Advances in Computational Mathematics manuscript No.
(will be inserted by the editor)

Randomized Iterative Methods for Matrix Approximation

Joy Azzam · Benjamin W. Ong · Allan A. Struthers

Received: date / Accepted: date

Abstract In this manuscript, novel methods are developed to iteratively embed aggregate pieces of information from a data matrix to generate an approximate data matrix. These algorithms form a potential framework for enabling discovery from data while protecting the individual data elements. The developed sub-sampled randomized algorithms converge with provable error bounds. A heuristic accelerated scheme is also developed, motivated by the sub-sample analysis. We compare our algorithms to current sampled algorithms on a substantial test-suite of matrices and verify that the theoretical convergence rates are numerically realized.

Keywords Matrix approximation · Randomized algorithms · Sub-sampled · Quasi-Newton

Mathematics Subject Classification (2010) 68W20 · 68W2 · , 65F35 · 90C53

1 Introduction

In a wide range of applications, data is represented by a real $m \times n$ matrix A. An emerging concern is data privacy, where one wants to share information about the data while withholding information about specific entries. There are various popular approaches to share this information, most notable of which is

Joy Azzam
Michigan Technological University
Department of Mathematical Sciences
E-mail: atazzamw@mtu.edu Benjamin W. Ong
Michigan Technological University
Department of Mathematical Sciences
E-mail: ongbw@mtu.edu Allan A. Struthers
Michigan Technological University
Department of Mathematical Sciences
E-mail: struther@mtu.edu

differential privacy [6] which is used by Google [17] and other tech companies that need to comply with privacy regulations, and more recently, will be used by the US Census in 2020 [1].

In this paper, we introduce a framework for *iteratively* embedding aggregate pieces of information from a data matrix to generate an approximate data matrix. The aggregate pieces of information from the data matrix is of size $s_1 \times s_2$, and is obtained by computing the product,

$$U^T A V \in \mathbb{R}^{s_1 \times s_2}. \tag{1}$$

where $U \in \mathbb{R}^{m \times s_1}$ and $V \in \mathcal{R}^{n \times s_2}$. The product, eq. (1) can be viewed as weighted linear combinations of the rows and columns of the data. We will show that we can extract useful information about the entire data matrix using only aggregate pieces of information from the data matrix.

Algorithms that utilize eq. (1) fall into an active research area known as randomized numerical linear algebra. Indeed, there is an extensive list of articles citing a comprehensive review article [9], providing a wide range of applications and algorithms. The algorithms presented in this paper differ in spirit from existing algorithms in the review article [9,2]. Existing algorithms expend significant computational effort to compute matrices U and V so that $U^T A V$ approximates the action of A associated with its dominant eigenspace. This is often viewed as preconditioning the data matrix (once) so that uniform random sampling of the projected matrix yields an insignificant loss in approximation accuracy. In contrast, our algorithms draw many U's and V's from a Gaussian distribution, $U \sim \mathcal{N}(0,1)^{m \times s_1}$ and $V \sim \mathcal{N}(0,1)^{n \times s_2}$, and iteratively embed these aggregate pieces of information, eq. (1), to generate an approximate data matrix.

Existing earlier uses of two sided samples eq. (1) are similarly focused on non-iterative algorithms. Schatten p norms (pth root of the sum of the pth power of the singular values) estimators from subsamples eq. (1) and samples AV are compared (with cost estimates ms_2 for samples AV and $s_1 s_2$ for $U^T AV$ called bilinear sketches) in [10]. Large eigenvalues are estimated using two sided random projectors in [2]. In [3] a number of proofs use two-sided samples to tighten bounds on low rank approximations.

The algorithms that we propose have an additional benefit. Given aggregate pieces of information $U^T AV$, our algorithms only operate on these $s_1 \times s_2$ pieces of aggregate information and are thus computationally efficient and tunable for modern hardware architectures. In § 2, we highlight this computational cost in the algorithm specifications.

Before shifting our discussion to randomized quasi-Newton algorithms (derived from non-linear optimization) which motivated this current work, we settle on some notation that will be used throughout the manuscript.

1.1 Notation

SPD is an acronym for symmetric positive definite and W will denote SPD weight matrices. The superscript $^+$ denotes the Moore-Penrose pseudo-inverse; $\langle X, Y\rangle_F = \mathrm{Tr}\left[X^T Y\right]$ and $\|X\|_F^2 = \langle X, X\rangle_F$ denote the Frobenius inner product and norm. Residuals are measured using weighted norms. For non-symmetric matrices, weighted norms are denoted

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

and for symmetric matrices, weighted norms are denoted

$$\|X\|^2_{F(W^{-1})} = \|W^{-1/2} X\, W^{-1/2}\|_F^2,$$

with conforming SPD weights W_1, W_2 and W. Algorithms in this manuscript are developed using the W-weighted projector, which projects onto the column space of $W\,U$,

$$\mathcal{P} = P_{W^{-1},U} = W\,U(U^T\,W\,U)^{-1}U^T. \tag{2}$$

The weighted projector satisfies

$$\mathcal{P}\,W = W\,\mathcal{P}^T = \mathcal{P}\,W\,\mathcal{P}^T \quad \text{and} \quad W^{-1}\mathcal{P} = \mathcal{P}^T W^{-1} = \mathcal{P}^T\,W^{-1}\,\mathcal{P}. \tag{3}$$

1.2 Randomized Quasi-Newton Algorithms

Our goal is to develop and analyze iterative approximations to A which use sub-samples $U^T A\,V$. The resulting algorithms are strongly connected to and motivated by quasi-Newton algorithms from nonlinear optimization and sampled quasi-Newton algorithms [8], which we now review.

Quasi-Newton schemes for SPD matrices A generate either a sequence of approximations satisfying $B_k \to A$, or a sequence of approximations satisfying $H_k \to A^{-1}$, generated by applying the Sherman-Morrison-Woodbury (SMW) formula to B_k. The schemes are formulated using constrained minimum change criteria (for $B \approx A$ and $H \approx A^{-1}$) in weighted Frobenius norms [12]. Randomized (sampled) update algorithms are similarly derived [8,7]. The KKT equations for the quadratic programs,

$$B_{k+1} = \arg\min_B \left\{ \frac{1}{2}\|B - B_k\|^2_{F(W^{-1})} \mid B\,U = A\,U \text{ and } B = B^T \right\} \tag{4}$$

$$H_{k+1} = \arg\min_H \left\{ \frac{1}{2}\|H - H_k\|^2_{F(W^{-1})} \mid U = H\,A\,U \text{ and } H = H^T \right\} \tag{5}$$

give 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 update formula that results from solving the constrained minimum change criteria, eqs. (4) and (5), are

$$B_{k+1} = B_k + \mathcal{P}_B(A - B_k) + (A - B_k)\mathcal{P}_B^T - \mathcal{P}_B(A - B_k)\mathcal{P}_B^T, \tag{6}$$

$$H_{k+1} = H_k + \mathcal{P}_H(A^{-1} - H_k) + (A^{-1} - H_k)\mathcal{P}_H^T - \mathcal{P}_H(A^{-1} - H_k)\mathcal{P}_H^T, \tag{7}$$

where the weighted projectors $\mathcal{P}_B$ and $\mathcal{P}_H$ defined by eq. (2) are

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

Familiar algorithms can be obtained by selecting different weights, W. Block DFP [15] is the B formulation, eq. (4), with $W = A$. The corresponding update formula is

$$B_{k+1} = (I_n - \mathcal{P}_{\text{DFP}})\; B_k(I_n - \mathcal{P}_{\text{DFP}}^T) + \mathcal{P}_{\text{DFP}}\,A, \tag{8}$$

where

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

Block BFGS [8,7] is the H formulation, eq. (5), (inverted using the Sherman-Morrison-Woodbury formula) with $W = A^{-1}$. The corresponding update formula is

$$B_{k+1} = B_k - B_kU\left(U^TB_kU\right)^{-1}U^TB_k + A\,U\left(U^TAU\right)^{-1}U^T\,A. \tag{9}$$

Although the main goal of the discussed approximation (randomized) methods for quasi-Newton methods are the construction of approximate matrix inverses for use as preconditioners, matrix approximation underlies the heart of such algorithms, coupled with SMW formula. In this work, we develop a framework and theory for matrix approximations using sub-sampled data. This will lay the foundation for future exploration of approximate matrix inverses.

1.3 Outline

The manuscript is organized into two main parts. In § 2, we introduce our randomized sub-sampled methods. Our methods are light-weight, self-correcting iterative updates for approximating matrices. We analyze these methods in § 3, providing convergence rates and error estimates. We then provide numerical evidence in § 4 demonstrating the effectiveness of our methods.

The second part of our manuscript uses our sub-sampled ideas to accelerate *sampled* algorithms heuristically. Specifically, § 5.1 develops block power iteration accelerated sub-sampled algorithms and numerically compares them to equivalent sampled based algorithms with similar heuristics.

2 Randomized Approximation Methods

We introduce three algorithms in this section. The first algorithm is presented in § 2.1. This algorithm is able to approximate non-square matrices, and arises from a sub-sampled analog to the sampled algorithm in § 1.2. If the input matrix A is symmetric, two additional algorithms are proposed in §§ 2.2 and 2.3 to maintain symmetry of the matrix approximations. The algorithms in this section and the analysis in § 3 are formulated with general SPD weight matrices, W, W_1 and W_2.

2.1 General Sub-Sampled Update

Using an analog to eq. (4), we seek to satisfy the minimal change criterion,

$$B_{k+1} = \arg\min_{B} \left\{ \frac{1}{2} \|B - B_k\|^2_{F(W_1^{-1}, W_2^{-1})} \mid U^T B\, V = U^T A\, V \right\}. \tag{10}$$

Solving eq. (10) gives rise to a self-correcting update (for details see appendix B)

$$B_{k+1} = B_k + P_{W_1^{-1}, U^k}(A - B_k) P^T_{W_2^{-1}, V^k}. \tag{11}$$

By construction, eq. (11) 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. (11) generates a sequence $\{B_k\}$ that converges to A monotonically in the appropriate weighted Frobenius norm. The resulting algorithm is summarized in algorithm 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.

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 1: NS: Non-Symmetric Sub-Sampled Approximation

Algorithm 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 1 with symmetric initialization $B_0 = B_0^T$ gives a sequence of symmetric ap-

proximations, B_k, to a symmetric $n \times n$ matrix A. The resulting algorithm is summarized in algorithm 2 with sample counts boxed as before.

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} ..... $\boxed{\boxed{(k+1)\left(s_1^2\right)}}$

Algorithm 2: SS1: Symmetric Sub-Sampled Approximation

Remark 1 Algorithm 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. (9) to get algorithm 2 with $W = I_n$. Algorithm 2 can be viewed as a sub-sampled DFP update.

Remark 2 Algorithm 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. (11) as follows

$$\begin{aligned} B_{k+1/2} &= B_k + P_{W^{-1},U^k}(A - B_k)P_{W^{-1},V^k}^T \\ B_{k+1} &= \frac{1}{2}\left(B_{k+1/2} + B_{k+1/2}^T\right). \end{aligned} \tag{12}$$

For symmetric A and B_0, it can be shown that the convergence rate for eq. (12) is comparable to Algorithm 2. However, for symmetric A the additional sample,

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

can be directly incorporated to give

$$\begin{aligned} B_{k+1/3} &= B_k + P_{W_1^{-1},U^k}(A - B_k)P_{W_2^{-1},V^k}^T \\ B_{k+2/3} &= B_{k+1/3} + P_{W_2^{-1},V^k}(A - B_{k+1/3}^T)P_{W_1^{-1},U^k}^T \\ B_{k+1} &= \frac{1}{2}\left(B_{k+2/3} + B_{k+2/3}^T\right), \end{aligned} \tag{13}$$

where the last line again enforces symmetry. We summarize this two-step symmetric algorithm in algorithm 3 with sample counts boxed as before. This two-step algorithm has superior convergence properties.

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 $B_{k+1} = \frac{1}{2}(B_{k+2/3} + B_{k+2/3}^T)$
8: **until** convergence
9: **return** B_{k+1} ..... $\boxed{\boxed{(k+1)\,(s_1 s_2)}}$

Algorithm 3: SS2: Two-Step Symmetric Sub-Sampled Approximation

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 [16]. Our algorithms use symmetric weighted rank s projectors,

$$\hat{z} = W^{1/2} U (U^T W U)^{-1} U^T W^{1/2}, \tag{14}$$

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 $z_1 \le z_2 \le \cdots \le z_n$. 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 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 1 (Random Projections) *For any distribution $\hat{z}$ of real, symmetric rank s projectors in $\mathbb{R}^n$,*

$$0 \le \lambda_{\min}(\boldsymbol{E}[\hat{z}]) \le \frac{s}{n} \le \lambda_{\max}(\boldsymbol{E}[\hat{z}]) \le 1. \tag{15}$$

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. (15). 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 2 (Projection Cancellation) *For $R \in \mathbb{R}^{m \times n}$ and conforming symmetric projections $\hat{y}, \hat{z}$,*

$$\langle R\, \hat{z}, R\, \hat{z} \rangle_F = \langle R, R\, \hat{z} \rangle_F \tag{16}$$

$$\langle \hat{y}\, R\, \hat{z}, \hat{y}\, R\, \hat{z} \rangle_F = \langle \hat{y}\, R\, \hat{z}, R\, \hat{z} \rangle_F = \langle \hat{y}\, R\, \hat{z}, R \rangle_F \tag{17}$$

Proof Expanding the definition of eq. (16),

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

since $\mathrm{Tr}[A\, B] = \mathrm{Tr}[B\, A]$ and $\hat{z}$ is a projector. Similarly for eq. (17),

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

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

Lemma 3 (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:*

$$\lambda_{\min}(S_1) \langle R, R \rangle_F \leq \langle S_1\, R, R \rangle_F \leq \lambda_{\max}(S_1) \langle R, R \rangle_F, \tag{18}$$

$$\lambda_{\min}(S_2) \langle R, R \rangle_F \leq \langle R, R\, S_2 \rangle_F \leq \lambda_{\max}(S_2) \langle R, R \rangle_F, \tag{19}$$

$$\lambda_{\min}(S)^2 \langle R, R \rangle_F \leq \langle S\, R, R\, S \rangle_F \leq \lambda_{\max}(S)^2 \langle R, R \rangle_F. \tag{20}$$

Proof To establish eq. (18) 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

$$\sum_{i=1}^n \lambda_{\min}(S_1)\, r_i^T r_i \leq \sum_{i=1}^n r_i^T S_1 r_i \leq \sum_{i=1}^n \lambda_{\max}(S_1)\, r_i^T r_i. \tag{21}$$

Equation (19) follows directly from eq. (18) 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. (20) 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 (20) then follows immediately with $T = S^{1/2}$ from eq. (18) applied to $S_1 = T^2$ and the standard bound eq. (21) with $S_1 = T^2$.

3.2 Convergence Theorems

Convergence results for algorithms 1 to 3. are for $\mathbf{E}[\|B-A\|_F^2]$. Such results dominate similar results for $\|\mathbf{E}[B-A]\|_F^2$ since

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

as shown in [8].

Theorem 1 (Convergence of NS algorithm 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.* (11) *generates a sequence, B_k, from an initial guess $B_0 \in \mathbb{R}^{m\times n}$ satisfying*

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

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

$$\hat{y}_k = W_1^{1/2}U_k(U_k^TW_1U_k)^{-1}U_k^TW_1^{1/2}, \quad \hat{z}_k = W_2^{1/2}V_k(V_k^TW_2V_k)^{-1}V_k^TW_2^{1/2}. \tag{22}$$

Proof Define the kth residual as $R_k := W_1^{-1/2}(B_k-A)W_2^{-1/2}$. With some algebraic manipulation, eq. (11) can be re-written as

$$R_{k+1} = R_k - \hat{y}_kR_k\hat{z}_k. \tag{23}$$

Computing the squared Frobenius norm of eq. (23),

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

where we have made use of lemma 2. Taking the expected value with respect to independent samples U_k (leaving V_k and R_k fixed) gives

$$\begin{aligned}\mathbf{E}\left[\|R_{k+1}\|_F^2 \mid V_k,R_k\right] &= \langle R_k,R_k\rangle_F - \langle \mathbf{E}[\hat{y}_k]R_k\hat{z}_k,R_k\hat{z}_k\rangle_F \\ &\leq \langle R_k,R_k\rangle_F - \lambda_{\min}(\mathbf{E}[\hat{y}_k])\langle R_k\hat{z}_k,R_k\hat{z}_k\rangle_F \\ &\leq \langle R_k,R_k\rangle_F - \lambda_{\min}(\mathbf{E}[\hat{y}_k])\langle R_k,R_k\hat{z}_k\rangle_F,\end{aligned} \tag{24}$$

where we applied lemma 3 to the symmetric positive semi-definite matrix $\mathbf{E}[\hat{y}_k]$, and utilized eq. (16). 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 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 2 (Convergence of SS1 algorithm 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, the iterates B_{k+1} satisfy*

$$\boldsymbol{E}[\|B_{k+1} - A\|^2_{F(W^{-1})}] \leq (\rho_{SS1})^k \boldsymbol{E}[\|B_0 - A\|^2_{F(W^{-1})}], \tag{25}$$

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 1, 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. (20),

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

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

Theorem 3 (Convergence of SS2 algorithm 3) *Let A, U_k, V_k and B_0 be defined as in theorem 1, and let W be a fixed SPD matrix. After applying k iterations of algorithm 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 1 with $W = W_1 = W_2$. The iteration given in eq. (13) 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 1 gives

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

and a repeated application of theorem 1 gives

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

Lastly, we observe via the triangle inequality that

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

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

3.3 Optimal Fixed Weight Convergence Rates

To discuss convergence rates, we define

$$\begin{aligned} \rho_{\mathrm{NS}}(y_1, z_1) &= 1 - y_1 z_1, \\ \rho_{\mathrm{SS1}}(z_1) &= 1 - z_1^2, \\ \rho_{\mathrm{SS2}}(y_1, z_1) &= (1 - y_1 z_1)^2, \end{aligned} \tag{26}$$

and note that the convergence rates for algorithms 1 to 3 can be expressed as

$$\|R_{k+1}\|^2_{F(W_1^{-1},W_2^{-1})} \leq \rho \|R_k\|^2_{F(W_1^{-1},W_2^{-1})} \tag{27}$$

with the appropriate ρ, eq. (26), 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 1 *(Optimal Convergence Rate) The optimal convergence rates for algorithms 1 to 3 are obtained attained for U_k and V_k sampled from rotationally invariant distributions,*

$$\begin{aligned} \rho_{NS}^{opt} &= 1 - \frac{s_1}{m}\frac{s_2}{n}, \\ \rho_{SS1}^{opt} &= 1 - \left(\frac{s_2}{n}\right)^2, \\ \rho_{SS2}^{opt} &= \left(1 - \frac{s_1}{m}\frac{s_2}{n}\right)^2. \end{aligned} \tag{28}$$

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

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

Remark 4 Theorems 1 to 3 all assume the weight matrix W and distributions are fixed. All our non-accelerated numerical experiments use fixed weights and sample from fixed rotationally invariant distributions.

Remark 5 Corollary 1 is an extremely strong result. Consider for simplicity $s_1 = s_2 = s$. Although the convergence rates are roughly $1 - \left(\frac{s}{n}\right)^2$, only $s \times s$ aggregated pieces of information are used each iteration. If a sampled algorithm uses $s \times n$ pieces of information, e.g. [8], our algorithm can take $\frac{n}{s}$ iterations with the *same amount of information/work*. Consequently the error decrease after $\frac{n}{s}$ satisfies

$$\left(1 - \frac{s^2}{n^2}\right)^{n/s} \approx 1 - \frac{n}{s}\cdot\frac{s^2}{n^2},$$

which is comparable to the convergence rates of sampled quasi-Newton methods.

3.4 Theoretical Lower Bound for Convergence Rates

Lower bounds (entirely analogous to the upper bounds in theorems 1 to 3 but using the upper bounds in lemma 3) are easily derived. For example, the two-sided error bound for algorithm 1 is

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

where as before $y_1 \leq y_2 \leq \cdots \leq y_m$ is the spectrum of $\mathbf{E}[\hat{y}]$, $z_1 \leq z_2 \leq \cdots \leq z_n$ is the spectrum of $\mathbf{E}[\hat{z}]$ and the explicit form for ρ_{NS} is in eq. (26). We collect the similar results for algorithms 1 to 3 in corollary 2.

Corollary 2 (Two-Sided Convergence Rates) *Given the assumptions of theorems 1 to 3 the explicit formulas eq.* (26) *for ρ give two-sided bounds,*

$$\rho_{NS}(y_m, z_n)^k \leq \frac{\boldsymbol{E}\left[\|B_{k+1} - A\|^2_{F(W_1^{-1}, W_2^{-1})}\right]}{\|B_0 - A\|^2_{F(W_1^{-1}, W_2^{-1})}} \leq \rho_{NS}(y_1, z_1)^k$$

$$\rho_{SS1}(z_n)^k \leq \frac{\boldsymbol{E}\left[\|B_{k+1} - A\|^2_{F(W^{-1})}\right]}{\|B_0 - A\|^2_{F(W^{-1})}} \leq \rho_{SS1}(z_1)^k$$

$$\rho_{SS2}(y_n, z_n)^k \leq \frac{\boldsymbol{E}\left[\|B_{k+1} - A\|^2_{F(W^{-1})}\right]}{\|B_0 - A\|^2_{F(W^{-1})}} \leq \rho_{SS2}(y_1, z_1)^k$$

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 (24) of theorem 1 and lemma 3 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 1 (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

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

Combine this with theorem 1 and unroll the iteration to obtain the NS result.

Remark 6 If $\hat{y}$ and $\hat{z}$ are rotationally invariant, the upper and lower probabilistic bounds in corollary 2 coincide since $z_1 = z_n = \frac{s_1}{n}$ and $y_1 = y_m = \frac{s_2}{m}$. Algorithms 1 to 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 1 to 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 [4]; and matrices chosen from the Sparse Suite Library [5]. Algorithms 1 to 3 were implemented within the MATLAB code framework in [8] and we test on the same problems from [4,5]. All computational tests were performed on **Superior**, a high-performance computing infrastructure at Michigan Technological University.

We assume 'black-box' matrix access which efficiently computes products AV, U^TA and/or U^TAV 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. Such a scenario will arise for example, if the owner of the data seeks to enable discovery from the data, and is willing to operate on the data to release aggregate pieces of information about the data. Hence, the size of the aggregated pieces of information will be the primary cost metric for our algorithms.

Although the algorithms in § 2 were formulated with general SPD weight matrices, W, W_1 and W_2, the numerical experiments utilize $W = I$ so that a fair comparison can be made with sampled algorithms [8].

The experiments are organized as follows: § 4.1 compares our algorithms with $s = s_1 = s_2 = \lceil\sqrt{n}\rceil$ (the sample size used in [8]) on one moderate sized $n \approx 5000$ matrix from each of the three classes tested in [8]; § 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 [8] are available as a supplementary document.

4.1 Convergence Test

The convergence,

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

of sampled algorithms [8] 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 $X \sim \mathcal{N}(0,1)^{n\times n}$ § 4.1; ($n = 5000$) Gisette-Scale [4] § 4.1; and ($n = 4704$) NASA [5] § 4.1. These figures show: BFGS(⋄) as specified by eq. (9); DFP (⋄) as specified by eq. (8); NS (⊗) as specified by Algorithm 1; SS1 (•) as specified by Algorithm 2; SS2 (■) as specified by Algorithm 3. Theoretical convergence rates from eq. (28) are shown in dotted lines. Runs were terminated after $5n^2$ iterations or when the relative residual norm fell below 10^{-2}. Algorithms 1 to 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 § 4.1 DFP and BFGS show dramatic improvement. However, § 4.1 and various

examples from [8] 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. (28) (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 $\left(1-\left(\frac{s}{n}\right)^2\right)^2 = 1-2(s/n)^2+(s/n)^4$.

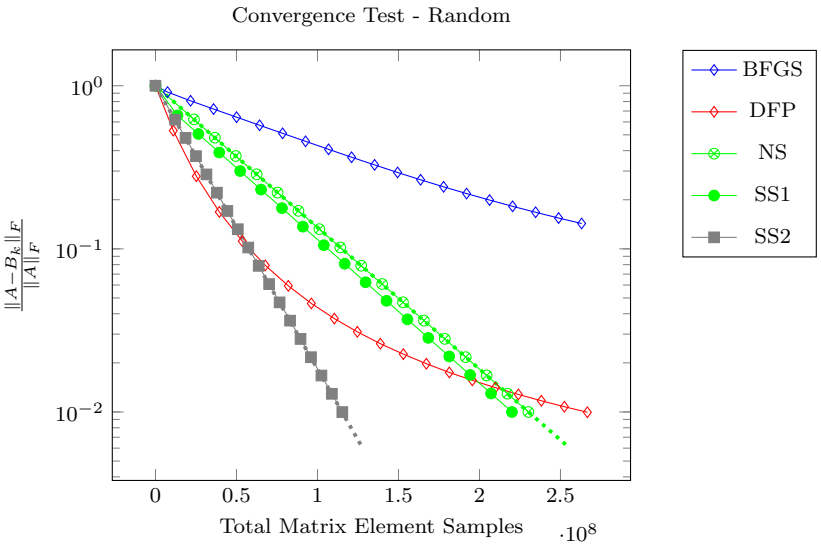

Fig. 1 ($n = 5000$) Approximation of XX^T where $X \sim \mathcal{N}(0,1)^{n\times n}$ with $s = 71 = \lceil\sqrt{5000}\rceil$. Dotted lines indicate sub-sampled theoretical convergence rates.

4.2 Sample Size Tests

Equation (28) 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 $A \in \mathbb{R}^{n\times n}$ 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

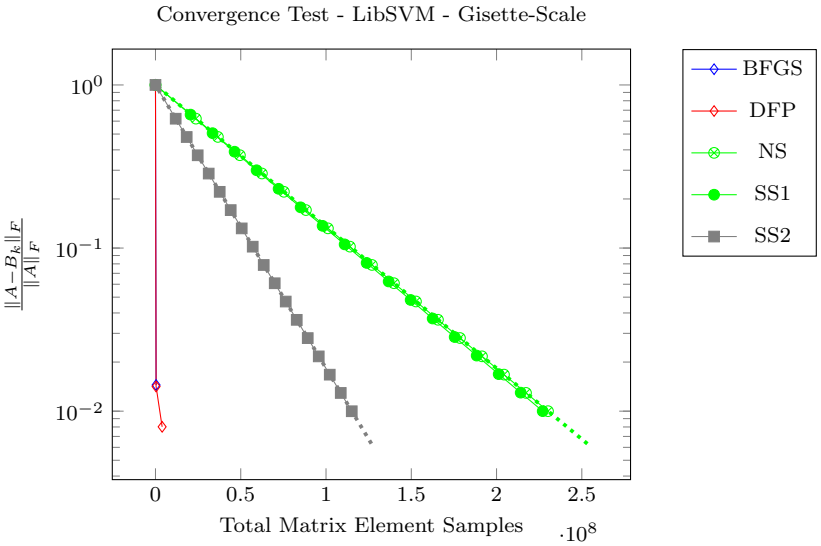

Fig. 2 ($n = 5000$) Approximation of Hessian from **Gisette Scale** [4] with $s = 71 = \lceil\sqrt{5000}\rceil$. Dotted lines are theoretical convergence rates. DFP and BFGS perform well.

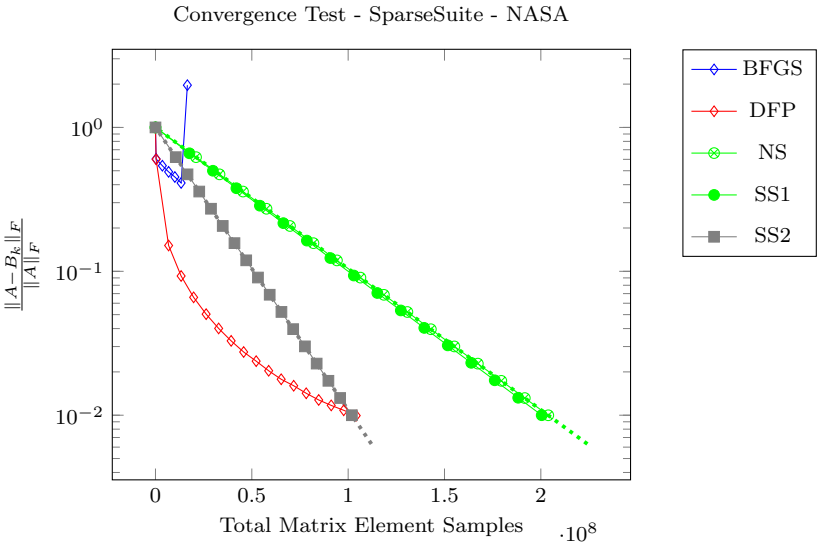

Fig. 3 ($n = 4700$) Approximation of **NASA4704** from [5]. $s = 69 = \lceil\sqrt{4704}\rceil$. Dotted lines indicate sub-sampled theoretical convergence rates. BFGS does not converge.

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. (28) have the same scaling behavior and as a result the expected convergence of all the sub-sampled algorithms should be essentially independent of 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 1 to 3 on the three test problems from § 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 [4]; and **NASA4704** ($n = 4704$) [5].

Remark 7 Before moving on to some heuristic accelerated schemes, it is perhaps helpful to place our work in context of other randomized quasi-Newton-like methods that use "sub-samples" in some fashion. Consider an optimization problem involving objective functions of the form

$$f(x) = \sum_{k=1}^{N} f_k(x), \quad x \in \mathbb{R}^D \tag{29}$$

A class of methods known as sub-sampled Newton methods [14,13] approximate Hessians computed from derivatives of eq. (29) by sampling the input vector x, and sampling components of the objective function, i.e.,

$$H \approx \nabla^2 \sum_{k=1}^{s_1} f_{\sigma(k)}(V\,x),$$

where $\sigma_k : \mathbb{Z}_N \to \mathbb{Z}_N$ and $V \in \mathbb{R}^{n\times n}$ has s non-zero columns. This is rightly termed as sub-sampled method in the sense that the input vector is sampled, and the objective function is also sampled. In some sense, both our

proposed methods and these sub-sampled Newton methods sample the input space. What differs is that our methods sample the output space, whereas sub-sampled Newton methods sample the objective function, two entirely different objects. Our proposed methods iterate on $s_1 \times s_2$ pieces of information, while the sub-sampled Newton methods essentially operate on $s_1 \times n$ pieces of information.

5 Heuristic Accelerated Schemes

Motivated by the sub-sampled analysis, we develop a heuristic accelerated scheme in § 5.1. Numerical convergence and acceleration is verified in § 5.2. Lastly, we make some observations about how our heuristic scheme is related to other accelerated sampled algorithms in § 5.3, and block Krylov iteration in § 5.4.

5.1 Eigenvector Acceleration

The update underlying algorithm 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 4 summarizes an extension to algorithm 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 4 is competitive for small values of p.

Require: $B_0 \in \mathbb{R}^{n\times n}$ satisfying $B_0^T = B_0$, SPD $W \in \mathbb{R}^{n\times n}$, $s \in \mathbb{N}$.
1: **repeat** $\{k = 0, 1, \ldots\}$
2: Sample $U_{0,k} \sim \mathcal{N}(0,1)^{n\times s}$
3: $B_{0,k} = B_k$
4: **loop** $\{i = 1, 2, \ldots, p\}$
5: $\Lambda = AU_{i-1,k} - B_{i-1,k}U_{i-1,k}$
6: $\Sigma = \Lambda(U_{i-1,k}^T W U_{i-1,k})^{-1} U_{i-1,k}^T W$
7: $B_{i,k} = B_{i-1,k} + \Sigma + \Sigma^T - WU_{i-1,k}(U_{i-1,k}^T W U_{i-1,k})^{-1} U_{i-1,k}^T \Sigma$
8: $U_{i,k} = \Lambda$
9: **end loop** ..... $\boxed{p\,n\,s}$
10: Compute residual $\Lambda_k = U_{p,k}^T A U_{p,k} - U_{p,k}^T B_{p,k} U_{m,k} \in \mathbb{R}^{s\times s}$ ..... $\boxed{s^2}$
11: Compute $\tilde{P}_k = W\,U_{p,k}(U_{p,k}^T W\,U_{p,k})^{-1}$
12: Update $B_{k+1} = B_k + \tilde{P}_k \Lambda_k \tilde{P}_k^T$
13: **until** convergence
14: **return** B_{k+1} ..... $\boxed{\boxed{(k+1)(p\,n\,s + s^2)}}$

Algorithm 4: SS1A: Accelerated Symmetric Approximation

Remark 8 Implementing similar acceleration for algorithm 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 4.

5.2 Acceleration Convergence Results

We now compare the performance of SS1A algorithm 4 (with rotationally invariant sampling and $p = 2$) to various algorithms: S1, BFGS, DFP, and a re-interpretation of the heuristic accelerated BFGS algorithm from [8] which we term BFGSA. Specifically, BFGSA is obtained by applying the Sherman-Morrison-Woodbury formula to the the adaptively sampled algorithm AdaRBFGS in [8], which approximates A^{-1}. The sampled algorithm, S1, is the B formulation in eq. (6) 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 [8] are compared to our accelerated algorithm with $s_1 = s_2 = s$ on the three matrices from § 4: ($n = 5000$) XX^T with $X \sim \mathcal{N}(0,1)^{n\times n}$ § 5.2; ($n = 5000$) Gisette-Scale [4] § 5.2; and ($n = 4704$) NASA [5] § 5.2. These figures show: BFGSA (∗) as specified by eq. (9) with adaptive sampling described in [8]; S1 (○) as specified by eq. (6); SS1A (⬠) as specified by Algorithm 4; BFGS (◇) as specified by eq. (9); DFP (◇) as specified by eq. (8). 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 §4.1. Further accelerated experiments are discussed in the supplementary documents.

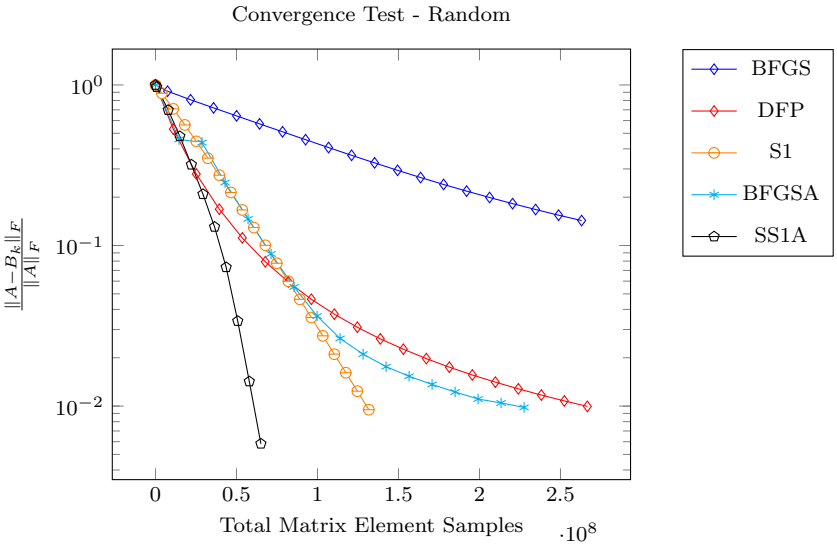

Fig. 4 ($n = 5000$) Approximation of XX^T where $X \sim \mathcal{N}(0,1)^{n\times n}$ with $s = 71$

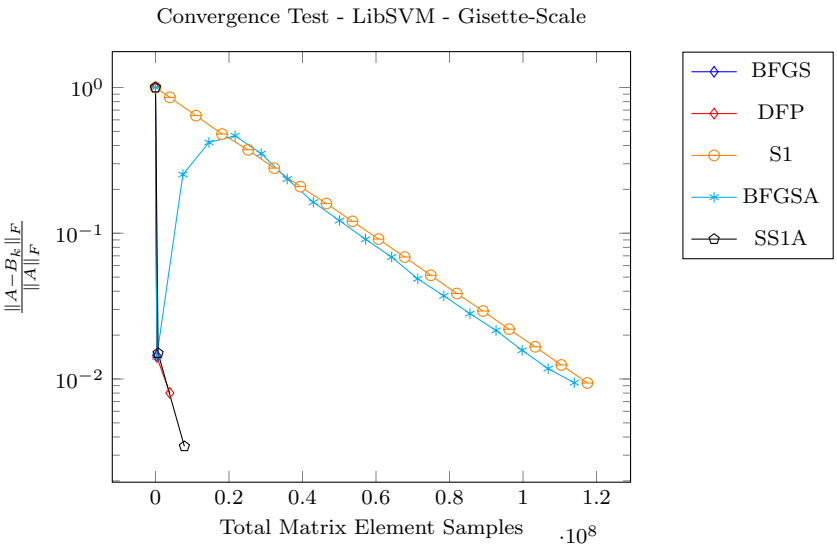

Fig. 5 ($n = 5000$) Approximation of Hessian from **Gisette Scale** [4] $s = 71$.

5.3 Relationship to Algorithms in [9]

We revisit the algorithms that fall in the general framework described in [9]. Recall that such algorithms construct a single (expensive) sub-sample, Q^*AQ, to approximate the action of A associated with its dominant eigenspace. This matrix Q can be computed using a modified block power method, as described

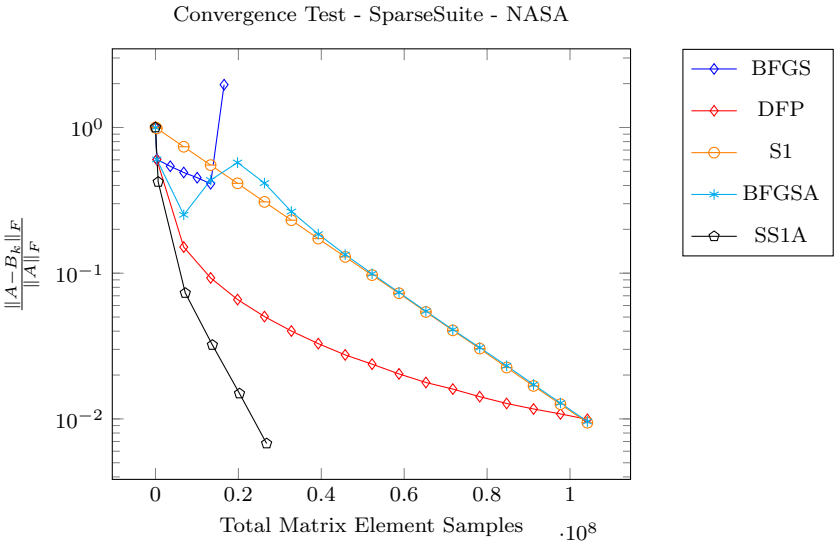

Fig. 6 ($n = 4700$) Approximation of **NASA4704** from [5] $s = 69$.

in algorithm 5. Further, recall that for SPD A, the sub-sampled data is em-

1: Sample $U \sim \mathcal{N}(0,1)^{n\times s}$
2: Compute $Y_0 = A\,U$ ..... $n\,s$
3: Compute QR-decomposition $Y_0 = Q_0\,R_0$
4: **loop** $\{i = 1, 2, \ldots, p\}$
5: Compute $\tilde{Y}_i = A^*\,Q_{i-1}$
6: Compute QR-decomposition $\tilde{Y}_i = \tilde{Q}_i\,\tilde{R}_i$
7: Compute $Y_i = A\,\tilde{Q}_i$
8: Compute QR-decomposition $Y_i = Q_i\,R_i$
9: **end loop** ..... $2\,p\,n\,s$
10: **return** Q_p ..... $(2\,p+1)(n\,s)$

Algorithm 5: From [9]: Randomized Subspace Iteration (Stage A)

bedded using the low-rank approximation $\mathcal{P}_{I_n,Q}\,A\,\mathcal{P}^T_{I_n,Q}$. Hence, algorithm 5 can be viewed as a single outer loop of algorithm 4 with the modification that intermediate data $\Lambda = AU_{i-1,k} - B_{i-1,k}U_{i-1,k}$ is not used.

5.4 Krylov Spaces

Block Krylov Iteration [11] computes a low-rank approximation by searching the Krylov space

$$\mathcal{V}_p(U_{0,k}) = \text{span}\{A\,U_{0,k}, (A\,A^T)A\,U_{0,k}, \ldots, (A\,A^T)^{p-1}\,A\,U_{0,k}\}$$

of A. The block Krylov iteration algorithm is summarized in algorithm 6.

1: Sample $U \sim \mathcal{N}(0,1)^{n\times s}$
2: Compute $K = \left[A\,U_{0,k} | (A\,A^T)A\,U_{0,k} | \ldots | (A\,A^T)^{p-1}\,A\,U_{0,k}\right]_{m\times ps}$
3: Compute QR-decomposition $K = Q\,R$
4: **return** Q ..... $\boxed{(2\,p-1)(n\,s)}$

Algorithm 6: [11]: Block Krylov Iteration (Stage A)

Algorithm 4 can also be viewed as a modified block Krylov method. Each inner iteration builds approximations in the space

$$\text{span}\{U_{0,k}, (A-B_{0,k})U_{0,k}, (A-B_{1,k})(A-B_{0,k})U_{0,k}, \ldots, \left(\prod_{i=0}^{p-1}(A-B_{i,k})\right)U_{0,k}\},$$

which approximates the Krylov space $\mathcal{V}_p(U_{0,k})$ of the residual $A-B$. In Algorithm 4 each intermediate space $U_{i-1,k} \approx (A-B)^i U_{0,k}$ is only stored during one inner iteration and the Krylov matrix K is never formed.

6 Conclusions and Future Work

In this manuscript, novel methods which we refer to as sub-sampled methods, are developed to iteratively embed aggregate pieces of information from a data matrix to generate an approximate data matrix. These methods are useful if the data matrix is unavailable (e.g., due to privacy concerns), but weighted linear combinations of the rows and columns of the data are available. These methods have a significantly smaller data-footprint than sampled algorithms and the footprint can be tuned by selecting sample sizes s_1 and s_2. The iterative methods are self-correcting with computable convergence rates under reasonable assumptions since they systematically reduce a weighted Frobenius norm of the residual $A-B_k$. The analysis demonstrates that rotationally symmetric sampling is desirable, and tight convergence rates can be derived for the algorithms. Experimentally the sub-sampled algorithms (algorithms 1 to 3) match their convergence rates and have rates comparable to those of sampled algorithms in the literature.

An accelerated hybrid method (algorithm 4) is developed by combining simultaneous iteration (to enrich a subspace) with the sub-sampled update algorithm 2. This accelerated method is shown experimentally to be competitive (in terms of matrix samples) with current accelerated schemes.

The sub-sampled matrix approximation algorithms and theory form a natural foundation for further investigations to generate low-rank matrix approximation and matrix inverse approximations, as well as applying the matrix approximations as preconditioners within a nonlinear optimization setting. Also of interest is the practicality of extending these algorithms to efficiently handle sparse input matrices.

A Weight Matrix Interpretation

The fixed non-rotationally symmetric weight matrices on which classical sampled methods are based (BFGS $W = A$ and DFP $W = A^{-1}$) produce an enhanced initial drop in the appropriate residuals. Implementing algorithms algorithms 1 to 3 with $W = A$ would produce the same temporary effect but as noted before algorithms with $W = A$ are automatically sampled algorithms Moreover, the enhancement is transitory and such weighted algorithms ultimately converge at the rates in theorems 1 to 3 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 4, 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 4 such dynamic weighting requires samples $A\,U$.

B Minimum Change Solutions

The KKT equations [12] for constrained minimum change formulations eqs. (4) and (5) 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. (4) 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

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

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

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

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. (11). The arguments for eq. (5) are similar.

References

1. Abowd, J.M.: The u.s. census bureau adopts differential privacy. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18, pp. 2867–2867. ACM, New York, NY, USA (2018). DOI 10.1145/3219819.3226070. URL http://doi.acm.org/10.1145/3219819.3226070
2. Andoni, A., ên, H.L.N.: Eigenvalues of a matrix in the streaming model, pp. 1729–1737. DOI 10.1137/1.9781611973105.124. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611973105.124
3. Avron, H., Clarkson, K.L., Woodruff, D.P.: Sharper Bounds for Regularized Data Fitting. In: K. Jansen, J.D.P. Rolim, D. Williamson, S.S. Vempala (eds.) Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017), *Leibniz International Proceedings in Informatics (LIPIcs)*, vol. 81, pp. 27:1–27:22. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2017). DOI 10.4230/LIPIcs.APPROX-RANDOM.2017.27. URL http://drops.dagstuhl.de/opus/volltexte/2017/7576
4. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2**, 27:1–27:27 (2011). Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
5. Davis, T.A., Hu, Y.: The university of florida sparse matrix collection. ACM Trans. Math. Softw. **38**(1), 1:1–1:25 (2011). DOI 10.1145/2049662.2049663. URL http://doi.acm.org/10.1145/2049662.2049663
6. Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. Foundations and Trends® in Theoretical Computer Science **9**(3–4), 211–407 (2014). DOI 10.1561/0400000042. URL http://dx.doi.org/10.1561/0400000042
7. Gao, W., Goldfarb, D.: Block BFGS Methods. SIAM J. Optim. **28**(2), 1205–1231 (2018). DOI 10.1137/16M1092106. URL https://doi.org/10.1137/16M1092106
8. Gower, R.M., Richtárik, P.: Randomized quasi-Newton updates are linearly convergent matrix inversion algorithms. SIAM J. Matrix Anal. Appl. **38**(4), 1380–1409 (2017). DOI 10.1137/16M1062053. URL https://doi.org/10.1137/16M1062053
9. Halko, N., Martinsson, P., Tropp, J.: Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM Review **53**(2), 217–288 (2011). DOI 10.1137/090771806. URL https://doi.org/10.1137/090771806
10. Li, Y., ên, H.L.N., Woodruff, D.P.: On Sketching Matrix Norms and the Top Singular Vector, pp. 1562–1581. DOI 10.1137/1.9781611973402.114. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611973402.114
11. Musco, C., Musco, C.: Randomized block krylov methods for stronger and faster approximate singular value decomposition. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1, NIPS'15, pp. 1396–1404. MIT Press, Cambridge, MA, USA (2015). URL http://dl.acm.org/citation.cfm?id=2969239.2969395
12. Nocedal, J., Wright, S.J.: Numerical optimization, second edn. Springer Series in Operations Research and Financial Engineering. Springer, New York (2006)
13. Roosta-Khorasani, F., Mahoney, M.W.: Sub-sampled newton methods i: Globally convergent algorithms. ArXiv **abs/1601.04737** (2016)
14. S. Berahas, A., Bollapragada, R., Nocedal, J.: An investigation of Newton-sketch and subsampled Newton methods. PrePrint (2017)
15. Schnabel, R.: Quasi-newton methods using multiple secant equations. Computer Science Technical Reports **244**, 41 (1983). DOI https://scholar.colorado.edu/csci_techreports/244/
16. Stewart, G.: The efficient generation of random orthogonal matrices with an application to condition estimators. SIAM Journal on Numerical Analysis **17**(3), 403–409 (1980). DOI 10.1137/0717034. URL https://doi.org/10.1137/0717034
17. Wilson, R.J., Zhang, C.Y., Lam, W., Desfontaines, D., Simmons-Marengo, D., Gipson, B.: Differentially private sql with bounded user contribution. arXiv e-prints p. arXiv:1909.01917 (2019)