $\{k = 0, 1, \ldots\}$
2: Sample $U_k \sim \mathcal{N}(0,1)^{n\times s_1}$
3: Compute residual $\Lambda_k = U_k^T A\, U_k - U_k^T B_k U_k \in \mathbb{R}^{s_1\times s_1}$ ..... $\boxed{s_1^2}$
4: Compute $\tilde{P}_k = W\, U_k (U_k^T W\, U_k)^{-1}$
5: Update $B_{k+1} = B_k + \tilde{P}_k \Lambda_k \tilde{P}_k^T$
6: **until** convergence
7: **return** B_{k+1} ..... $\boxed{\boxed{(k+1)\,(s_1^2)}}$

Remark 2.1. Algorithm 2.2 (with $W = I_n$) can be viewed as a sub-sampled BFGS update: apply the orthogonal projection $\mathcal{P}_{I_n,U} = UU^T$ to both sides of Eq. (2.5) to get Algorithm 2.2 with $W = I_n$. Algorithm 2.2 can be viewed as a sub-sampled DFP update.

Remark 2.2. Algorithm 2.2 does not preserve positivity. A non-SPD result can

be observed when

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 9 \end{bmatrix}, \quad \text{and} \quad U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

2.3. Multi-Step Symmetric Updates. An alternative approach to generate symmetric approximations is to symmetrize Eq. (2.7) as follows

$$\begin{aligned} B_{k+1/2} &= B_k + P_{W^{-1},U^k}(A - B_k)P^T_{W^{-1},V^k} \\ B_{k+1} &= \frac{1}{2}\left(B_{k+1/2} + B^T_{k+1/2}\right). \end{aligned} \tag{2.8}$$

For symmetric A and B_0, it can be shown that the convergence rate for Eq. (2.8) is comparable to Algorithm 2.2. However, for symmetric A the additional sample,

$$P_{W_2^{-1},V^k} A\, P^T_{W_1^{-1},U^k} = \left(P_{W_1^{-1},U^k} A\, P^T_{W_2^{-1},V^k}\right)^T,$$

can be directly incorporated to give

$$\begin{aligned} B_{k+1/3} &= B_k + P_{W_1^{-1},U^k}(A - B_k)P^T_{W_2^{-1},V^k} \\ B_{k+2/3} &= B_{k+1/3} + P_{W_2^{-1},V^k}(A - B^T_{k+1/3})P^T_{W_1^{-1},U^k} \\ B_{k+1} &= \frac{1}{2}\left(B_{k+2/3} + B^T_{k+2/3}\right), \end{aligned} \tag{2.9}$$

where the last line again enforces symmetry. We summarize this two-step symmetric algorithm in Algorithm 2.3 with sample counts boxed as before. This two-step algorithm has superior convergence properties.

Algorithm 2.3 SS2: Two-Step Symmetric Sub-Sampled Approximation

Require: $B_0 \in \mathbb{R}^{n\times n}$ satisfying $B_0 = B_0^T$, SPD $W \in \mathbb{R}^{m\times m}$, $\{s_1, s_2\} \in \mathbb{N}$.

1: **repeat** $\{k = 0, 1, \ldots\}$
2: Sample $U_k \sim N(0,1)^{n\times s_1}$ and $V_k \sim N(0,1)^{n\times s_2}$
3: Compute residual $\Lambda_k = U_k^T A\, V_k - U_k^T B_k V_k \in \mathbb{R}^{s_1\times s_2}$ ..... $\boxed{s_1 s_2}$
4: Compute $B_{k+1/3} = B_k + W\, U_k (U_k^T W\, U_k)^{-1}\Lambda_k (V_k^T W\, V_k)^{-1} V_k^T W$
5: Compute residual $\Lambda_{k+1/3} = (U_k^T A\, V_k)^T - V_k^T B_{k+1/3} U_k \in \mathbb{R}^{s_2\times s_1}$
6: Compute $B_{k+2/3} = B_{k+1/3} + W\, V_k (V_k^T W\, V_k)^{-1}\Lambda_{k+1/3} (U_k^T W\, U_k)^{-1} U_k^T W$
7: Update $B_{k+1} = \frac{1}{2}(B_{k+2/3} + B^T_{k+2/3})$
8: **until** convergence
9: **return** B_{k+1} ..... $\boxed{\boxed{(k+1)\,(s_1 s_2)}}$

3. Convergence Analysis. Our convergence results rely extensively on properties of randomly generated projectors. In our computational tests, projections are generated by orthogonalizing matrices with individual entries drawn from $N(0,1)$. For square matrices, this process gives rotations drawn from a distribution which is invariant under rotations [11]. Our algorithms use symmetric weighted rank s projectors,

$$\hat{z} = W^{1/2} U (U^T W U)^{-1} U^T W^{1/2}, \tag{3.1}$$

where W is an SPD weight matrix and U is simply the first s columns of such a random rotation. The expectation of random symmetric $n \times n$ projections $\hat{z}$, $\mathbf{E}[\hat{z}] \in \mathbb{R}^{n\times n}$, is crucial in our analysis. We write z_i for the eigenvalues of $\mathbf{E}[\hat{z}]$ with the standard ordering $z_1 \leq z_2 \leq \cdots \leq z_n$. The extreme eigenvalues z_1 and z_n determine our algorithms convergence with the best results when $z_1 = z_n$.

For clarity the next section collects a number of useful definitions and lemmas.

3.1. Mathematical Preliminaries.

DEFINITION 3.1. *A random matrix $\hat{X} \in \mathbb{R}^{m\times n}$ is rotationally invariant if the distribution of $Q_m\,\hat{X}\,Q_n$ is the same for all rotations $Q_i \in \mathcal{O}(i)$.*

LEMMA 3.2 (Random Projections). *For any distribution $\hat{z}$ of real, symmetric rank s projectors in $\mathbb{R}^n$,*

$$0 \leq \lambda_{\min}(\boldsymbol{E}[\hat{z}]) \leq \frac{s}{n} \leq \lambda_{\max}(\boldsymbol{E}[\hat{z}]) \leq 1. \tag{3.2}$$

Further, if $\hat{z}$ is rotationally invariant, then $\boldsymbol{E}[\hat{z}] = \frac{s}{n} I_n$.

Proof. Let $x \in \mathbb{R}^n$ with $x^T x = 1$. Since $\hat{z}$ is a projector,

$$0 = \lambda_{\min}(\hat{z}) \leq x^T \hat{z}\, x \leq \lambda_{\max}(\hat{z}) = 1.$$

Since $\mathbf{E}[x^T \hat{z}\, x] = x^T \mathbf{E}[\hat{z}]\, x$, taking the expectation gives

$$0 \leq x^T \mathbf{E}[\hat{z}]\, x \leq 1,$$

for all unit vectors x. Since the trace is linear, the sum of the eigenvalues of $\mathbf{E}[\hat{z}]$ equals $\mathrm{Tr}(\mathbf{E}[\hat{z}]) = \mathbf{E}[\mathrm{Tr}(\hat{z})] = E(s) = s$, which establishes Eq. (3.2). Rotationally invariant $\hat{z}$ satisfy $\mathbf{E}[\hat{z}] = \alpha I_n$ since for all $Q_1, Q_2 \in \mathcal{O}(n)$,

$$\mathbf{E}[\hat{z}] = \mathbf{E}[Q_1\, \hat{z}\, Q_2] = Q_1\, \mathbf{E}[\hat{z}]\, Q_2,$$

Using a similar argument, linearity of the trace gives $\alpha = \frac{s}{n}$. ∎

LEMMA 3.3 (Projection Cancellation). *For $R \in \mathbb{R}^{m\times n}$ and conforming symmetric projections $\hat{y}, \hat{z}$,*

$$\langle R\,\hat{z}, R\,\hat{z}\rangle_F = \langle R, R\,\hat{z}\rangle_F \tag{3.3}$$

$$\langle \hat{y}\,R\,\hat{z}, \hat{y}\,R\,\hat{z}\rangle_F = \langle \hat{y}\,R\,\hat{z}, R\,\hat{z}\rangle_F = \langle \hat{y}\,R\,\hat{z}, R\rangle_F \tag{3.4}$$

Proof. Expanding the definition of Eq. (3.3),

$$\langle R\hat{z}, R\hat{z}\rangle_F = \mathrm{Tr}[\hat{z}^T R^T R\,\hat{z}] = \mathrm{Tr}[R^T R\,\hat{z}\,\hat{z}^T] = \mathrm{Tr}[R^T R\,\hat{z}] = \langle R, R\,\hat{z}\rangle_F,$$

since $\mathrm{Tr}[A\,B] = \mathrm{Tr}[B\,A]$ and $\hat{z}$ is a projector. Similarly for Eq. (3.4),

$$\langle \hat{y}\,R\,\hat{z}, \hat{y}\,R\,\hat{z}\rangle_F = \mathrm{Tr}[\hat{z}^T R^T \hat{y}^T \hat{y}\,R\,\hat{z}] = \mathrm{Tr}[\hat{z}^T R^T \hat{y}^T R\,\hat{z}] = \langle \hat{y}\,R\,\hat{z}, R\,\hat{z}\rangle_F,$$

$$\langle \hat{y}\,R\,\hat{z}, R\,\hat{z}\rangle_F = \mathrm{Tr}[\hat{z}^T R^T \hat{y}^T R\,\hat{z}] = \mathrm{Tr}[\hat{z}\,\hat{z}^T R^T \hat{y}^T R] = \mathrm{Tr}[\hat{z}^T R^T \hat{y}^T R] = \langle \hat{y}\,R\,\hat{z}, R\rangle_F.$$ ∎

LEMMA 3.4 (Spectral Bounds). *For any $R \in \mathbb{R}^{m\times n}$ and conforming symmetric positive semi-definite matrices S_1, S_2, and (in the special case $m = n$) S we have the bounds:*

$$\lambda_{\min}(S_1)\langle R, R\rangle_F \leq \langle S_1\,R, R\rangle_F \leq \lambda_{\max}(S_1)\langle R, R\rangle_F, \tag{3.5}$$

$$\lambda_{\min}(S_2)\langle R, R\rangle_F \leq \langle R, R\,S_2\rangle_F \leq \lambda_{\max}(S_2)\langle R, R\rangle_F, \tag{3.6}$$

$$\lambda_{\min}(S)^2\langle R, R\rangle_F \leq \langle S\,R, R\,S\rangle_F \leq \lambda_{\max}(S)^2\langle R, R\rangle_F. \tag{3.7}$$

Proof. To establish Eq. (3.5) write $R = [r_1|r_2|\cdots|r_n]$ and note that the results follows immediately from $\langle S_1\, R, R\rangle_F = \sum_{i=1}^n r_i^T S_1 r_i$ and $\langle R, R\rangle_F = \sum_{i=1}^n r_i^T r_i$ since

$$\sum_{i=1}^n \lambda_{\min}(S_1)\, r_i^T r_i \le \sum_{i=1}^n r_i^T S_1 r_i \le \sum_{i=1}^n \lambda_{\max}(S_1)\, r_i^T r_i. \tag{3.8}$$

Equation (3.6) follows directly from Eq. (3.5) applied to S_2 and R^T since

$$\langle R, R\, S_2\rangle_F = \langle R^T, S_2^T R^T\rangle_F = \langle S_2^T\, R^T, R^T\rangle_F = \langle S_2\, R^T, R^T\rangle_F.$$

To establish Eq. (3.7) note that for symmetric positive semi-definite T

$$\langle T^2 R, R\, T^2\rangle_F = \langle T\, R, T^2\, T\, R\rangle_F \quad \text{and} \quad \langle T\, R, T\, R\rangle_F = \sum_{i=1}^n r_i^T T^2 r_i.$$

Equation (3.7) then follows immediately with $T = S^{1/2}$ from Eq. (3.5) applied to $S_1 = T^2$ and the standard bound Eq. (3.8) with $S_1 = T^2$. ∎

3.2. Convergence Theorems. Convergence results for Algorithms 2.1 to 2.3. are for $\mathbf{E}[\|B - A\|_F^2]$. Such results dominate similar results for $\|\mathbf{E}[B - A]\|_F^2$ since

$$\|\mathbf{E}\,[B - A]\|_F^2 = \mathbf{E}\left[\|B - A\|_F^2\right] - \mathbf{E}\left[\|B - \mathbf{E}\,[B]\|_F^2\right],$$

as shown in [5].

THEOREM 3.5 (Convergence of NS Algorithm 2.1). *Let $A \in \mathbb{R}^{m\times n}$ and $W_1 \in \mathbb{R}^{m\times m}$ and $W_2 \in \mathbb{R}^{n\times n}$ be fixed SPD weight matrices. If $U_k \in \mathbb{R}^{m\times s_1}$ and $V_k \in \mathbb{R}^{n\times s_2}$ are random, independently selected matrices with full column rank (with probability one), then Eq. (2.7) generates a sequence, B_k, from an initial guess $B_0 \in \mathbb{R}^{m\times n}$ satisfying*

$$\boldsymbol{E}\left[\|B_{k+1} - A\|^2_{F(W_1^{-1},W_2^{-1})}\right] \le (\rho_{NS})^k\, \boldsymbol{E}\left[\|B_0 - A\|^2_{F(W_1^{-1},W_2^{-1})}\right],$$

where $\rho_{NS} = 1 - \lambda_{\min}(\boldsymbol{E}[\hat{y}])\,\lambda_{\min}(\boldsymbol{E}[\hat{z}])$, with

$$\hat{y}_k = W_1^{1/2} U_k (U_k^T W_1 U_k)^{-1} U_k^T W_1^{1/2}, \quad \hat{z}_k = W_2^{1/2} V_k (V_k^T W_2 V_k)^{-1} V_k^T W_2^{1/2}. \tag{3.9}$$

Proof. Define the kth residual as $R_k := W_1^{-1/2}(B_k - A)W_2^{-1/2}$. With some algebraic manipulation, Eq. (2.7) can be re-written as

$$R_{k+1} = R_k - \hat{y}_k R_k \hat{z}_k. \tag{3.10}$$

Computing the squared Frobenius norm of Eq. (3.10),

$$\begin{aligned}
\langle R_{k+1}, R_{k+1}\rangle_F &= \langle R_k - \hat{y}_k R_k \hat{z}_k, R_k - \hat{y}_k R_k \hat{z}_k\rangle_F \\
&= \langle R_k, R_k\rangle_F - \langle R_k, \hat{y}_k R_k \hat{z}_k\rangle_F - \langle \hat{y}_k R_k \hat{z}_k, R_k\rangle_F + \langle \hat{y}_k R_k \hat{z}_k, \hat{y}_k R_k \hat{z}_k\rangle_F \\
&= \langle R_k, R_k\rangle_F - \langle \hat{y}_k R_k \hat{z}_k, R_k \hat{z}_k\rangle_F,
\end{aligned}$$

where we have made use of Lemma 3.3. Taking the expected value with respect to independent samples U_k (leaving V_k and R_k fixed) gives

$$\begin{aligned}
\mathbf{E}\left[\|R_{k+1}\|_F^2 \mid V_k, R_k\right] &= \langle R_k, R_k\rangle_F - \langle \mathbf{E}[\hat{y}_k] R_k \hat{z}_k, R_k \hat{z}_k\rangle_F \\
&\le \langle R_k, R_k\rangle_F - \lambda_{\min}(\mathbf{E}[\hat{y}_k])\langle R_k \hat{z}_k, R_k \hat{z}_k\rangle_F \\
&\le \langle R_k, R_k\rangle_F - \lambda_{\min}(\mathbf{E}[\hat{y}_k])\langle R_k, R_k \hat{z}_k\rangle_F,
\end{aligned} \tag{3.11}$$

where we applied Lemma 3.4 to the symmetric positive semi-definite matrix $\mathbf{E}[\hat{y}_k]$, and utilized Eq. (3.3). Taking the expected value with respect to independent samples V_k and leaving R_k fixed gives

$$\begin{aligned}\mathbf{E}[\|R_{k+1}\|_F^2 \mid R_k] &\leq \langle R_k, R_k\rangle_F - \lambda_{\min}(\mathbf{E}[\hat{y}_k])\langle R_k, R_k\mathbf{E}[\hat{z}_k]\rangle_F \\ &\leq \langle R_k, R_k\rangle_F - \lambda_{\min}(\mathbf{E}[\hat{y}_k])\lambda_{\min}(\mathbf{E}[\hat{z}_k])\langle R_k, R_k\rangle_F.\end{aligned}$$

Taking the full expectation gives

$$\begin{aligned}\mathbf{E}[\|R_{k+1}\|_F^2] &\leq \mathbf{E}\left[\langle R_k, R_k\rangle_F\right] - \lambda_{\min}(\mathbf{E}[\hat{y}_k])\lambda_{\min}(\mathbf{E}[\hat{z}_k])\mathbf{E}\left[\langle R_k, R_k\rangle_F\right] \\ &= (1 - \lambda_{\min}(\mathbf{E}[\hat{y}_k])\lambda_{\min}(\mathbf{E}[\hat{z}_k]))\mathbf{E}[\langle R_k, R_k\rangle_F].\end{aligned}$$

Since

$$\mathbf{E}[\|R_{k+1}\|_F^2] = \mathbf{E}[\|B_k - A\|^2_{F(W_1^{-1}, W_2^{-1})},$$

un-rolling the recurrence for k iterations yields the desired result. ∎

Remark 3.6. The condition that U_k and V_k are chosen independently of each other is required to justify $\mathbf{E}[\langle \hat{y}_k R_k \hat{z}_k, R_k\hat{z}_k\rangle_F] = \langle \mathbf{E}[\hat{y}_k] R_k\hat{z}_k, R_k\hat{z}_k\rangle_F$.

THEOREM 3.7 (Convergence of SS1 Algorithm 2.2). *Let $A, W \in \mathbb{R}^{n\times n}$ be fixed SPD matrices and $U_k \in \mathbb{R}^{n\times s}$ be a randomly selected matrix having full column rank with probability* 1. *If $B_0 \in \mathbb{R}^{n\times n}$ is an initial guess for A with $B_0 = B_0^T$, then after applying k iterations of the update in Algorithm 2.2, the iterates B_{k+1} satisfy*

$$\boldsymbol{E}[\|B_{k+1} - A\|^2_{F(W^{-1})}] \leq (\rho_{SS1})^k \boldsymbol{E}[\|B_0 - A\|^2_{F(W^{-1})}], \tag{3.12}$$

where $\rho_{SS1} = 1 - \lambda_{\min}(\boldsymbol{E}[\hat{z}])^2$ and

$$\hat{z}_k = W^{1/2}U_k(U_k^T W\, U_k)^{-1}U_k^T W^{1/2}.$$

Proof. Following similar steps outlined in the proof in Theorem 3.5, we arrive at

$$\langle R_{k+1}, R_{k+1}\rangle_F = \langle R_k, R_k\rangle_F - \langle R_k, \hat{z}_k R_k \hat{z}_k\rangle_F.$$

Taking the expected value with respect to U_k leaving R_k fixed we have

$$\begin{aligned}\mathbf{E}\left[\|R_{k+1}\|_F^2 \mid R_k\right] &= \langle R_k, R_k\rangle_F - \mathbf{E}\left[\langle R_k, \hat{z}_k R_k\hat{z}_k\rangle_F\right] \\ &= \langle R_k, R_k\rangle_F - \mathbf{E}\left[\mathrm{Tr}[R_k^T\hat{z}_k R_k\hat{z}_k]\right] \\ &= \langle R_k, R_k\rangle_F - \mathrm{Tr}\left[\mathbf{E}\left[R_k\hat{z}_k R_k\hat{z}_k\right]\right] \\ &\leq \langle R_k, R_k\rangle_F - \mathrm{Tr}\left[\mathbf{E}\left[R_k\hat{z}_k\right]^2\right],\end{aligned}$$

where the inequality arises from application of Jensen's Inequality. Simplifying and applying Eq. (3.7),

$$\begin{aligned}\mathbf{E}[\|R_{k+1}\|^2_{F(W^{-1})} \mid R_k] &\leq \langle R_k, R_k\rangle_F - \mathrm{Tr}\left[\mathbf{E}\left[R_k\hat{z}_k\right]^2\right] \\ &= \langle R_k, R_k\rangle_F - \mathrm{Tr}\left[R_k\mathbf{E}\left[\hat{z}_k\right] R_k\mathbf{E}\left[\hat{z}_k\right]\right] \\ &= \langle R_k, R_k\rangle_F - \langle \mathbf{E}[\hat{z}_k]R_k, R_k\mathbf{E}[\hat{z}_k]\rangle_F \\ &\leq \langle R_k, R_k\rangle_F - \lambda_{\min}(\mathbf{E}[\hat{z}_k])^2\langle R_k, R_k\rangle_F.\end{aligned}$$

Taking the full expectation and un-rolling the recurrence yields the desired result. ∎

THEOREM 3.8 (Convergence of SS2 Algorithm 2.3). *Let A, U_k, V_k and B_0 be defined as in Theorem 3.5, and let W be a fixed SPD matrix. After applying k iterations of Algorithm 2.3 with $W = W_1 = W_2$, the iterates B_k satisfy*

$$\boldsymbol{E}\left[\|B_k - A\|^2_{F(W^{-1})}\right] \leq (\rho_{SS2})^k \boldsymbol{E}\left[\|B_0 - A\|^2_{F(W^{-1})}\right],$$

where

$$\rho_{SS2} = 1 - 2\lambda_{\min}(\boldsymbol{E}[\hat{y}])\lambda_{\min}(\boldsymbol{E}[\hat{z}]) + \lambda_{\min}(\boldsymbol{E}[\hat{y}])^2\lambda_{\min}(\boldsymbol{E}[\hat{z}])^2.$$

Proof. Define kth residual R_k and projectors $\hat{y}_k$ and $\hat{z}_k$ as in Theorem 3.5 with $W = W_1 = W_2$. The iteration given in Eq. (2.9) can be re-written in terms of R_k as follows.

$$\begin{aligned} R_{k+1/3} &= R_k - \hat{y}_k R_k \hat{z}_k \\ R^T_{k+2/3} &= R^T_{k+1/3} - \hat{z}_k R^T_{k+1/3} \hat{y}_k \\ R_{k+1} &= \frac{1}{2}\left(R_{k+2/3} + R^T_{k+2/3}\right) \end{aligned}$$

Theorem 3.5 gives

$$\mathbf{E}\left[\left\|R_{k+1/3}\right\|^2_F\right] \leq (\rho_{\rm NS})\mathbf{E}\left[\|R_k\|^2_F\right],$$

and a repeated application of Theorem 3.5 gives

$$\mathbf{E}\left[\left\|R_{k+2/3}\right\|^2_F\right] \leq (\rho_{\rm NS})\mathbf{E}\left[\|R_{k+1/3}\|^2_F\right] \leq (\rho_{\rm NS})^2\mathbf{E}\left[\|R_k\|^2_F\right].$$

Lastly, we observe via the triangle inequality that

$$\begin{aligned} \mathbf{E}\left[\|R_{k+1}\|^2_F\right] &= \mathbf{E}\left[\left\|\frac{1}{2}\left(R_{k+2/3} + R^T_{k+2/3}\right)\right\|^2_F\right] \\ &\leq \frac{1}{2}\mathbf{E}\left[\left\|R_{k+2/3}\right\|^2_F\right] + \frac{1}{2}\mathbf{E}\left[\left\|R^T_{k+2/3}\right\|^2_F\right] \\ &= (\rho_{\rm NS})^2\mathbf{E}\left[\|R_k\|^2_F\right], \end{aligned}$$

Un-rolling the loop for k iterations gives the desired result. ∎

3.3. Optimal Fixed Weight Convergence Rates. To discuss convergence rates, we define

$$\begin{aligned} \rho_{\rm NS}(y_1, z_1) &= 1 - y_1 z_1, \\ \rho_{\rm SS1}(z_1) &= 1 - z_1^2, \\ \rho_{\rm SS2}(y_1, z_1) &= (1 - y_1 z_1)^2, \end{aligned} \tag{3.13}$$

and note that the convergence rates for Algorithms 2.1 to 2.3 can be expressed as

$$\|R_{k+1}\|^2_{F(W_1^{-1}, W_2^{-1})} \leq \rho \|R_k\|^2_{F(W_1^{-1}, W_2^{-1})} \tag{3.14}$$

with the appropriate ρ, Eq. (3.13), evaluated at $y_1 = \lambda_{\min}(\mathbf{E}[\hat{y}])$ and $z_1 = \lambda_{\min}(\mathbf{E}[\hat{z}])$. Since any symmetric rank s random projection $\hat{z}$ on $\mathbb{R}^n$ satisfies $0 \leq z_1 \leq \frac{s}{n} \leq z_n \leq 1$ and rotationally invariant distributions, e.g. UU^+ with $U \sim N(0,1)^{n\times s}$, further satisfy $\mathbf{E}[\hat{z}] = \frac{s}{n}$, minimizing the various convergence rates ρ over the appropriate domains gives the following optimal rates.

COROLLARY 3.9. *(Optimal Convergence Rate) The optimal convergence rates for Algorithms 2.1 to 2.3 are obtained attained for U_k and V_k sampled from rotationally invariant distributions,*

$$\begin{aligned} \rho_{NS}^{opt} &= 1 - \frac{s_1}{m}\frac{s_2}{n}, \\ \rho_{SS1}^{opt} &= 1 - \left(\frac{s_2}{n}\right)^2, \\ \rho_{SS2}^{opt} &= \left(1 - \frac{s_1}{m}\frac{s_2}{n}\right)^2. \end{aligned} \tag{3.15}$$

Proof. Each part is simply the result of an explicit optimization,

$$\begin{aligned} \rho_{\mathrm{NS}}^{\mathrm{opt}} &= \min_{\substack{0\le y\le s_1/m \\ 0\le z\le s_2/n}} (1-yz) = 1 - \left(\frac{s_1}{m}\right)\left(\frac{s_2}{n}\right) \\ \rho_{\mathrm{SS1}}^{\mathrm{opt}} &= \min_{0\le z\le s_2/n} (1-z^2) = 1 - \left(\frac{s_2}{n}\right)^2 \\ \rho_{\mathrm{SS2}}^{\mathrm{opt}} &= \min_{\substack{0\le y\le s_1/m \\ 0\le z\le s_2/n}} (1-yz)^2 = \left(1 - \frac{s_1}{m}\frac{s_2}{n}\right)^2 \end{aligned}$$

∎

Remark 3.10. Theorems 3.5, 3.7, and 3.8 all assume the weight matrix W and distributions are fixed. All our non-accelerated numerical experiments use fixed wights and sample from fixed rotationally invariant distributions.

3.4. Theoretical Lower Bound for Convergence Rates. Lower bounds (entirely analogous to the upper bounds in Theorems 3.5, 3.7, and 3.8 but using the upper bounds in Lemma 3.4) are easily derived. For example, the two-sided error bound for Algorithm 2.1 is

$$\rho_{\mathrm{NS}}(y_m, z_n)\mathbf{E}[\|R_k\|_F^2] \le \mathbf{E}[\|R_{k+1}\|_F^2] \le \rho_{\mathrm{NS}}(y_1, z_1)\mathbf{E}[\|R_k\|_F^2],$$

where as before $y_1 \le y_2 \le \cdots \le y_m$ is the spectrum of $\mathbf{E}[\hat{y}]$, $z_1 \le z_2 \le \cdots \le z_n$ is the spectrum of $\mathbf{E}[\hat{z}]$ and the explicit form for ρ_{NS} is in Eq. (3.13). We collect the similar results for Algorithms 2.1 to 2.3 in Corollary 3.11.

COROLLARY 3.11 (Two-Sided Convergence Rates). *Given the assumptions of Theorems 3.5, 3.7, and 3.8 the explicit formulas Eq. (3.13) for ρ give two-sided bounds,*

$$\begin{aligned} \rho_{NS}(y_m, z_n)^k &\le \frac{\boldsymbol{E}\left[\|B_{k+1} - A\|^2_{F(W_1^{-1}, W_2^{-1})}\right]}{\|B_0 - A\|^2_{F(W_1^{-1}, W_2^{-1})}} \le \rho_{NS}(y_1, z_1)^k \\ \rho_{SS1}(z_n)^k &\le \frac{\boldsymbol{E}\left[\|B_{k+1} - A\|^2_{F(W^{-1})}\right]}{\|B_0 - A\|^2_{F(W^{-1})}} \le \rho_{SS1}(z_1)^k \\ \rho_{SS2}(y_n, z_n)^k &\le \frac{\boldsymbol{E}\left[\|B_{k+1} - A\|^2_{F(W^{-1})}\right]}{\|B_0 - A\|^2_{F(W^{-1})}} \le \rho_{SS2}(y_1, z_1)^k \end{aligned}$$

where y_1, y_m, z_1, z_n are the extreme eigenvalues of $\boldsymbol{E}[\hat{y}]$ and $\boldsymbol{E}[\hat{z}]$.

Proof. We prove the NS result; the proofs for SS1 and SS2 are analogous. Equation (3.11) of Theorem 3.5 and Lemma 3.4 gives

$$\begin{aligned}\mathbf{E}\left[\|R_{k+1}\|_F^2 \mid V_k, R_k\right] &= \langle R_k, R_k\rangle_F - \langle \mathbf{E}[\hat{y}_k] R_k \hat{z}_k, R_k \hat{z}_k\rangle_F \\ &\geq \langle R_k, R_k\rangle_F - \lambda_{\max}(\mathbf{E}[\hat{y}_k])\, \langle R_k, R_k \hat{z}_k\rangle_F.\end{aligned}$$

Following Theorem 3.5 (expectation in V_k and repeating the inequality) gives

$$\begin{aligned}\mathbf{E}[\|R_{k+1}\|_F^2 \mid R_k] &\geq \langle R_k, R_k\rangle_F - \lambda_{\max}(\mathbf{E}[\hat{y}_k])\, \langle R_k, R_k \mathbf{E}[\hat{z}_k]\rangle_F \\ &\geq \langle R_k, R_k\rangle_F - \lambda_{\max}(\mathbf{E}[\hat{y}_k])\, \lambda_{\max}(\mathbf{E}[\hat{z}_k])\, \langle R_k, R_k\rangle_F.\end{aligned}$$

Then taking the full expectation gives the inequality

$$\begin{aligned}\mathbf{E}[\|R_{k+1}\|_F^2] &\geq \mathbf{E}\left[\langle R_k, R_k\rangle_F\right] - \lambda_{\max}(\mathbf{E}[\hat{y}_k])\, \lambda_{\max}(\mathbf{E}[\hat{z}_k])\, \mathbf{E}\left[\langle R_k, R_k\rangle_F\right] \\ &= \left(1 - \lambda_{\max}(\mathbf{E}[\hat{y}_k])\, \lambda_{\max}(\mathbf{E}[\hat{z}_k])\right) \mathbf{E}[\langle R_k, R_k\rangle_F].\end{aligned}$$

Combine this with Theorem 3.5 and unroll the iteration to obtain the NS result. ∎

Remark 3.12. If $\hat{y}$ and $\hat{z}$ are rotationally invariant, the upper and lower probabilistic bounds in Corollary 3.11 coincide since $z_1 = z_n = \frac{s_1}{n}$ and $y_1 = y_m = \frac{s_2}{m}$. Algorithms 2.1 to 2.3 all use rotationally invariant distributions and converge predictably at the expected rate. The algorithms still converge with other distributions provided the smallest eigenvalue of the expectation is positive.

4. Numerical Results. Our sub-sampled algorithms Algorithms 2.1 to 2.3 are tested on a variety of SPD matrices: $A = XX^T$, $X \sim \mathcal{N}(0,1)^{n\times n}$; ridge regression matrices chosen from [2]; and matrices chosen from the Sparse Suite Library [3]. Algorithms 2.1 to 2.3 were implemented within the MATLAB code framework in [5] and we test on the same problems from [2, 3]. All computational tests were performed on **Superior**, a high-performance computing infrastructure at Michigan Technological University.

The experiments are organized as follows: Subsection 4.1 compares our algorithms with $s = s_1 = s_2 = \lceil\sqrt{n}\rceil$ (the sample size used in [5]) on one moderate sized $n \approx 5000$ matrix from each of the three classes tested in [5]; Subsection 4.2 demonstrates the independence of the convergence on the sample size $s \ll n$ for the same three matrices; the convergence of our algorithms on the remaining matrices from [5] are available as a supplementary document.

4.1. Convergence Test. The convergence,

$$\frac{\|A - B_k\|_F}{\|A\|_F},$$

of sampled algorithms [5] with sample size $s = \lceil\sqrt{n}\rceil$ are compared to our sub-sampled algorithms with $s_1 = s_2 = s$ on three matrices: ($n = 5000$) XX^T with $X \sim \mathcal{N}(0,1)^{n\times n}$ Figure 1; ($n = 5000$) Gisette-Scale [2] Figure 2; and ($n = 4704$) NASA [3] Figure 3. These figures show: BFGS(⋄) as specified by Eq. (2.5); DFP (⋄) as specified by Eq. (2.4); NS (⊗) as specified by Algorithm 2.1; SS1 (•) as specified by Algorithm 2.2; SS2 (■) as specified by Algorithm 2.3. Theoretical convergence rates from Eq. (3.15) are shown in dotted lines. Runs were terminated after $5n^2$ iterations or when the relative residual norm fell below 10^{-2}. Algorithms 2.1 to 2.3 converge predictably: linear in the semilog plots matching the theoretical convergence rates

(dotted lines). DFP and BFGS have target dependent weight matrices which may initially improve convergence. For the Gisette-Scale matrix Figure 2 DFP and BFGS show dramatic improvement. However, Figure 3 and various examples from [5] in the supplementary materials show that BFGS can fail to converge.

Since we sample U_k and V_k from rotationally invariant distributions, all our experiments show the predictable optimal convergence rates from Eq. (3.15) (dotted lines). With these choices the expected convergence rate of both NS and SS1 is $1-\left(\frac{s}{n}\right)^2$ while the expected convergence rate of SS2 is $\left(1-\left(\frac{s}{n}\right)^2\right)^2 = 1-2(s/n)^2+(s/n)^4$.

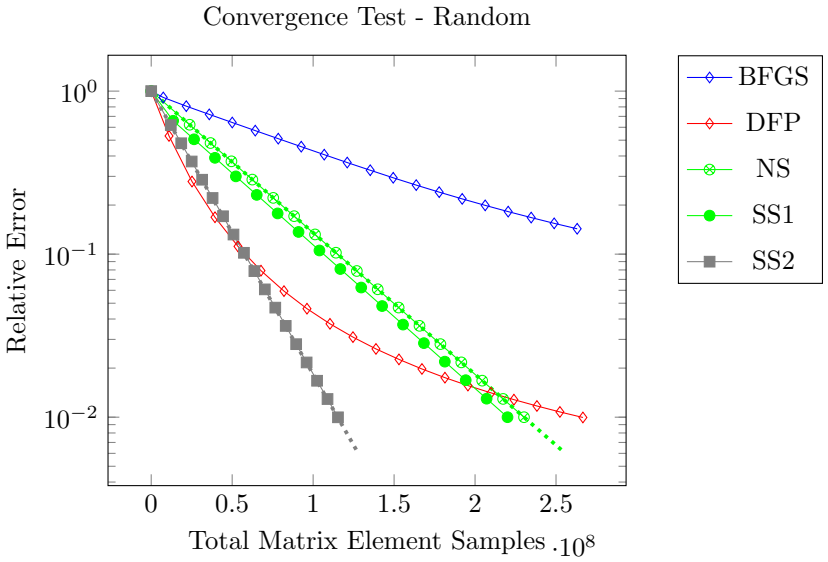

FIG. 1. *Approximating XX^T, with $X \sim \mathcal{N}(0,1)^{5000\times 5000}$ with $s = 71 = \lceil\sqrt{5000}\rceil$. Dotted lines indicate sub-sampled theoretical convergence rates.*

4.2. Sample Size Tests. Equation (3.15) gives the expected convergence rate, ρ, of the various algorithms as a function of the ratio of sample size s and matrix dimension n. Consider two experiments running SS1 with rotationally invariant sampling on the same $A \in \mathbb{R}^n$ with sample size s and $2s$: the first experiment involves s^2 matrix samples at each step, and one expects the residual to be reduced by a factor of $1-\left(\frac{s}{n}\right)^2$ after each step; the second experiment involves $(2s)^2$ matrix samples each step, and one expects the residual to be reduced by a factor of of $1-\left(\frac{2s}{n}\right)^2$ after each step. Since our primary cost metric for our algorithms is the number of matrix samples, four steps of size s^2 is the same amount of work as one step of size $(2s)^2$. Taking four steps of size s^2 gives approximately the same reduction as one step of size $(2s)^2$ since

$$\left(1-\left(\frac{s}{n}\right)^2\right)^4 = \left(1-\left(\frac{2s}{n}\right)^2\right)^1 + O\left(\left(\frac{s}{n}\right)^4\right).$$

All formulas in Eq. (3.15) have the same scaling behavior and as a result the expected convergence of all the sub-sampled algorithms should be essentially independent of

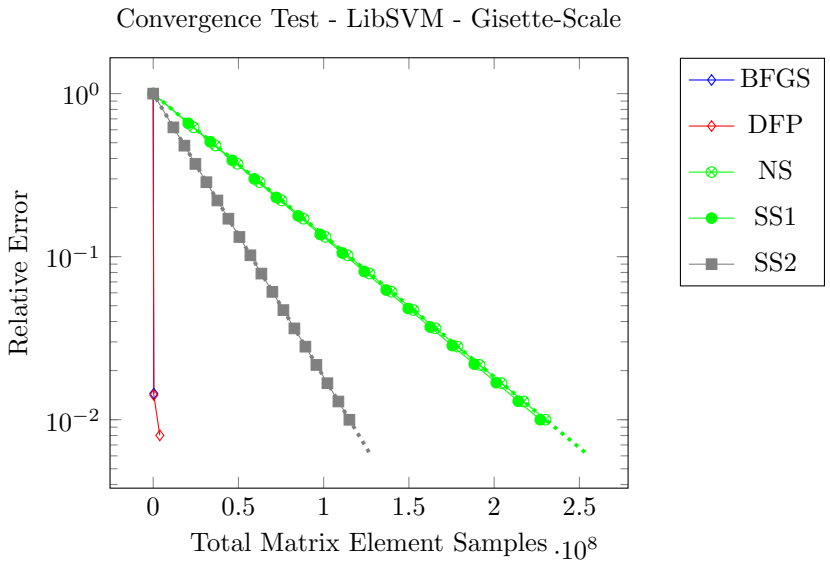

FIG. 2. *Hessian approximation for* ***Gisette Scale*** *($n = 5000$) from [2] with $s = 71 = \lceil\sqrt{5000}\rceil$. Dotted lines are theoretical convergence rates. Note, DFP and BFGS perform well.*

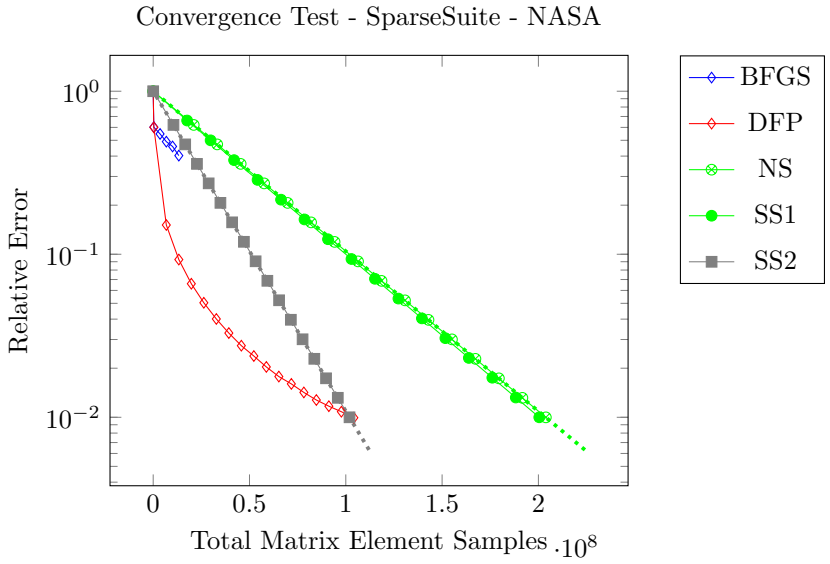

FIG. 3. *Approximating* ***NASA4704*** *from [3]. $s = 69 = \lceil\sqrt{4704}\rceil$. Dotted lines indicate sub-sampled theoretical convergence rates. Note: BFGS does not converge.*

s for $1 \ll s \ll n$. In practice, we would advocate choosing s to suit the available computational hardware.

This behavior is verified for the sub-sampled algorithms Algorithms 2.1 to 2.3 on the three test problems from subsection 4.1. In Table 1, we report the total computational effort for each matrix, normalized by the corresponding number of matrix samples for $s = 512$. All of the entries are very close to one, indicating that

the computational effort is independent of s.

Matrix	s	NS	SS1	SS2
Rand	128	0.997	0.992	0.999
	256	0.996	0.996	1.004
	512	1.000	1.000	1.000
Gisette Scale	128	0.996	0.990	0.995
	256	0.996	0.993	0.998
	512	1.000	1.000	1.000
NASA4704	128	0.996	0.998	0.994
	256	0.996	1.002	0.998
	512	1.000	1.000	1.000

TABLE 1

Computational effort relative to $s = 512$ for $s = 512, 256, 128$ for: ***Rand*** XX^T *($n = 5000$), with $X \sim \mathcal{N}(0,1)^{n\times n}$;* ***Gisette Scale*** *($n = 5000$) Hessian [2]; and* ***NASA4704*** *($n = 4704$) [3].*

5. Eigenvector Acceleration. The update underlying Algorithm 2.2 samples and then corrects the sample mismatch in the residual $R_k = A - B_k$. Larger corrections (and consequently more significant improvements in the approximation B_{k+1}) occur if $U^T R_k U$ is large. Block-power iteration on R_k is a simple heuristic to enhance subspaces associated with the larger eigenvalues of R_k. Algorithm 5.1 summarizes an extension to Algorithm 2.2 by incorporating a fixed number, p, of inner block-power iterations. As before, work estimates are boxed on the right (p steps of a block power iteration involving $p\,n\,s$ matrix samples and a square symmetric sample involving s^2 matrix samples) at each step with the total double boxed on the result line. This is not a sub-sampled algorithm (each internal power iteration involves a sample) and involves significantly more matrix samples per iteration. Despite this Algorithm 5.1 is competitive for small values of p.

Algorithm 5.1 SS1A: Accelerated Symmetric Approximation

Require: $B_0 \in \mathbb{R}^{n\times n}$ satisfying $B_0^T = B_0$, SPD $W \in \mathbb{R}^{n\times n}$, $s \in \mathbb{N}$.

repeat $\{k = 0, 1, \ldots\}$
  Sample $U_{0,k} \sim \mathcal{N}(0,1)^{n\times s}$
  $B_{0,k} = B_k$
  loop $\{i = 1, 2, \ldots, p\}$
    $\Lambda = AU_{i-1,k} - B_{i-1,k}U_{i-1,k}$
    $\Sigma = \Lambda(U_{i-1,k}^T W U_{i-1,k})^{-1} U_{i-1,k}^T W$
    $B_{i,k} = B_{i-1,k} + \Sigma + \Sigma^T - WU_{i-1,k}(U_{i-1,k}^T W U_{i-1,k})^{-1} U_{i-1,k}^T \Sigma$
    $U_{i,k} = \Lambda$
  end loop ..... $\boxed{p\,n\,s}$
  Compute residual $\Lambda_k = U_{p,k}^T A U_{p,k} - U_{p,k}^T B_{p,k} U_{m,k} \in \mathbb{R}^{s\times s}$ ..... $\boxed{s^2}$
  Compute $\tilde{P}_k = W\,U_{p,k}(U_{p,k}^T W\,U_{p,k})^{-1}$
  Update $B_{k+1} = B_k + \tilde{P}_k \Lambda_k \tilde{P}_k^T$
until convergence
return B_{k+1} ..... $\boxed{\boxed{(k+1)(p\,n\,s + s^2)}}$

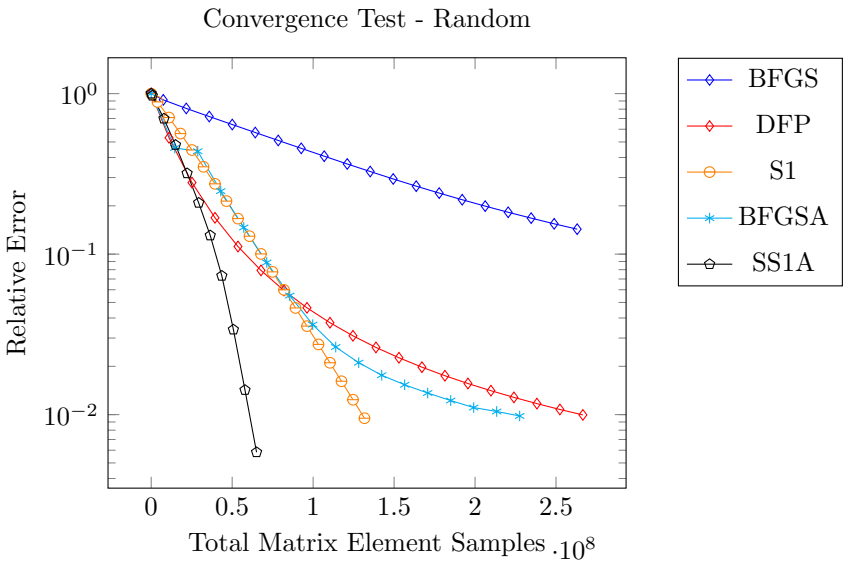

FIG. 4. *($n = 5000$) Approximation of XX^T where $X \sim \mathcal{N}(0,1)^{n\times n}$ with $s = 71$*

Remark 5.1. Implementing similar acceleration for Algorithm 2.3 would target the input/output spaces of the interior non-symmetric updates. Since, R_k is symmetric little acceleration is realized unless the input and output spaces match as in Algorithm 5.1.

Acceleration Convergence Results. We now compare the performance of SS1A Algorithm 5.1 (with rotationally invariant sampling and $p = 2$) to various algorithms: S1, BFGS, DFP, and a re-interpretation of the heuristic accelerated BFGS algorithm from [5] which we term BFGSA. Specifically, BFGSA is obtained by applying the Shermann-Morrison-Woodbury formula to the the adaptively sampled algorithm AdaRBFGS in [5], which approximates A^{-1}. The sampled algorithm, S1, is the B formulation in Eq. (2.3) with rotationally invariant weight $W = I_n$.

The convergence (relative Frobenius residual $\|A - B_k\|_F/\|A\|_F$ against matrix samples) of accelerated algorithms with sample size $s = \lceil\sqrt{n}\rceil$ from [5] are compared to our accelerated algorithm with $s_1 = s_2 = s$ on the three matrices from section 4: ($n = 5000$) XX^T with $X \sim \mathcal{N}(0,1)^{n\times n}$ Figure 4; ($n = 5000$) Gisette-Scale [2] Figure 5; and ($n = 4704$) NASA [3] Figure 6. These figures show: BFGSA ($*$) as specified by Eq. (2.5) with adaptive sampling described in [5]; S1 ($\circ$) as specified by Eq. (2.3); SS1A ($\pentagon$) as specified by Algorithm 5.1; BFGS ($\diamond$) as specified by Eq. (2.5); DFP ($\diamond$) as specified by Eq. (2.4). Runs were terminated after $5n^2$ iterations or when the relative residual norm fell below 10^{-2}. The results show SS1A matching or outperforming the other algorithms for the three matrices from subsection 4.1. Further accelerated experiments are discussed in the supplementary documents.

6. Conclusions. Sub-sampled methods have numerous advantages over current randomized block quasi-newton schemes, most notably, a smaller data footprint. Non-accelerated sub-sampled algorithms, Algorithms 2.1 to 2.3, have provable expected convergence rates which are independent of samples size. The non-accelerated algorithms realize these convergence rates in practice with extremely predictable convergence. Section 5 provides an easily interpreted heuristic acceleration which is

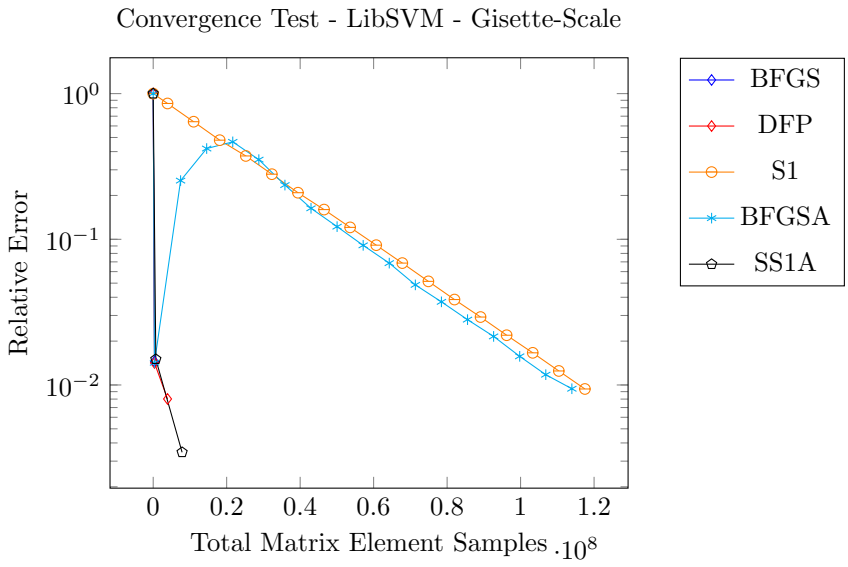

FIG. 5. *($n = 5000$) Hessian approximation for* ***Gisette Scale*** *[2]* $s = 71$.

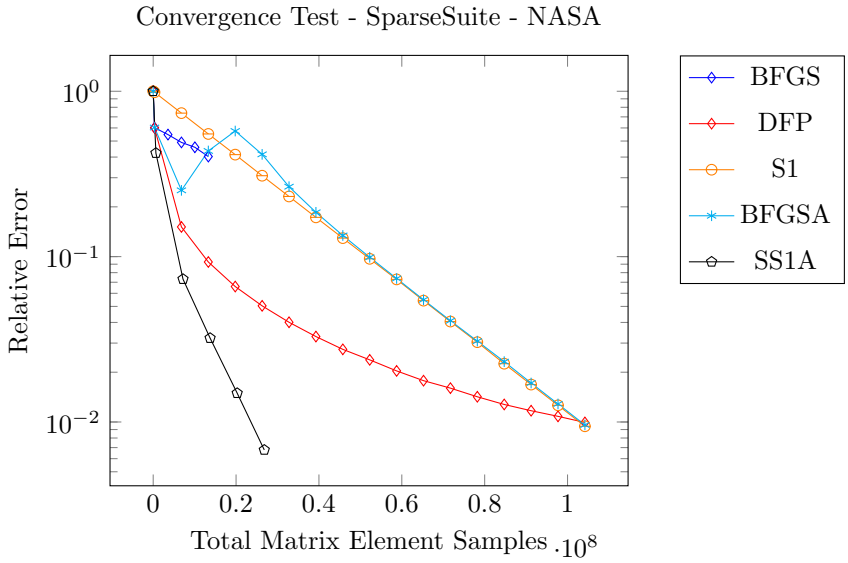

FIG. 6. *($n = 4700$) Approximation* ***NASA4704*** *from [3]* $s = 69$.

competitive with other accelerated matrix approximation methods while maintaining a small data footprint.

Appendix A. Weight Matrix Interpretation. The fixed non-rotationally symmetric weight matrices on which classical sampled methods are based (BFGS $W = A$ and DFP $= W^{-1}$) produce an enhanced initial drop in the appropriate residuals. Implementing sub-sampled algorithms Algorithms 2.1 to 2.3 with $W = A$ produces the same temporary effect. However, the enhancement is transitory and such

weighted algorithms ultimately converge at the rates in Theorems 3.5, 3.7, and 3.8 as B_k resolves A. This is to be expected since the algorithms sample and correct the residual $A - B_k$. Weights tuned to A become irrelevant as $B_k \to A$. The heuristic underlying the accelerated algorithm, Algorithm 5.1, is that non-constant weighting based on the residual $W_k = A - B_k$ should sample directions that are not yet well resolved: as noted in the discussion of Algorithm 5.1 such dynamic weighting requires samples $A\,U$.

Appendix B. Minimum Change Solutions. The KKT equations [9] for constrained minimum change formulations (2.1) and (2.2) are solved analytically using a change of variables. Substitute

$$\hat{A} = W_1^{-1/2}\,A\,W_2^{-1/2}, \quad \hat{B} = W_1^{-1/2}\,B\,W_2^{-1/2}, \quad \hat{B}_k = W_1^{-1/2}\,B_k\,W_2^{-1/2},$$

and

$$\hat{U} = W_1^{1/2}\,U, \quad \hat{V} = W_2^{1/2}\,V,$$

into Eq. (2.1) to get the unweighted problem,

$$\hat{B}_{k+1} = \arg\min_{\hat{B}} \left\{ \frac{1}{2}\|\hat{B} - \hat{B}_k\|_F^2 : \hat{U}^T\hat{B}\,\hat{V} = \hat{U}^T\hat{A}\,\hat{V} \right\}.$$

This reduces to

$$\arg\min_{E} \left\{ \frac{1}{2}\|E\|_F^2 : \hat{U}^T E\,\hat{V} - Z = 0 \right\},$$

where $E = \hat{B} - \hat{B}_k$ and $Z = \hat{U}^T(\hat{A} - \hat{B}_k)\hat{V}$. Writing Λ for the matrix of Lagrange multipliers, the Lagrangian is

$$\mathcal{L}(E, \Lambda) = \frac{1}{2}\,\mathrm{Tr}[E^T E] + \mathrm{Tr}[\Lambda^T(\hat{U}^T E\,\hat{V} - Z)].$$

Setting the derivative of $\mathcal{L}(E, \Lambda)$ with respect to the matrix argument E to 0 gives the Lagrange condition

$$\frac{\partial \mathcal{L}}{\partial E} = \frac{1}{2}\,\mathrm{Tr}[dE^T E + E^T dE] + \mathrm{Tr}[\hat{V}\Lambda^T\hat{U}^T dE] = 0,$$

which simplifies to

$$0 = E + \hat{U}\Lambda\hat{V}^T.$$

Substituting into the constraint equation gives

$$\hat{U}^T(\hat{U}\Lambda\hat{V}^T)\hat{V} + Z = 0,$$

which gives the multiplier matrix

$$\Lambda = -(\hat{U}^T\hat{U})^{-1}\hat{U}^T(\hat{A} - \hat{B}_k)\hat{V}(\hat{V}^T\hat{V})^{-1}.$$

Substituting and converting back to the original variables gives Eq. (2.7). The arguments for Eq. (2.2) are similar.

REFERENCES

[1] R. Bryd, R. Schnabel, and G. Schulz, Parallel quasi-newton methods for unconstrained optimization, Mathematical Programming, 42 (1988), pp. 273–306, https://doi.org/https://doi.org/10.1007?BF01589407.

[2] C.-C. Chang and C.-J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology, 2 (2011), pp. 27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[3] T. A. Davis and Y. Hu, The university of florida sparse matrix collection, ACM Trans. Math. Softw., 38 (2011), pp. 1:1–1:25, https://doi.org/10.1145/2049662.2049663, http://doi.acm.org/10.1145/2049662.2049663.

[4] W. Gao and D. Goldfarb, Block BFGS Methods, SIAM J. Optim., 28 (2018), pp. 1205–1231, https://doi.org/10.1137/16M1092106, https://doi.org/10.1137/16M1092106.

[5] R. M. Gower and P. Richtárik, Randomized quasi-Newton updates are linearly convergent matrix inversion algorithms, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 1380–1409, https://doi.org/10.1137/16M1062053, https://doi.org/10.1137/16M1062053.

[6] N. Halko, P. Martinsson, and J. Tropp, Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions, SIAM Review, 53 (2011), pp. 217–288, https://doi.org/10.1137/090771806, https://doi.org/10.1137/090771806, https://arxiv.org/abs/https://doi.org/10.1137/090771806.

[7] R. M. Gower, F. Hanzely, P. Richtárik, and S. Stich, Accelerated stochastic matrix inversion: General theory and speeding up bfgs rules for faster second-order optimization, PrePrint, (2018).

[8] U. Naumann, The Art of Differentiating Computer Programs, Society for Industrial and Applied Mathematics, 2011, https://doi.org/10.1137/1.9781611972078, https://epubs.siam.org/doi/abs/10.1137/1.9781611972078, https://arxiv.org/abs/https://epubs.siam.org/doi/pdf/10.1137/1.9781611972078.

[9] J. Nocedal and S. J. Wright, Numerical optimization, Springer Series in Operations Research and Financial Engineering, Springer, New York, second ed., 2006.

[10] R. Schnabel, Quasi-newton methods using multiple secant equations, Computer Science Technical Reports, 244 (1983), p. 41, https://doi.org/https://scholar.colorado.edu/csci_techreports/244/.

[11] G. Stewart, The efficient generation of random orthogonal matrices with an application to condition estimators, SIAM Journal on Numerical Analysis, 17 (1980), pp. 403–409, https://doi.org/10.1137/0717034, https://doi.org/10.1137/0717034, https://arxiv.org/abs/https://doi.org/10.1137/0717034.