SIAM J. MATRIX ANAL. APPL.
Vol. 38, No. 4, pp. 1380–1409

RANDOMIZED QUASI-NEWTON UPDATES ARE LINEARLY CONVERGENT MATRIX INVERSION ALGORITHMS*

ROBERT M. GOWER† AND PETER RICHTÁRIK†

Abstract. We develop and analyze a broad family of stochastic/randomized algorithms for calculating an approximate inverse matrix. We also develop specialized variants maintaining symmetry or positive definiteness of the iterates. All methods in the family converge globally and linearly (i.e., the error decays exponentially), with explicit rates. In special cases, we obtain stochastic block variants of several quasi-Newton updates, including bad Broyden (BB), good Broyden (GB), Powell-symmetric-Broyden (PSB), Davidon–Fletcher–Powell (DFP), and Broyden–Fletcher–Goldfarb–Shanno (BFGS). Ours are the first stochastic versions of these updates shown to converge to an inverse of a fixed matrix. Through a dual viewpoint we uncover a fundamental link between quasi-Newton updates and approximate inverse preconditioning. Further, we develop an adaptive variant of randomized block BFGS, where we modify the distribution underlying the stochasticity of the method throughout the iterative process to achieve faster convergence. By inverting several matrices from varied applications, we demonstrate that adaptive randomized BFGS (AdaRBFGS) is highly competitive when compared to the Newton–Schulz method, a minimal residual method and direct inversion method based on a Cholesky decomposition. In particular, on large-scale problems our method outperforms the standard methods by orders of magnitude at calculating an approximate inverse. Development of efficient methods for estimating the inverse of very large matrices is a much needed tool for preconditioning and variable metric optimization methods in the advent of the big data era.

Key words. matrix inversion, stochastic methods, iterative methods, quasi-Newton, BFGS, stochastic convergence

AMS subject classifications. 15A09, 90C53, 68W20, 65N75, 65F35, 65Y20, 68Q25, 68W40

DOI. 10.1137/16M1062053

1. Introduction.

Matrix inversion is a standard tool in numerics that is needed, for instance, in computing a projection matrix or a Schur complement, which are commonplace calculations. When only an approximate inverse is required, then iterative methods are the methods of choice, for they can terminate the iterative process when the desired accuracy is reached. This can be far more efficient than using a direct method. Calculating an approximate inverse is a much needed tool in preconditioning [33], and, if the output is guaranteed to be positive definite, then it can be used to design variable metric optimization methods. Furthermore, iterative methods can make use of an initial estimate of the inverse when available.

The driving motivation of this work is the need to develop algorithms capable of computing an approximate inverse of very large matrices, where standard techniques take an exorbitant amount of time or simply fail. In particular, we develop a family of randomized/stochastic methods for inverting a matrix, with specialized variants maintaining symmetry or positive definiteness of the iterates. All methods in the family converge globally (i.e., from any starting point) and linearly (i.e., the error decays exponentially). We give an explicit expression for the convergence rate.

*Received by the editors February 19, 2016; accepted for publication (in revised form) by M. P. Friedlander September 19, 2017; published electronically November 14, 2017.

http://www.siam.org/journals/simax/38-4/M106205.html

Funding: The work of the second author was supported by the EPSRC grant EP/K02325X/1, *Accelerated Coordinate Descent Methods for Big Data Optimization*, and the EPSRC Fellowship EP/N005538/1, *Randomized Algorithms for Extreme Convex Optimization.*

†School of Mathematics, The Maxwell Institute for Mathematical Sciences, University of Edinburgh, Edinburgh EH9 3FD, UK (gowerrobert@gmail.com, peter.richtarik@ed.ac.uk).

As special cases, we obtain stochastic block variants of several quasi-Newton (qN) updates, including bad Broyden (BB), good Broyden (GB), Powell-symmetric-Broyden (PSB), Davidon–Fletcher–Powell (DFP), and Broyden–Fletcher–Goldfarb–Shannono (BFGS). To the best of our knowledge, these are the first stochastic versions of qN updates. Moreover, this is the first time that qN updates are shown to be iterative methods for inverting a matrix. We offer a new interpretation of the qN methods through a Lagrangian dual viewpoint, uncovering a fundamental link between qN updates and approximate inverse preconditioning.

We develop an adaptive variant of randomized block BFGS, in which we modify the distribution underlying the stochasticity of the method throughout the iterative process to achieve faster convergence. Through extensive numerical experiments with large matrices arising from several applications, we show that adaptive randomized BFGS (AdaRBFGS) can significantly outperform the well-established Newton–Schulz and minimal residual methods.

1.1. Outline. The rest of the paper is organized as follows. In section 2 we summarize the main contributions of this paper. In section 3 we describe the qN methods, which are the main inspiration of our methods. Subsequently, section 4 describes two algorithms, each corresponding to a variant of the inverse equation, for inverting general square matrices. We also provide insightful dual viewpoints for both methods. In section 5 we describe a method specialized to inverting symmetric matrices. Convergence in expectation is examined in section 6, where we consider two types of convergence: the convergence of (i) the expected norm of the error, and the convergence of (ii) the norm of the expected error. In section 7 we specialize our methods to discrete distributions and comment on how one may construct a probability distribution leading to better complexity rates (i.e., importance sampling). We then describe a convenient probability distribution which leads to convergence rates which can be described in terms of spectral properties of the original matrix to be inverted. In section 8 we describe several instantiations of our family of methods. We show how via the choice of the parameters of the method, we obtain *stochastic block variants* of several well-known qN updates and a simultaneous randomized Kaczmarz method. Section 9 is dedicated to the development of an adaptive variant of our randomized BFGS method, AdaRBFGS, for inverting positive definite matrices. This method adapts stochasticity throughout the iterative process to obtain faster practical convergence. Finally, in section 10 we show through numerical tests that AdaRBFGS significantly outperforms the Newton–Schulz and minimal residual methods on large-scale matrices.

1.2. Notation. By I we denote the $n \times n$ identity matrix. Let $\langle X, Y \rangle_{F(W^{-1})} \overset{\text{def}}{=} \mathbf{Tr}(X^T W^{-1} Y W^{-1})$ denote the weighted Frobenius inner product, where $X, Y \in \mathbb{R}^{n \times n}$ and $W \in \mathbb{R}^{n \times n}$ is a symmetric positive definite "weight" matrix. Further, let

$$\|X\|^2_{F(W^{-1})} \overset{\text{def}}{=} \mathbf{Tr}(X^T W^{-1} X W^{-1}) = \|W^{-1/2} X W^{-1/2}\|^2_F, \tag{1.1}$$

where we have used the convention $F = F(I)$, since $\|\cdot\|_{F(I)}$ is the standard Frobenius norm. Let $\|\cdot\|_2$ denote the induced operator norm for square matrices defined via $\|Y\|_2 \overset{\text{def}}{=} \max_{\|v\|_2=1} \|Yv\|_2$. Finally, for positive definite $W \in \mathbb{R}^{n \times n}$, we define the weighted induced norm via $\|Y\|_{W^{-1}} \overset{\text{def}}{=} \|W^{-1/2} Y W^{-1/2}\|_2$.

1.3. Previous work. A widely studied iterative method for inverting matrices is the Newton–Schulz method [34], introduced in 1933, and its variants; these methods

are still the subject of ongoing research [27]. The drawback of the Newton–Schulz methods is that they do not converge for every initial estimate. Instead, an initial estimate X_0 such that $\|I - AX_0\|_2 < 1$ is required. In contrast, the methods we present converge globally for any initial estimate. Bingham [3] introduced in 1941 a method that uses the characteristic polynomial to recursively calculate the inverse, though it requires calculating the coefficients of the characteristic polynomial when initiated, which is costly, and thus the method has fallen into disuse. Goldfarb [11] uses Broyden's method [4] for iteratively inverting matrices. Our methods include a stochastic variant of Broyden's method.

The *approximate inverse preconditioning* (AIP) methods [6, 33, 13, 1] calculate an approximate inverse by minimizing the residual $\|XA - I\|_F$ in X. A considerable drawback of the AIP methods is that the iterates are not guaranteed to be positive definite or symmetric, even when A is both. A solution to the lack of symmetry is to symmetrize the estimate between iterations. However, it is then difficult to guarantee the quality of the new symmetric estimate. Another solution is to calculate directly a factored form $LL^T = X$ and minimize in L the residual $\left\|L^T AL - I\right\|_F$. But this residual is a nonconvex function and is thus difficult to minimize. A variant of our method naturally maintains symmetry of the iterates.

2. Contributions.

We now describe the main contributions of this work.

2.1. New algorithms.

We develop a novel and surprisingly simple family of stochastic algorithms for inverting matrices. The problem of finding the inverse of an $n \times n$ invertible matrix A can be characterized as finding the solution to either one of the two *inverse equations*[1] $AX = I$ or $XA = I$. Our methods make use of randomized sketching [31, 17, 30, 32, 14] to reduce the dimension of the inverse equations in an iterative fashion. To the best of our knowledge, these are the first stochastic algorithms for inverting a matrix with global complexity rates.

In particular, our nonsymmetric method (Algorithm 1) is based on the inverse equation $AX = I$ and performs the *sketch-and-project* iteration

$$X_{k+1} = \arg\min_{X \in \mathbb{R}^{n\times n}} \tfrac{1}{2}\|X - X_k\|^2_{F(W^{-1})} \quad \text{subject to} \quad S^T AX = S^T, \tag{2.1}$$

where $S \in \mathbb{R}^{n\times q}$ is a random matrix drawn in an i.i.d. fashion from a fixed distribution $\mathcal{D}$, and $W \in \mathbb{R}^{n\times n}$ is the positive definite "weight" matrix. The distribution $\mathcal{D}$ and matrix W are the parameters of the method. Note that if we choose $q \ll n$, the constraint in the projection problem (2.1) will be of a much smaller dimension than the original inverse equation, and hence the iteration (2.1) will become cheap.

In an analogous way, we design a method based on the inverse equation $XA = I$ (Algorithm 2). Adding the symmetry constraint $X = X^T$ leads to Algorithm 3—a specialized method for symmetric A capable of maintaining symmetric iterates.

2.2. Dual formulation.

Besides the *primal formulation* described in section 2.1, *sketch-and-project*, we also provide *dual formulations* of all three methods (Algorithms 1, 2, and 3). For instance, the dual formulation of (2.1) is

$$X_{k+1} = \arg_X \min_{X,Y} \tfrac{1}{2}\left\|X_k - A^{-1}\right\|^2_{F(W^{-1})} \quad \text{s.t.} \quad X = X_k + WA^T SY^T, \tag{2.2}$$

[1] One may use other equations uniquely defining the inverse, such as $AXA = A$, but we do not explore these in this paper.

TABLE 2.1
Our main complexity results.

$\mathbf{E}\left[X_{k+1}-A^{-1}\right]=(I-W\mathbf{E}\left[Z\right])\mathbf{E}\left[X_k-A^{-1}\right]$	Theorem 4.1
$\left\|\mathbf{E}\left[X_{k+1}-A^{-1}\right]\right\|^2_{W^{-1}}\leq\rho^2\ \cdot\ \left\|\mathbf{E}\left[X_k-A^{-1}\right]\right\|^2_{W^{-1}}$	Theorem 6.4
$\mathbf{E}\left[\left\|X_{k+1}-A^{-1}\right\|^2_{F(W^{-1})}\right]\leq\rho\ \cdot\ \mathbf{E}\left[\left\|X_k-A^{-1}\right\|^2_{F(W^{-1})}\right]$	Theorem 6.5

where the minimization is performed over $X\in\mathbb{R}^{n\times n}$ and $Y\in\mathbb{R}^{n\times q}$. We call the dual formulation *constrain-and-approximate* as one seeks to perform the best approximation of the inverse (with respect to the weighted Frobenius distance) while constraining the search to a random affine space of matrices passing through X_k. While the projection (2.2) cannot be performed directly since A^{-1} is not known, it can be performed indirectly via the equivalent primal formulation (2.1).

2.3. Quasi-Newton updates and approximate inverse preconditioning. As we will discuss in section 3, through the lens of the sketch-and-project formulation, Algorithm 3 can be seen as a *randomized block extension of the quasi-Newton (qN) updates* [4, 10, 12, 35]. We distinguish here between qN methods, which are algorithms used in optimization, and qN updates, which are *matrix-update* rules used in qN methods. Standard qN updates work with $q=1$ ("block" refers to the choice $q>1$) and S chosen in a deterministic way, depending on the sequence of iterates of the underlying optimization problem. To the best of our knowledge, this is the first time stochastic versions of qN updates have been designed and analyzed. On the other hand, through the lens of the constrain-and-approximate formulation, our methods can be seen as *new variants of the approximate inverse preconditioning* (AIP) *methods* [6, 33, 13, 1]. Moreover, the equivalence between these two formulations reveals deep connections between what were before seen as distinct fields: the qN and AIP literature. Our work also provides several new insights for *deterministic* qN updates. For instance, the *bad Broyden* (BB) *update* [4, 21] is a particular best rank-1 update that minimizes the distance to the inverse of A under the Frobenius norm. The *BFGS update* [4, 10, 12, 35] can be seen as a projection of A^{-1} onto a space of rank-2 symmetric matrices. It seems this has not been observed before.

2.4. Complexity: General results. Our framework leads to global linear convergence (i.e., exponential decay) under very weak assumptions on $\mathcal{D}$. In particular, we provide an explicit convergence rate ρ for the exponential decay of the norm of the expected error of the iterates (line 2 of Table 2.1) and the expected norm of the error (line 3 of Table 2.1), where the rate is given by

$$\rho=1-\lambda_{\min}(W^{1/2}\mathbf{E}\left[Z\right]W^{1/2}), \tag{2.3}$$

where $Z\overset{\text{def}}{=}A^TS(S^TAWA^TS)^{-1}SA^T$. We show that ρ is always bounded between 0 and 1. Furthermore, we provide a lower bound on ρ that shows that the rate can potentially improve as the number of columns in S increases. This sets our method apart from current methods for inverting matrices that lack global guarantees, such as Newton–Schulz, or the self-conditioning variants of the minimal residual method.

2.5. Complexity: Discrete distributions. We detail a convenient choice of probability for discrete distributions $\mathcal{D}$ that gives easy-to-interpret convergence results

depending on a scaled condition number of A. This way we obtain methods for inverting matrices with the same convergence rate as the randomized Kaczmarz method [37] and randomized coordinate descent [25] for solving linear systems. We also obtain importance sampling results by optimizing an upper bound on the convergence rate.

2.6. Adaptive randomized BFGS. We develop an additional highly efficient method—adaptive randomized BFGS (AdaRBFGS)—for calculating an approximate inverse of *positive definite matrices*. Not only does the method greatly outperform the Newton–Schulz and approximate inverse preconditioning methods, but it also preserves positive definiteness, a quality not present in previous methods. Therefore, AdaRBFGS can be used to precondition positive definite systems and to design new variable-metric optimization methods. Since the inspiration behind this method comes from the desire to design an *optimal adaptive* distribution for S by examining the complexity rate ρ, this work also highlights the importance of developing algorithms with explicit convergence rates.

2.7. Extensions. This work opens up many possible avenues for extensions. For instance, new efficient methods could be designed by experimenting and analyzing through our framework with different sophisticated sketching matrices S, such as the Walsh–Hadamard matrix [28, 31]. Furthermore, our methods produce low rank estimates of the inverse and can be adapted to calculate low rank estimates of any matrix. They can be applied to singular matrices, in which case they converge to a particular pseudoinverse. Our results can be used to advance work in stochastic variable metric optimization methods, such as the work by Leventhal and Lewis [26], where they present a randomized iterative method for estimating Hessian matrices that converge in expectation with known convergence rates for any initial estimate. Stich, Müller, and Gärtner [36] use Leventhal and Lewis's method to design a stochastic variable metric method for black-box minimization with explicit convergence rates and promising numeric results. We leave these and other extensions to future work.

3. Randomization of quasi-Newton updates. Our methods are inspired by, and in some cases can be considered to be, randomized block variants of the qN updates. Here we explain how our algorithms arise naturally from the qN setting. Readers familiar with qN methods may jump ahead to section 3.3.

3.1. Quasi-Newton methods. A problem of fundamental interest in optimization is the unconstrained minimization problem

$$\min_{x\in\mathbb{R}^n} f(x), \tag{3.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a sufficiently smooth function. qN methods, first proposed by Davidon in 1959 [8], are an extremely powerful and popular class of algorithms for solving this problem, especially in the regime of moderately large n. In each iteration of a qN method, one approximates the function locally around the current iterate x_k by a quadratic of the form

$$f(x_k + s) \approx f(x_k) + (\nabla f(x_k))^T s + \tfrac{1}{2} s^T B_k s, \tag{3.2}$$

where B_k is a suitably chosen approximation of the Hessian ($B_k \approx \nabla^2 f(x_k)$). After this, a direction s_k is computed by minimizing the quadratic approximation in s:

$$s_k = -B_k^{-1} \nabla f(x_k), \tag{3.3}$$

assuming B_k is invertible. The next iterate is then set to

$$x_{k+1} = x_k + h_k, \quad h_k = \alpha_k s_k,$$

for a suitable choice of stepsize α_k, often chosen by a line-search procedure (i.e., by approximately minimizing $f(x_k + \alpha s_k)$ in α).

Gradient descent arises as a special case of this process by choosing B_k to be constant throughout the iterations. A popular choice is $B_k = LI$, where I is the identity matrix and $L \in \mathbb{R}_+$ is the Lipschitz constant of the gradient of f. In such a case, the quadratic approximation (3.2) is a global upper bound on $f(x_k + s)$, which means that $f(x_k + s_k)$ is guaranteed to be at least as good as (i.e., smaller than or equal to) $f(x_k)$, leading to guaranteed descent. Newton's method also arises as a special case, by choosing $B_k = \nabla^2 f(x_k)$. These two algorithms are extreme cases on opposite ends of a spectrum. Gradient descent benefits from a trivial update rule for B_k and from cheap iterations due to the fact that no linear systems need to be solved. However, curvature information is largely ignored, which slows down the practical convergence of the method. Newton's method utilizes the full curvature information contained in the Hessian, but requires the computation of the Hessian in each step, which is expensive for large n. qN methods aim to find a sweet spot on the continuum between these two extremes. In particular, the qN methods choose B_{k+1} to be a matrix for which the *secant equation* is satisfied:

$$B_{k+1}(x_{k+1} - x_k) = \nabla f(x_{k+1}) - \nabla f(x_k). \tag{3.4}$$

The basic reasoning behind this requirement is the following: if f is a convex quadratic, then the Hessian satisfies the secant equation for all pairs of vectors x_{k+1} and x_k. If f is not a quadratic, the reasoning is as follows. Using the fundamental theorem of calculus, we have

$$\left(\int_0^1 \nabla^2 f(x_k + t h_k)\, dt\right)(x_{k+1} - x_k) = \nabla f(x_{k+1}) - \nabla f(x_k) \stackrel{\text{def}}{=} y_k.$$

By selecting B_{k+1} that satisfies the secant equation, we are enforcing B_{k+1} to mimic the action of the integrated Hessian along the line segment joining x_k and x_{k+1}. Unless $n = 1$, the secant equation (3.4) does not have a unique solution in B_{k+1}. All qN methods differ only in which particular solution is used. The formulas transforming B_k to B_{k+1} are called *qN updates*.

Since these matrices are used to compute the direction s_k via (3.3), it is often more reasonable to instead maintain a sequence of inverses $X_k = B_k^{-1}$. By multiplying both sides of (3.4) by X_{k+1}, we arrive at the *secant equation for the inverse*:

$$X_{k+1}(\nabla f(x_{k+1}) - \nabla f(x_k)) = x_{k+1} - x_k. \tag{3.5}$$

The most popular classes of qN updates choose X_{k+1} as the closest matrix to X_k, in a suitable norm (usually a weighted Frobenius norm with various weight matrices), subject to the secant equation, often with an explicit symmetry constraint:

$$X_{k+1} = \arg\min_{X \in \mathbb{R}^{n\times n}} \left\{ \|X - X_k\| \ : \ X y_k = h_k,\ X = X^T \right\}. \tag{3.6}$$

3.2. Quasi-Newton updates. Consider now problem (3.1) with

$$f(x) = \tfrac{1}{2} x^T A x - b^T x + c, \tag{3.7}$$

where A is an $n \times n$ symmetric positive definite matrix, $b \in \mathbb{R}^n$, and $c \in \mathbb{R}$. Granted, this is not a typical problem for which qN methods would be used by a practitioner. Indeed, the Hessian of f does not change, and hence one *does not have to* track it. The problem can simply be solved by setting the gradient to zero, which leads to the system $Ax = b$, the solution being $x_* = A^{-1}b$. As solving a linear system is much simpler than computing the inverse A^{-1}, approximately tracking the (inverse) Hessian of f along the path of the iterates $\{x_k\}$—the basic strategy of all qN methods—seems like too much effort for what is ultimately a much simpler problem.

However, and this is one of the main insights of this work, instead of viewing qN methods as optimization algorithms, we can alternatively interpret them as iterative algorithms producing a sequence of matrices, $\{B_k\}$ or $\{X_k\}$, hopefully converging to some matrix of interest. In particular, one would hope that $X_k \to A^{-1}$ if a qN method is applied to (3.7), with any symmetric positive definite initial guess X_0. In this case, the qN updates of the minimum distance variety given by (3.6) take the form

$$X_{k+1} = \arg\min_{X \in \mathbb{R}^{n\times n}} \left\{ \|X - X_k\| \ : \ XAh_k = h_k, \ X = X^T \right\}. \tag{3.8}$$

3.3. Randomized quasi-Newton updates. While the motivation for our work comes from optimization, having arrived at the update (3.8), we can dispense with some of the implicit assumptions and propose and analyze a wider class of methods. In particular, in this paper we analyze a large class of *randomized algorithms* of the type (3.8), where the vector h_k is replaced by a random matrix S and A is *any* invertible,[2] and not necessarily symmetric or positive definite, matrix. This constitutes a randomized block extension of the qN updates.

4. Inverting nonsymmetric matrices. In this paper we are concerned with the development of a family of stochastic algorithms for computing the inverse of a nonsingular matrix $A \in \mathbb{R}^{n\times n}$. The starting point in the development of our methods is the simple observation that the inverse A^{-1} is the (unique) solution of a linear matrix equation, which we shall refer to as the *inverse equation*:

$$AX = I. \tag{4.1}$$

Alternatively, one can use the inverse equation $XA = I$ instead. Since (4.1) is difficult to solve directly, our approach is to iteratively solve a small randomly relaxed version of (4.1). That is, we choose a random matrix $S \in \mathbb{R}^{n\times q}$, with $q \ll n$, and instead solve the following *sketched inverse equation*:

$$S^T AX = S^T. \tag{4.2}$$

If we base the method on the second inverse equation, the sketched inverse equation $XAS = S$ should be used instead. Note that A^{-1} satisfies (4.2). If $q \ll n$, the sketched inverse equation is of a much smaller dimension than the original one and hence easier to solve. However, the equation will no longer have a unique solution, and in order to design an algorithm, we need a way of picking a particular solution. Our algorithm defines X_{k+1} to be the solution that is closest to the current iterate X_k in a weighted Frobenius norm. This is repeated in an iterative fashion, each time drawing S independently from a fixed distribution $\mathcal{D}$. The distribution $\mathcal{D}$ and the

[2] In fact, one can apply the method to an arbitrary real matrix A, in which case the iterates $\{X_k\}$ converge to the Moore–Penrose pseudoinverse of A. However, this development is outside the scope of this paper and is left for future work.

matrix W can be seen as parameters of our method. The flexibility of being able to adjust $\mathcal{D}$ and W is important: by varying these parameters we obtain various specific instantiations of the generic method, with varying properties and convergence rates. This gives the practitioner the flexibility to adjust the method to the structure of A, to the computing environment, and so on.

4.1. Projection viewpoint: Sketch-and-project. The next iterate X_{k+1} is the nearest point to X_k that satisfies a *sketched* version of the inverse equation:

$$\boxed{X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}} \tfrac{1}{2}\left\|X - X_k\right\|^2_{F(W^{-1})} \quad \text{subject to} \quad S^TAX = S^T} \tag{4.3}$$

In the special case when $S = I$, the only such matrix is the inverse itself, and (4.3) is not helpful. However, if S is "simple," (4.3) will be easy to compute, and the hope is that through a sequence of such steps, where the matrices S are sampled in an i.i.d. fashion from some distribution, X_k will converge to A^{-1}.

Alternatively, we can sketch the equation $XA = I$ and project onto $XAS = S$:

$$\boxed{X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}} \tfrac{1}{2}\left\|X - X_k\right\|^2_{F(W^{-1})} \quad \text{subject to} \quad XAS = S} \tag{4.4}$$

While method (4.3) sketches the rows of A, method (4.4) sketches the columns of A. Thus, we refer to (4.3) as the row variant and to (4.4) as the column variant. Both variants converge to the inverse of A, as will be established in section 6. If A is singular, then it can be shown that the iterates of (4.4) converge to the left inverse, while the iterates of (4.3) converge to the right inverse.

4.2. Optimization viewpoint: Constrain-and-approximate. The row sketch-and-project method can be cast in an apparently different yet equivalent way:

$$\boxed{X_{k+1} = \arg_X \min_{X,Y} \tfrac{1}{2}\left\|X - A^{-1}\right\|^2_{F(W^{-1})} \quad \text{s.t.} \quad X = X_k + WA^TSY^T} \tag{4.5}$$

Minimization is done over $X \in \mathbb{R}^{n\times n}$ and $Y \in \mathbb{R}^{n\times q}$. In this viewpoint, in each iteration (4.5), we select a random affine space that passes through X_k and then select the point in this space that is as close as possible to the inverse. This random search space is special in that, independently of the input pair (W, S), we can efficiently compute the projection of A^{-1} onto this space, without knowing A^{-1} explicitly.

Method (4.4) also has an equivalent constrain-and-approximate formulation:

$$\boxed{X_{k+1} = \arg_X \min_{X,Y} \tfrac{1}{2}\left\|X - A^{-1}\right\|^2_{F(W^{-1})} \quad \text{s.t.} \quad X = X_k + YS^TA^TW} \tag{4.6}$$

Methods (4.5) and (4.6) can be viewed as new variants of approximate inverse preconditioning (AIP) [1, 13, 24, 23], which is a class of methods for computing an approximate inverse of A by minimizing $\|XA - I\|_F$ via iterative optimization algorithms, such as steepest descent or the minimal residual method. Our methods use a different iterative procedure (projection onto a randomly generated affine space) and work with a more general norm (weighted Frobenius norm).

4.3. Equivalence. We now prove that (4.3) and (4.4) are equivalent to (4.5) and (4.6), respectively, and give their explicit solution.

THEOREM 4.1. *Viewpoints* (4.3) *and* (4.5) *are equivalent to* (4.4) *and* (4.6)*, respectively. Further, if S has full column rank, then the explicit solution to* (4.3) *is*

$$\boxed{X_{k+1} = X_k + WA^TS(S^TAWA^TS)^{-1}S^T(I - AX_k)} \tag{4.7}$$

and the explicit solution to (4.4) *is*

$$\boxed{X_{k+1} = X_k + (I - X_kA^T)S(S^TA^TWAS)^{-1}S^TA^TW} \tag{4.8}$$

Proof. We will prove all the claims for the row variant; that is, we prove that (4.3) and (4.5) are equivalent and that their solution is given by (4.7). The remaining claims, that (4.4) and (4.6) are equivalent and that their solution is given by (4.8), follow with analogous arguments. In view of (1.1), with the change of variables

$$\hat{X} \overset{\text{def}}{=} W^{-1/2}XW^{-1/2}, \quad \hat{A} \overset{\text{def}}{=} W^{1/2}AW^{1/2}, \quad \hat{S} \overset{\text{def}}{=} W^{-1/2}S, \tag{4.9}$$

(4.3) becomes

$$\min_{\hat{X}\in\mathbb{R}^{n\times n}} \tfrac{1}{2}\|\hat{X} - \hat{X}_k\|_F^2 \quad \text{subject to} \quad \hat{S}^T\hat{A}\hat{X} = \hat{S}^T. \tag{4.10}$$

Moreover, if we let $\hat{Y} = W^{-1/2}Y$, then (4.5) becomes

$$\min_{\hat{X}\in\mathbb{R}^{n\times n},\hat{Y}\in\mathbb{R}^{n\times q}} \tfrac{1}{2}\|\hat{X} - \hat{A}^{-1}\|_F^2 \quad \text{subject to} \quad \hat{X} = \hat{X}_k + \hat{A}^T\hat{S}\hat{Y}^T. \tag{4.11}$$

By substituting the constraint in (4.11) into the objective function, then differentiating to find the stationary point, we obtain that

$$\hat{X} = \hat{X}_k + \hat{A}^T\hat{S}(\hat{S}^T\hat{A}\hat{A}^T\hat{S})^{-1}\hat{S}^T(I - \hat{A}\hat{X}_k) \tag{4.12}$$

is the solution to (4.11). Changing the variables back using (4.9), (4.12) becomes (4.7).

Now we prove the equivalence of (4.10) and (4.11) using Lagrangian duality. The sketch-and-project viewpoint (4.10) has a convex quadratic objective function with linear constraints; thus strong duality holds. Introducing Lagrangian multiplier $\hat{Y} \in \mathbb{R}^{n\times q}$, the Langrangian dual of (4.10) is given by

$$L(\hat{X}, \hat{Y}) = \tfrac{1}{2}\|\hat{X} - \hat{X}_k\|_F^2 - \langle \hat{Y}^T, \hat{S}^T\hat{A}(\hat{X} - \hat{A}^{-1})\rangle_F. \tag{4.13}$$

Clearly, (4.10) $= \min_{X\in\mathbb{R}^{n\times n}} \max_{\hat{Y}\in\mathbb{R}^{n\times q}} L(\hat{X}, \hat{Y})$. We will now prove that (4.11) $= \max_{\hat{Y}\in\mathbb{R}^{n\times q}} \min_{X\in\mathbb{R}^{n\times n}} L(\hat{X}, \hat{Y})$, thus proving that (4.10) and (4.11) are equivalent by strong duality. Differentiating the Lagrangian in $\hat{X}$ and setting to zero gives

$$\hat{X} = \hat{X}_k + \hat{A}^T\hat{S}\hat{Y}^T. \tag{4.14}$$

Substituting into (4.13) gives $L(\hat{X}, \hat{Y}) = \frac{1}{2}\|\hat{A}^T\hat{S}\hat{Y}^T\|_F^2 - \langle \hat{A}^T\hat{S}\hat{Y}^T, \hat{X}_k + \hat{A}^T\hat{S}\hat{Y}^T - \hat{A}^{-1}\rangle_F = -\frac{1}{2}\|\hat{A}^T\hat{S}\hat{Y}^T\|_F^2 - \langle \hat{A}^T\hat{S}\hat{Y}^T, \hat{X} - \hat{A}^{-1}\rangle_F$. Adding $\pm\frac{1}{2}\|\hat{X}_k - \hat{A}^{-1}\|_F^2$ to the above, we get $L(\hat{X}, \hat{Y}) = -\frac{1}{2}\|\hat{A}^T\hat{S}\hat{Y}^T + \hat{X}_k - \hat{A}^{-1}\|_F^2 + \frac{1}{2}\|\hat{X}_k - \hat{A}^{-1}\|_F^2$. Finally, substituting (4.14) into the last equation, minimizing in $\hat{X}$, then maximizing in $\hat{Y}$, and dispensing with the term $\frac{1}{2}\|\hat{X}_k - \hat{A}^{-1}\|_F^2$ as it does not depend on $\hat{Y}$ or $\hat{X}$, we obtain the dual problem:

$$\max_{\hat{Y}} \min_{\hat{X}} L(\hat{X}, \hat{Y}) = \min_{\hat{X},\hat{Y}} \tfrac{1}{2}\|\hat{X} - \hat{A}^{-1}\|_F^2 \quad \text{subject to} \quad \hat{X} = \hat{X}_k + \hat{A}^T\hat{S}\hat{Y}^T.$$

It now remains to change variables using (4.9) and set $Y = W^{1/2}\hat{Y}$ to obtain (4.5). ∎

Algorithm 1. Stochastic Iterative Matrix Inversion (SIMI) – nonsymmetric row variant.

input: invertible matrix $A \in \mathbb{R}^{n\times n}$
parameters: $\mathcal{D}$ = distribution over random matrices; positive definite matrix $W \in \mathbb{R}^{n\times n}$
initialize: arbitrary square matrix $X_0 \in \mathbb{R}^{n\times n}$
for $k = 0, 1, 2, \ldots$ **do**
  Sample an independent copy $S \sim \mathcal{D}$
  Compute $\Lambda = S(S^T AWA^T S)^{-1} S^T$
  $X_{k+1} = X_k + WA^T \Lambda(I - AX_k)$ ▷ This is equivalent to (4.3) and (4.5)
output: last iterate X_k

Algorithm 2. Stochastic Iterative Matrix Inversion (SIMI) – nonsymmetric column variant.

input: invertible matrix $A \in \mathbb{R}^{n\times n}$
parameters: $\mathcal{D}$ = distribution over random matrices; pos. def. matrix $W \in \mathbb{R}^{n\times n}$
initialize: arbitrary square matrix $X_0 \in \mathbb{R}^{n\times n}$
for $k = 0, 1, 2, \ldots$ **do**
  Sample an independent copy $S \sim \mathcal{D}$
  Compute $\Lambda = S(S^T A^T WAS)^{-1} S^T$
  $X_{k+1} = X_k + (I - X_k A^T)\Lambda A^T W$ ▷ This is equivalent to (4.4) and (4.6)
output: last iterate X_k

Based on Theorem 4.1, we can summarize the methods described in this section as Algorithms 1 and 2. The explicit formulas (4.7) and (4.8) for (4.3) and (4.4) allow us to efficiently implement these methods and facilitate convergence analysis. In particular, we now see that the convergence analysis of (4.8) follows trivially from analyzing (4.7). This is because (4.7) and (4.8) differ only in terms of a transposition. That is, transposing (4.8) gives $X_{k+1}^T = X_k^T + WAS(S^T A^T WAS)^{-1} S^T (I - A^T X_k^T)$, which is the solution to the row variant of the sketch-and-project viewpoint but where the equation $A^T X^T = I$ is sketched instead of $AX = I$. Thus, since the weighted Frobenius norm is invariant under transposition, it suffices to study the convergence of (4.7); convergence of (4.8) follows by simply swapping the role of A for that of A^T. We summarize this observation in the following remark.

Remark 4.1. The expression for the rate of convergence of Algorithm 2 is the same as the expression for the rate of convergence of Algorithm 1, but with every occurrence of A swapped for A^T.

4.4. Relation to multiple linear systems. Any iterative method for solving linear systems can be applied to the n linear systems that define the inverse through $AX = I$ to obtain an approximate inverse. However, not all methods for solving linear systems can be applied to solve these n linear systems simultaneously, which is necessary for efficient matrix inversion.

The recently proposed methods in [17] for solving linear systems can be easily and efficiently generalized to inverting a matrix, and the resulting method is equivalent to our row variant method (4.3) and (4.5). To show this, we perform the change of

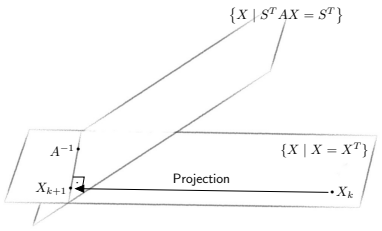


FIG. 5.1. *The new estimate X_{k+1} is obtained by projecting X_k onto the affine space formed by intersecting $\{X \mid X = X^T\}$ and $\{X \mid S^T A X = S^T\}$.*

variables $\hat{X}_k = X_k W^{-1/2}$, $\hat{A} = W^{1/2} A$, and $\hat{S} = W^{-1/2} S$; then (4.3) becomes

$$\hat{X}_{k+1} \overset{\text{def}}{=} X_{k+1} W^{-1/2} = \arg \min_{\hat{X} \in \mathbb{R}^{n \times n}} \tfrac{1}{2} \|W^{-1/2}(\hat{X} - \hat{X}_k)\|_F^2 \quad \text{s.t.} \quad \hat{S}^T \hat{A} \hat{X} = \hat{S}^T.$$

The above is a separable problem, and each column of $\hat{X}_{k+1}$ can be calculated separately. Let $\hat{x}_{k+1}^i$ be the ith column of $\hat{X}_{k+1}$ which can be calculated through

$$\hat{x}_{k+1}^i = \arg \min_{\hat{x} \in \mathbb{R}^n} \tfrac{1}{2} \|W^{-1/2}(\hat{x} - \hat{x}_k^i)\|_2^2 \quad \text{s.t.} \quad \hat{S}^T \hat{A} \hat{x} = \hat{S}^T e_i.$$

The above was proposed as a method for solving linear systems in [17] applied to the system $\hat{A}\hat{x} = e_i$. Thus, the convergence results established in [17] carry over to our row variant (4.3) and (4.5). In particular, the theory in [17] proves that the expected norm difference of each column of $W^{-1/2} X_k$ converges to $W^{-1/2} A^{-1}$ with rate ρ as defined in (2.3). This equivalence breaks down when we impose additional matrix properties through constraints, such as symmetry.

5. Inverting symmetric matrices. When A is symmetric, it may be useful to maintain symmetry in the iterates, in which case the nonsymmetric methods—Algorithms 1 and 2—have an issue, as they do not guarantee that the iterates are symmetric. However, we can modify (4.3) by adding a symmetry constraint. The resulting *symmetric* method naturally maintains symmetry in the iterates.

5.1. Projection viewpoint: Sketch-and-project. The new iterate X_{k+1} is the result of projecting X_k onto the space of matrices which satisfy a sketched inverse equation and which are also symmetric, that is,

$$\boxed{X_{k+1} = \arg \min_{X \in \mathbb{R}^{n \times n}} \tfrac{1}{2} \|X - X_k\|_{F(W^{-1})}^2 \quad \text{s.t.} \quad S^T A X = S^T, \quad X = X^T} \tag{5.1}$$

See Figure 5.1 for an illustration of the symmetric update (5.1).

This viewpoint can be seen as a randomized block version of the qN methods [12, 20], as detailed in section 3. The flexibility in using a weighted norm is important for choosing a norm that better reflects the geometry of the problem. For instance, when A is symmetric positive definite, it turns out that $W^{-1} = A$ results in a good method. This added freedom of choosing an appropriate weighting matrix has proven very useful in the qN literature; in particular, the highly successful BFGS method [4, 10, 12, 35] selects W^{-1} as an estimate of the Hessian matrix.

5.2. Optimization viewpoint: Constrain-and-approximate. The viewpoint (5.1) also has an interesting dual viewpoint:

(5.2)
$$\boxed{X_{k+1} = \arg_X \min_{X,Y} \tfrac{1}{2} \left\|X - A^{-1}\right\|^2_{F(W^{-1})} \quad \text{s.t.} \quad X = X_k + \tfrac{1}{2}(YS^TAW + WA^TSY^T)}$$

The minimum is taken over $X \in \mathbb{R}^{n\times n}$ and $Y \in \mathbb{R}^{n\times q}$. The next iterate, X_{k+1}, is the best approximation to A^{-1} restricted to a random affine space of symmetric matrices. Furthermore, (5.2) is a symmetric equivalent of (4.5); that is, the constraint in (5.2) is the result of intersecting the constraint in (4.5) with the space of symmetric matrices.

5.3. Equivalence. We now prove that the two viewpoints (5.1) and (5.2) are equivalent and show their explicit solution.

THEOREM 5.1. *If A and X_k are symmetric, then the viewpoints* (5.1) *and* (5.2) *are equivalent. That is, they define the same X_{k+1}. Furthermore, if S has full column rank, and we let $\Lambda \overset{def}{=} (S^TAWAS)^{-1}$, then the explicit solution to* (5.1) *and* (5.2) *is*

(5.3)
$$\boxed{X_{k+1} = X_k - (X_kA - I)S\Lambda S^TAW + WAS\Lambda S^T(AX_k - I)(AS\Lambda S^TAW - I)}$$

Proof. It was recently shown in [15, section 2] and [22, section 4][3] that (5.3) is the solution to (5.1). We now prove the equivalence of (5.1) and (5.2) using Lagrangian duality. It suffices to prove the claim for $W = I$ as we did in the proof of Theorem 4.1, since the change of variables (4.9) applied to (5.1) shows that (5.1) is equivalent to

$$\min_{\hat{X}\in\mathbb{R}^{n\times n}} \tfrac{1}{2}\|\hat{X} - \hat{X}_k\|_F^2 \quad \text{subject to} \quad \hat{S}^T\hat{A}\hat{X} = \hat{S}^T, \quad \hat{X} = \hat{X}^T. \tag{5.4}$$

Since (5.1) has a convex quadratic objective with linear constraints, strong duality holds. Thus we will derive a dual formulation for (5.4) and then use the change of coordinates (4.9) to recover the solution to (5.1). Let $\hat{Y} \in \mathbb{R}^{n\times q}$ and $\Gamma \in \mathbb{R}^{n\times n}$ and consider the Lagrangian of (5.4) which is

$$L(\hat{X}, \hat{Y}, \Gamma) = \tfrac{1}{2}\|\hat{X} - \hat{X}_k\|_F^2 - \langle \hat{Y}^T, \hat{S}^T\hat{A}(\hat{X} - \hat{A}^{-1})\rangle_F - \langle \Gamma, \hat{X} - \hat{X}^T\rangle_F. \tag{5.5}$$

Differentiating in $\hat{X}$ and setting to zero gives

$$\hat{X} = \hat{X}_k + \hat{A}^T\hat{S}\hat{Y}^T + \Gamma - \Gamma^T. \tag{5.6}$$

Applying the symmetry constraint $X = X^T$ gives $\Gamma - \Gamma^T = \frac{1}{2}(\hat{Y}\hat{S}^T\hat{A} - \hat{A}^T\hat{S}\hat{Y}^T)$. Substituting the above into (5.6) gives

$$\hat{X} = \hat{X}_k + \tfrac{1}{2}(\hat{Y}\hat{S}^T\hat{A} + \hat{A}^T\hat{S}\hat{Y}^T). \tag{5.7}$$

Now let $\Theta = \frac{1}{2}(\hat{Y}\hat{S}^T\hat{A} + \hat{A}^T\hat{S}\hat{Y}^T)$ and note that, since the matrix $\Theta + \hat{X}_k - \hat{A}^{-1}$ is symmetric, we get

$$\langle \hat{A}^T\hat{S}\hat{Y}^T, \Theta + \hat{X}_k - \hat{A}^{-1}\rangle_F = \langle \Theta, \Theta + \hat{X}_k - \hat{A}^{-1}\rangle_F. \tag{5.8}$$

[3] To reinterpret methods for solving linear systems through Bayesian inference, Hennig constructs estimates of the inverse system matrix using the sampled action of a matrix taken during a linear solve [22].

Substituting (5.7) into (5.5) gives $L(\hat{X},\hat{Y},\Gamma) = \frac{1}{2}\|\Theta\|_F^2 - \langle \hat{A}^T\hat{S}\hat{Y}^T, \Theta + \hat{X}_k - \hat{A}^{-1}\rangle_F \overset{(5.8)}{=} \frac{1}{2}\|\Theta\|_F^2 - \langle \Theta, \Theta + \hat{X}_k - \hat{A}^{-1}\rangle_F = -\frac{1}{2}\|\Theta\|_F^2 - \langle \Theta, \hat{X}_k - \hat{A}^{-1}\rangle_F$. Adding $\pm\frac{1}{2}\|\hat{X}_k - \hat{A}^{-1}\|_F^2$ to the above, we obtain $L(\hat{X},\hat{Y},\Gamma) = -\frac{1}{2}\|\Theta + \hat{X}_k - \hat{A}^{-1}\|_F^2 + \frac{1}{2}\|\hat{X}_k - \hat{A}^{-1}\|_F^2$. Finally, using (5.7), maximizing over $\hat{Y}$, and then minimizing over X gives the dual:

$$\min_{\hat{X},\hat{Y}} \tfrac{1}{2}\|\hat{X} - \hat{A}^{-1}\|_F^2 \quad \text{subject to} \quad \hat{X} = \hat{X}_k + \tfrac{1}{2}(\hat{Y}\hat{S}^T\hat{A} + \hat{A}^T\hat{S}\hat{Y}^T).$$

It now remains to change variables according to (4.9) and set $Y = W^{1/2}\hat{Y}$. ∎

Algorithm 3. Stochastic Iterative Matrix Inversion (SIMI) – symmetric variant.

input: symmetric invertible matrix $A \in \mathbb{R}^{n\times n}$
parameters: $\mathcal{D}$ = distribution over random matrices; symmetric pos. def. $W \in \mathbb{R}^{n\times n}$
initialize: symmetric matrix $X_0 \in \mathbb{R}^{n\times n}$
for $k = 0, 1, 2, \ldots$ **do**
  Sample an independent copy $S \sim \mathcal{D}$
  Compute $\Lambda \leftarrow S(S^TAWAS)^{-1}S^T$, $\Theta \leftarrow \Lambda AW$, $M_k \leftarrow X_kA - I$
  $X_{k+1} = X_k - M_k\Theta - (M_k\Theta)^T + \Theta^T(AX_kA - A)\Theta$ ▷ Equiv. to (5.1) & (5.2)
output: last iterate X_k

6. Convergence. We now analyze the convergence of the *error*, $X_k - A^{-1}$, for iterates of Algorithms 1, 2, and 3. For the sake of economy of space, we only analyze Algorithms 1 and 3. Convergence of Algorithm 2 follows from convergence of Algorithm 1 by observing Remark 4.1.

The first analysis we present in section 6.1 is concerned with the convergence of $\left\|\mathbf{E}\left[X_k - A^{-1}\right]\right\|^2$, that is, the *norm of the expected error*. We then analyze the convergence of $\mathbf{E}\left[\left\|X_k - A^{-1}\right\|\right]^2$, the *expected norm of the error*. The latter is a stronger type of convergence, as explained in the following proposition.

PROPOSITION 6.1. *Let $X \in \mathbb{R}^{n\times n}$ be a random matrix, let $\|\cdot\|$ be a matrix norm induced by an inner product, and fix $A^{-1} \in \mathbb{R}^{n\times n}$. Then*

$$\|\mathbf{E}\left[X - A^{-1}\right]\|^2 = \mathbf{E}\left[\|X - A^{-1}\|^2\right] - \mathbf{E}\left[\|X - \mathbf{E}\left[X\right]\|^2\right].$$

Proof. Note that $\mathbf{E}[\|X - \mathbf{E}[X]\|^2] = \mathbf{E}[\|X\|^2] - \|\mathbf{E}[X]\|^2$. Adding and subtracting $\left\|A^{-1}\right\|^2 - 2\langle \mathbf{E}\left[X\right], A^{-1}\rangle$ from the right-hand side and then grouping the appropriate terms yields the desired result. ∎

This shows that if $\mathbf{E}[\|X_k - A^{-1}\|^2]$ converges to zero, then $\left\|\mathbf{E}[X_k - A^{-1}]\right\|^2$ converges to zero. But the converse is not necessarily true. Rather, the variance $\mathbf{E}[\|X_k - \mathbf{E}\left[X_k\right]\|^2]$ must converge to zero for the converse to be true.[4] The convergence of Algorithms 1 and 3 can be characterized by studying the random matrix

$$Z \overset{\text{def}}{=} A^TS(S^TAWA^TS)^{-1}S^TA. \tag{6.1}$$

[4]The convergence of $\|\mathbf{E}[X_k - A^{-1}]\|^2$ is also known in the probability literature as L_2-norm convergence. It also follows trivially from the Markov inequality that convergence in the L_2-norm implies convergence in probability.

The update step of Algorithm 1 can be rewritten as a simple fixed point formula

$$X_{k+1} - A^{-1} = (I - WZ)(X_k - A^{-1}). \tag{6.2}$$

We can also simplify the iterates of Algorithm 3 to

$$X_{k+1} - A^{-1} = (I - WZ)(X_k - A^{-1})(I - ZW). \tag{6.3}$$

The only stochastic component in both methods is contained in the matrix Z, and ultimately, the convergence of the iterates will depend on $\mathbf{E}[Z]$, the expected value of this matrix. Thus we start with two lemmas concerning the Z and $\mathbf{E}[Z]$ matrices.

LEMMA 6.2. *If Z is defined as in* (6.1)*, then*

1. *the eigenvalues of $W^{1/2}ZW^{1/2}$ are either* 0 *or* 1*, and*
2. *$W^{1/2}ZW^{1/2}$ projects onto the q-dimensional subspace* $\mathbf{Range}\left(W^{1/2}A^TS\right)$.

Proof. Using (6.1), simply verifying that $(W^{1/2}ZW^{1/2})^2 = W^{1/2}ZW^{1/2}$ proves that it is a projection matrix and thus has eigenvalues 0 or 1. Furthermore, the matrix $W^{1/2}ZW^{1/2}$ projects onto $\mathbf{Range}\left(W^{1/2}A^TS\right)$, which follows from $W^{1/2}ZW^{1/2}(W^{1/2}A^TS) = W^{1/2}A^TS$ and the fact that $W^{1/2}ZW^{1/2}y = 0$ for all $y \in \mathbf{Null}\left(W^{1/2}A^TS\right)$. Finally, $\dim\left(\mathbf{Range}\left(W^{1/2}A^TS\right)\right) = \mathbf{Rank}\left(W^{1/2}A^TS\right) = \mathbf{Rank}(S) = q$. ∎

LEMMA 6.3. *The spectrum of $W^{1/2}\mathbf{E}[Z]W^{1/2}$ is contained in* $[0, 1]$.

Proof. Let $\hat{Z} = W^{1/2}ZW^{1/2}$; thus $W^{1/2}\mathbf{E}[Z]W^{1/2} = \mathbf{E}[\hat{Z}]$. Since the mapping $A \mapsto \lambda_{\max}(A)$ is convex, by Jensen's inequality we get $\lambda_{\max}(\mathbf{E}[\hat{Z}]) \le \mathbf{E}[\lambda_{\max}(\hat{Z})]$. Applying Lemma 6.2, we conclude that $\lambda_{\max}(\mathbf{E}[\hat{Z}]) \le 1$. The inequality $\lambda_{\min}(\mathbf{E}[\hat{Z}]) \ge 0$ can be shown analogously using convexity of the mapping $A \mapsto -\lambda_{\min}(A)$. ∎

6.1. Norm of the expected error. We start by proving that the norm of the expected error of the iterates of Algorithms 1 and 3 converges to zero. The following theorem is remarkable in that we do not need to make any assumptions on the distribution S, except that S has full column rank. Rather, the theorem pinpoints that convergence depends solely on the spectrum of $I - W^{-1/2}\mathbf{E}[Z]W^{-1/2}$.

THEOREM 6.4. *Let S be a random matrix which has full column rank with probability* 1 *(so that Z is well defined). Then the iterates X_{k+1} of Algorithm* 1 *satisfy*

$$\mathbf{E}\left[X_{k+1} - A^{-1}\right] = (I - W\mathbf{E}[Z])\mathbf{E}\left[X_k - A^{-1}\right]. \tag{6.4}$$

Let $X_0 \in \mathbb{R}^{n\times n}$. If X_k is calculated by either

1. *applying k iterations of Algorithm* 1 *or*
2. *applying k iterations of Algorithm* 3 *(assuming A and X_0 are symmetric),*

then X_k converges to the inverse exponentially fast, according to

$$\left\|\mathbf{E}\left[X_k - A^{-1}\right]\right\|_{W^{-1}} \le \rho^k \left\|X_0 - A^{-1}\right\|_{W^{-1}}, \tag{6.5}$$

where

$$\rho \overset{def}{=} 1 - \lambda_{\min}(W^{1/2}\mathbf{E}[Z]W^{1/2}). \tag{6.6}$$

Moreover, we have the following lower and upper bounds on the convergence rate:

$$0 \le 1 - \mathbf{E}[q]/n \le \rho \le 1. \tag{6.7}$$

Proof. Let

$$R_k \overset{\text{def}}{=} W^{-1/2}(X_k - A^{-1})W^{-1/2} \quad \text{and} \quad \hat{Z} \overset{\text{def}}{=} W^{1/2}ZW^{1/2} \tag{6.8}$$

for all k. Left and right multiplying (6.2) by $W^{-1/2}$ gives

$$R_{k+1} = (I - \hat{Z})R_k. \tag{6.9}$$

Taking expectation with respect to S in (6.9) gives

$$\mathbf{E}[R_{k+1} \mid R_k] = (I - \mathbf{E}[\hat{Z}])R_k. \tag{6.10}$$

Taking full expectation in (6.9) and using the tower rule gives

$$\mathbf{E}\left[R_{k+1}\right] = \mathbf{E}\left[\mathbf{E}\left[R_{k+1} \mid R_k\right]\right] \overset{(6.10)}{=} \mathbf{E}[(I - \mathbf{E}[\hat{Z}])R_k] = (I - \mathbf{E}[\hat{Z}])\mathbf{E}\left[R_k\right]. \tag{6.11}$$

Applying the norm in (6.11) gives

$$\begin{aligned} \|\mathbf{E}\left[X_{k+1} - A^{-1}\right]\|_{W^{-1}} = \|\mathbf{E}[R_{k+1}]\|_2 &\leq \|I - \mathbf{E}[\hat{Z}]\|_2 \|\mathbf{E}[R_k]\|_2 \\ &= \|I - \mathbf{E}[\hat{Z}]\|_2 \|\mathbf{E}[X_k - A^{-1}]\|_{W^{-1}}. \end{aligned} \tag{6.12}$$

Furthermore,

$$\|I - \mathbf{E}[\hat{Z}]\|_2 = \lambda_{\max}\left(I - \mathbf{E}[\hat{Z}]\right) = 1 - \lambda_{\min}\left(\mathbf{E}[\hat{Z}]\right) \overset{(6.6)}{=} \rho, \tag{6.13}$$

where we used the symmetry of $(I - \mathbf{E}[\hat{Z}])$ when passing from the operator norm to the spectral radius. Note that the symmetry of $\mathbf{E}[\hat{Z}]$ derives from the symmetry of $\hat{Z}$. It now remains to unroll the recurrence in (6.12) to get (6.5). Now we analyze the iterates of Algorithm 3. Left and right multiplying (6.3) by $W^{-1/2}$, we have

$$R_{k+1} = P(R_k) \overset{\text{def}}{=} (I - \hat{Z})R_k(I - \hat{Z}). \tag{6.14}$$

Defining $\bar{P} : R \mapsto \mathbf{E}\left[P(R) \,|\, R_k\right]$, taking expectation in (6.14) conditioned on R_k, gives $\mathbf{E}\left[R_{k+1} \mid R_k\right] = \bar{P}(R_k)$. As $\bar{P}$ is a linear operator, taking expectation again yields

$$\mathbf{E}\left[R_{k+1}\right] = \mathbf{E}\left[\bar{P}(R_k)\right] = \bar{P}(\mathbf{E}\left[R_k\right]). \tag{6.15}$$

Letting $|||\bar{P}|||_2 \overset{\text{def}}{=} \max_{\|R\|_2=1} \left\|\bar{P}(R)\right\|_2$, applying norm in (6.15), gives

$$\begin{aligned} \left\|\mathbf{E}\left[X_{k+1} - A^{-1}\right]\right\|_{W^{-1}} = \left\|\mathbf{E}\left[R_{k+1}\right]\right\|_2 &\leq |||\bar{P}|||_2 \left\|\mathbf{E}\left[R_k\right]\right\|_2 \\ &= |||\bar{P}|||_2 \left\|\mathbf{E}\left[X_k - A^{-1}\right]\right\|_{W^{-1}}. \end{aligned} \tag{6.16}$$

Clearly, P is a *positive linear map*; that is, it is linear and maps positive semidefinite matrices to positive semidefinite matrices. Thus, by Jensen's inequality, the map $\bar{P}$ is also a positive linear map. As every positive linear map attains its norm at the identity matrix (see Corollary 2.3.8 in [2]), we have

$$|||\bar{P}|||_2 = \left\|\bar{P}(I)\right\|_2 \overset{(6.14)}{=} \|\mathbf{E}[(I - \hat{Z})I(I - \hat{Z})]\|_2 \overset{\text{(Lemma 6.2)}}{=} \|\mathbf{E}[I - \hat{Z}]\|_2 \overset{(6.13)}{=} \rho.$$

Inserting the above equivalence into (6.16) and unrolling the recurrence gives (6.5).

Finally, to prove (6.7), as proven in Lemma 6.3, the spectrum of $W^{1/2}\mathbf{E}[Z]W^{1/2}$ is contained in $[0, 1]$ consequently $0 \leq \rho \leq 1$. Furthermore, as the trace of a matrix is equal to the sum of its eigenvalues, we have

$$\begin{aligned} \mathbf{E}[q] &\overset{\text{(Lemma 6.2)}}{=} \mathbf{E}[\mathbf{Tr}(W^{1/2}ZW^{1/2})] = \mathbf{Tr}(\mathbf{E}[W^{1/2}ZW^{1/2}]) \\ &\geq n\lambda_{\min}(\mathbf{E}[W^{1/2}ZW^{1/2}]), \end{aligned} \tag{6.17}$$

where we used that $W^{1/2}ZW^{1/2}$ projects onto a q-dimensional subspace (Lemma 6.2), and thus $\mathbf{Tr}(W^{1/2}ZW^{1/2}) = q$. Rearranging (6.17) gives (6.7). ∎

If $\rho = 1$, this theorem does not guarantee convergence. However, $\rho < 1$ when $\mathbf{E}[Z]$ is positive definite, which is the case in all practical variants of our method, some of which we describe in section 8.

6.2. Expectation of the norm of the error. Now we consider the convergence of the expected norm of the error. This form of convergence is preferred, as it also proves that the variance of the iterates converges to zero (see Proposition 6.1).

THEOREM 6.5. *Let S be a random matrix that has full column rank with probability 1 and such that $\mathbf{E}[Z]$ is positive definite, where Z is defined in* (6.1). *Let $X_0 \in \mathbb{R}^{n\times n}$. If X_k is calculated by either*

1. *applying k iterations of Algorithm* 1 *or*
2. *applying k iterations of Algorithm* 3 *(assuming A and X_0 are symmetric),*

then X_k converges to the inverse according to

$$\mathbf{E}\left[\left\|X_k - A^{-1}\right\|^2_{F(W^{-1})}\right] \leq \rho^k \left\|X_0 - A^{-1}\right\|^2_{F(W^{-1})}. \tag{6.18}$$

Proof. First consider Algorithm 1, where X_{k+1} is calculated by iteratively applying (6.2). Using the substitution (6.8) again, from (6.2) we have $R_{k+1} = (I - \hat{Z})R_k$, from which we obtain

$$\begin{aligned} \|R_{k+1}\|^2_F &= \|(I-\hat{Z})R_k\|^2_F = \mathbf{Tr}((I-\hat{Z})(I-\hat{Z})R_kR_k^T) \\ &\overset{\text{(Lemma 6.2)}}{=} \mathbf{Tr}((I-\hat{Z})R_kR_k^T) = \|R_k\|^2_F - \mathbf{Tr}(\hat{Z}R_kR_k^T). \end{aligned} \tag{6.19}$$

Taking expectations, we get $\mathbf{E}[\|R_{k+1}\|^2_F \mid R_k] = \|R_k\|^2_F - \mathbf{Tr}(\mathbf{E}[\hat{Z}]R_kR_k^T)$. Using the inequality $\mathbf{Tr}(\mathbf{E}[\hat{Z}]R_kR_k^T) \geq \lambda_{\min}(\mathbf{E}[\hat{Z}])\mathbf{Tr}(R_kR_k^T)$, which relies on the symmetry of $\mathbf{E}[\hat{Z}]$, we get $\mathbf{E}[\|R_{k+1}\|^2_F \mid R_k] \leq (1 - \lambda_{\min}(\mathbf{E}[\hat{Z}]))\|R_k\|^2_F = \rho \cdot \|R_k\|^2_F$. In order to arrive at (6.18), it now remains to take full expectation, unroll the recurrence, and use the substitution (6.8).

Now we assume that A and X_0 are symmetric and $\{X_k\}$ are the iterates computed by Algorithm 3. Left and right multiplying (6.3) by $W^{-1/2}$, we have

$$R_{k+1} = (I - \hat{Z})R_k(I - \hat{Z}). \tag{6.20}$$

Taking norms, we have $\|R_{k+1}\|^2_F \overset{\text{(Lem. 6.2)}}{=} \mathbf{Tr}(R_k(I-\hat{Z})R_k(I-\hat{Z})) = \mathbf{Tr}(R_kR_k(I - \hat{Z})) - \mathbf{Tr}(R_k\hat{Z}R_k(I-\hat{Z})) \leq \mathbf{Tr}(R_kR_k(I-\hat{Z}))$, where in the last inequality we used that $I - \hat{Z}$ is an orthogonal projection and thus it is symmetric positive semidefinite, whence $\mathbf{Tr}(R_k\hat{Z}R_k(I-\hat{Z})) = \mathbf{Tr}(\hat{Z}^{1/2}R_k(I-\hat{Z})R_k\hat{Z}^{1/2}) \geq 0$. The remainder of the proof follows similar steps as those we used in the first part of the proof from (6.19) onward. ∎

Theorem 6.5 establishes that the expected norm of the error converges exponentially fast to zero. Moreover, the convergence rate ρ is the same as that which appeared in Theorem 6.4, where we established the convergence of the norm of the expected error. Both results can be recast as iteration complexity bounds. For instance, using standard arguments, from Theorem 6.4 we observe that for a given $0 < \epsilon < 1$ we get

$$k \geq \left(\tfrac{1}{2}\right) \tfrac{1}{1-\rho} \log\left(\tfrac{1}{\epsilon}\right) \quad \Rightarrow \quad \left\|\mathbf{E}\left[X_k - A^{-1}\right]\right\|^2_{W^{-1}} \leq \epsilon \left\|X_0 - A^{-1}\right\|^2_{W^{-1}}. \tag{6.21}$$

On the other hand, from Theorem 6.5 we have

$$k \geq \tfrac{1}{1-\rho} \log\left(\tfrac{1}{\epsilon}\right) \quad \Rightarrow \quad \mathbf{E}\left[\left\|X_k - A^{-1}\right\|^2_{F(W^{-1})}\right] \leq \epsilon \left\|X_0 - A^{-1}\right\|^2_{F(W^{-1})}. \tag{6.22}$$

To push the expected norm of the error below ϵ (see (6.22)), we require double the iterates compared to bringing the norm of expected error below ϵ (see (6.21)). This is because in Theorem 6.5 we determined that ρ is the rate at which the expectation of the *squared* norm error converges, while in Theorem 6.4 we determined that ρ is the rate at which the norm, without the square, of the expected error converges. However, as proven in Proposition 6.1, the former is a stronger form of convergence. Thus, Theorem 6.4 does not give a stronger result than Theorem 6.5 gives, but rather, these theorems give qualitatively different results.

7. Discrete random matrices. We now consider the case of a discrete random matrix S. We show that when S is a *complete discrete sampling*, then $\mathbf{E}[Z]$ is positive definite, and thus from Theorems 6.4 and 6.5, Algorithms 1–3 converge.

DEFINITION 7.1 (complete discrete sampling). *The random matrix S has a finite discrete distribution with r outcomes. In particular, $S = S_i \in \mathbb{R}^{n \times q_i}$ with probability $p_i > 0$ for $i = 1, \ldots, r$, where S_i is of full column rank. We say that S is a complete discrete sampling when $\mathbf{S} \overset{def}{=} [S_1, \ldots, S_r] \in \mathbb{R}^{n \times n}$ has full row rank.*

As an example of a complete discrete sampling, let $S = e_i$ (the ith unit coordinate vector in $\mathbb{R}^n$) with probability $p_i = 1/n$ for $i = 1, \ldots, n$. Then $\mathbf{S}$, as defined in Definition 7.1, is equal to the identity matrix: $\mathbf{S} = I$. Consequently, S is a complete discrete sampling. In fact, from any basis of $\mathbb{R}^n$ we could construct a complete discrete sampling in an analogous way.

Next we establish that when S is discrete random matrix, S having a complete discrete distribution is a necessary and sufficient condition for $\mathbf{E}[Z]$ to be positive definite. This will allow us to determine an optimized distribution for S in section 7.1.

PROPOSITION 7.2. *Let S be a discrete random matrix with r outcomes S_r, all of which have full column rank. The matrix $\mathbf{E}[Z]$ is positive definite if and only if S is a complete discrete sampling. Furthermore,*

$$\mathbf{E}[Z] = A^T \mathbf{S} D^2 \mathbf{S}^T A, \qquad \textit{where} \tag{7.1}$$

$$D \overset{def}{=} Diag\left(\sqrt{p_1}(S_1^T A W A^T S_1)^{-1/2}, \ldots, \sqrt{p_r}(S_r^T A W A^T S_r)^{-1/2}\right). \tag{7.2}$$

Proof. Taking the expectation of Z as defined in (6.1) gives

$$\begin{aligned}
\mathbf{E}[Z] &= \textstyle\sum_{i=1}^r A^T S_i (S_i^T A W A^T S_i)^{-1} S_i^T A p_i \\
&= A^T \left(\textstyle\sum_{i=1}^r S_i \sqrt{p_i} (S_i^T A W A^T S_i)^{-1/2} (S_i^T A W A^T S_i)^{-1/2} \sqrt{p_i} S_i^T\right) A \\
&= \left(A^T \mathbf{S} D\right)\left(D \mathbf{S}^T A\right),
\end{aligned}$$

and $\mathbf{E}[Z]$ is clearly positive semidefinite. Note that, since we assume that S has full column rank with probability 1, the matrix D is well defined and nonsingular. Given that $\mathbf{E}[Z]$ is positive semidefinite, we need only show that $\mathbf{Null}(\mathbf{E}[Z])$ contains only the zero vector if and only if S is a complete discrete sampling. Let $v \in \mathbf{Null}(\mathbf{E}[Z])$ and $v \neq 0$; thus $0 = v^\top A^\top \mathbf{S} D^2 \mathbf{S}^\top A v = \|D\mathbf{S}^\top A v\|_2^2$, which shows that $\mathbf{S}^\top A v = 0$ and thus $v \in \mathbf{Null}(\mathbf{S}^\top A)$. As A is nonsingular, it follows that $v = 0$ if and only if $\mathbf{S}^\top$ has full column rank. ∎

With a closed form expression for $\mathbf{E}[Z]$ we can optimize ρ over the possible distributions of S to yield a better convergence rate.

7.1. Optimizing an upper bound on the convergence rate. So far we have proven two different types of convergence for Algorithms 1, 2, and 3 in Theorems 6.4 and 6.5. Furthermore, both forms of convergence depend on the same convergence rate ρ for which we have a closed form expression (6.6).

The availability of a closed form expression for the convergence rate opens up the possibility of designing particular distributions for S optimizing the rate. In [17] it was shown that (in the context of solving linear systems) for a complete discrete sampling, computing the optimal probability distribution, assuming that the matrices $\{S_i\}_{i=1}^r$ are fixed, leads to a semidefinite program (SDP). In some cases, the gain in performance from the optimal probabilities is much larger than the loss incurred by having to solve the SDP. However, this is not always the case. Here we propose an alternative, which is to optimize the following upper bound on the convergence rate:

$$\rho = 1 - \lambda_{\min}(W^{1/2}\mathbf{E}[Z]W^{1/2}) \leq 1 - \frac{1}{\mathbf{Tr}(W^{-1/2}(\mathbf{E}[Z_p])^{-1}W^{-1/2})} \stackrel{\text{def}}{=} \gamma(p). \tag{7.3}$$

By writing Z_p instead of Z, we emphasized the dependence of Z on $p = (p_1, \ldots, p_r) \in \mathbb{R}^r$, belonging to the probability simplex

$$\Delta_r \stackrel{\text{def}}{=} \{p = (p_1, \ldots, p_r) \in \mathbb{R}^r \ : \ \textstyle\sum_{i=1}^r p_i = 1, \ p \geq 0\}.$$

Our goal is to minimize $\gamma(p)$ over Δ_r.

THEOREM 7.3. *Let S be a complete discrete sampling, and let $\overline{S}_i \in \mathbb{R}^{n \times q_i}$, for $i = 1, 2, \ldots, r$, be such that $\mathbf{S}^{-T} = [\overline{S}_1, \ldots, \overline{S}_r]$. Then*

$$\min_{p \in \Delta_r} \gamma(p) \quad = \quad 1 - \left(\textstyle\sum_{i=1}^r \|W^{1/2}A^T S_i \overline{S}_i^T A^{-T} W^{-1/2}\|_F\right)^{-2}. \tag{7.4}$$

Proof. In view of (7.3), minimizing γ in p is equivalent to minimizing the expression $\mathbf{Tr}(W^{-1/2}(\mathbf{E}[Z_p])^{-1}W^{-1/2})$ in p. Further, we have

$$\begin{aligned}
\mathbf{Tr}(W^{-1/2}(\mathbf{E}[Z_p])^{-1}W^{-1/2}) &\stackrel{(7.1)}{=} \mathbf{Tr}(W^{-1/2}(A^T\mathbf{S}D^2\mathbf{S}^TA)^{-1}W^{-1/2}) \\
&= \mathbf{Tr}(W^{-1/2}A^{-1}\mathbf{S}^{-T}D^{-2}\mathbf{S}^{-1}A^{-T}W^{-1/2}) \\
&\stackrel{(7.2)}{=} \textstyle\sum_{i=1}^r \frac{1}{p_i}\mathbf{Tr}(W^{-1/2}A^{-1}\overline{S}_i(S_i^TAWA^TS_i)\overline{S}_i^TA^{-T}W^{-1/2}) \\
&= \textstyle\sum_{i=1}^r \frac{1}{p_i}\left\|W^{1/2}A^{-1}\overline{S}_iS_i^TAW^{-1/2}\right\|_F^2. \qquad (7.5)
\end{aligned}$$

It is now easy to minimize the above subject to the constraint $p \in \Delta_r$, obtaining

$$p_i = \frac{\left\|W^{1/2}A^{-1}\overline{S}_iS_i^TAW^{-1/2}\right\|_F}{\sum_{j=1}^r \left\|W^{1/2}A^{-1}\overline{S}_jS_j^TAW^{-1/2}\right\|_F}, \quad i = 1, 2, \ldots, r. \tag{7.6}$$

Plugging this into (7.5) gives the result (7.4). ∎

Observe that in general, the optimal probabilities (7.6) cannot be calculated, since the formula involves the inverse of A, which is not known. However, if A is symmetric positive definite, we can choose $W = A^2$, which eliminates this issue. If A is not symmetric positive definite, or if we do not wish to choose $W = A^2$, we can approach the formula (7.6) as a recipe for a heuristic choice of the probabilities: we can use the iterates $\{X_k\}$ as a proxy for A^{-1}. With this setup, the resulting method is not guaranteed to converge by the theory developed in this paper. However, in practice one would expect it to work well. We have not done extensive experiments to test this, and leave this to future research. To illustrate, let us consider a concrete simple example. Choose $W = I$ and $S_i = e_i$ (the unit coordinate vector in $\mathbb{R}^n$). We have $\mathbf{S} = [e_1, \dots, e_n] = I$, whence $\overline{S}_i = e_i$ for $i = 1, \dots, r$. Plugging into (7.6), we obtain

$$p_i = \frac{\left\|X_k e_i e_i^T A\right\|_F}{\sum_{j=1}^r \left\|X_k e_j e_j^T A\right\|_F} = \frac{\|X_k e_i\|_2 \left\|e_i^T A\right\|_2}{\sum_{j=1}^r \|X_k e_j\|_2 \left\|e_j^T A\right\|_2}.$$

7.2. Convenient sampling. We now ask the following question: given matrices $S_1, \dots, S_r$ defining a complete discrete sampling, what probabilities p_i should we assign to S_i so that the convergence rate ρ becomes *easy to interpret*? The following result, first stated in [17] in the context of solving linear systems, gives a *convenient* choice of probabilities resulting in ρ which depends on a (scaled) condition number of A.

PROPOSITION 7.4. *Let S be a complete discrete sampling, where $S = S_i$ with probability*

$$p_i = \|W^{1/2} A^T S_i\|_F^2 \Big/ \|W^{1/2} A^T \mathbf{S}\|_F^2. \tag{7.7}$$

Then the convergence rate takes the form

$$\rho = 1 - 1/\kappa_{2,F}^2(W^{1/2} A^T \mathbf{S}), \qquad \textit{where} \tag{7.8}$$

(7.9)

$$\kappa_{2,F}(W^{1/2} A^T \mathbf{S}) \overset{def}{=} \|(W^{1/2} A^T \mathbf{S})^{-1}\|_2 \|W^{1/2} A^T \mathbf{S}\|_F = \sqrt{\frac{\mathbf{Tr}(\mathbf{S}^T A W A^T \mathbf{S})}{\lambda_{\min}(\mathbf{S}^T A W A^T \mathbf{S})}} \geq \sqrt{n}.$$

Proof. Theorem 5.1 in [17] gives (7.8). The bound in (7.9) follows trivially. ∎

Following from Remark 4.1, we can determine a convergence rate for Algorithm 2 based on Theorem 7.4.

Remark 7.1. Let S be a complete discrete sampling where $S = S_i$ with probability

$$p_i = \|W^{1/2} A S_i\|_F^2 \Big/ \|W^{1/2} A \mathbf{S}\|_F^2. \tag{7.10}$$

Then Algorithm 2 converges at the rate

$$\rho_2 = 1 - 1/\kappa_{2,F}^2(W^{1/2} A \mathbf{S}). \tag{7.11}$$

7.3. Optimal and adaptive samplings. Having decided on the probabilities $p_1, \dots, p_r$ associated with the matrices $S_1, \dots, S_r$ in Proposition 7.4, we can now ask the following question: How should we choose the matrices $\{S_i\}$ if we want ρ to be as small as possible? Since the rate of convergence improves as the *condition number* $\kappa_{2,F}^2(W^{1/2} A^T \mathbf{S})$ decreases, we should aim for matrices that minimize the condition number. Notice that the lower bound in (7.9) is reached for $\mathbf{S} = (W^{1/2} A^T)^{-1} =$

$A^{-T}W^{-1/2}$. While we do not know A^{-1}, we can use our best current approximation of it, X_k, in its place. This leads to a method which *adapts* the probability distribution governing S throughout the iterative process. This observation inspires a very efficient modification of Algorithm 3, which we call AdaRBFGS (adaptive randomized BFGS) and describe in section 9. Notice that, luckily and surprisingly, our twin goals of computing the inverse and optimizing the convergence rate via the above adaptive trick are compatible. Indeed, we wish to find A^{-1}, whose knowledge gives us the optimal rate. This should be contrasted with the SDP approach mentioned earlier: (i) the SDP could potentially be harder than the inversion problem, and (ii) having found the optimal probabilities $\{p_i\}$, we are still not guaranteed the optimal rate. Indeed, optimality is relative to the choice of the matrices $S_1, \ldots, S_r$, which can be suboptimal.

Remark 7.2 (adaptive sampling). The convergence rate (7.8) suggests how one can select a sampling distribution for S that would result in faster practical convergence. We now detail several practical choices for W and indicate how to sample S. These suggestions require that the distribution of S depend on the iterate X_k and thus no longer fit into our framework. Nonetheless, we collect these suggestions here in the hope that others will wish to extend these ideas further, and as a demonstration of the utility of developing convergence rates.

(a) If $W = I$, then Algorithm 1 converges at the rate $\rho = 1 - 1/\kappa_{2,F}^2(A^T\mathbf{S})$, and hence S should be chosen so that $\mathbf{S}$ is a preconditioner of A^T. For example, $\mathbf{S} = X_k^T$; that is, S should be a sampling of the rows of X_k.

(b) If $W = I$, then Algorithm 2 converges at the rate $\rho = 1 - 1/\kappa_{2,F}^2(A\mathbf{S})$, and hence S should be chosen so that $\mathbf{S}$ is a preconditioner of A. For example, $\mathbf{S} = X_k$; that is, S should be a sampling of the columns of X_k.

(c) If A is symmetric positive definite, we can choose $W = A^{-1}$, in which case Algorithm 3 converges at the rate $\rho = 1 - 1/\kappa_{2,F}^2(A^{1/2}\mathbf{S})$. This rate suggests that S should be chosen so that $\mathbf{S}$ is an approximation of $A^{-1/2}$. In section 9 we develop this idea further and design the AdaRBFGS algorithm.

(d) If $W = (A^TA)^{-1}$, then Algorithm 1 can be efficiently implemented with $S = AV$, where V is a complete discrete sampling. Furthermore, $\rho = 1 - 1/\kappa_{2,F}^2(A\mathbf{V})$, where $\mathbf{V} = [V_1, \ldots, V_r]$. This rate suggests that V should be chosen so that $\mathbf{V}$ is a preconditioner of A. For example, we can set $\mathbf{V} = X_k$; i.e., V should be a sampling of the rows of X_k.

(e) If $W = (AA^T)^{-1}$, then Algorithm 2 can be efficiently implemented with $S = A^TV$, where V is a complete discrete sampling. From Remark 7.1, the convergence rate of the resulting method is given by $1 - 1/\kappa_{2,F}^2(A^T\mathbf{V})$. This rate suggests that V should be chosen so that $\mathbf{V}$ is a preconditioner of A^T. For example, $\mathbf{V} = X_k^T$; that is, V should be a sampling of the columns of X_k.

(f) If A is symmetric positive definite, we can choose $W = A^2$, in which case Algorithm 3 can be efficiently implemented with $S = AV$. Furthermore, $\rho = 1 - 1/\kappa_{2,F}^2(A\mathbf{V})$. This rate suggests that V should be chosen so that $\mathbf{V}$ is a preconditioner of A. For example, $\mathbf{V} = X_k$; that is, V should be a sampling of the rows or the columns of X_k.

8. Randomized quasi-Newton updates. Algorithms 1, 2, and 3 are in fact families of algorithms indexed by the two parameters (i) positive definite matrix W and (ii) distribution $\mathcal{D}$ (from which we pick random matrices S). This allows us to design a myriad of specific methods by varying these parameters.

Table 8.1 highlights some of these possibilities, focusing on complete discrete distributions for S so that convergence of the iterates is guaranteed through Theorems 6.4 and 6.5. These possibilities include new interpretations and connections to existing qN methods and AIP methods. With the exception of the *randomized BFGS method*, details on each instantiation described in Table 8.1 have been omitted here but can be found in the preprint [18].

TABLE 8.1

Specific randomized updates for inverting matrices discussed in this section, obtained as special cases of our algorithms. First column: "sym" means "symmetric" and "s.p.d." means "symmetric positive definite." Third column: "inv" means invertible. Block versions of all these updates are obtained by choosing S as a matrix with more than one column.

A	W	S	Inverse equation	Randomized update
any	any	inv.	any	one step
any	I	e_i	$AX = I$	simultaneous Kaczmarz (SK)
any	I	vector	$XA = I$	bad Broyden (BB)
sym.	I	vector	$AX = I, X = X^T$	Powell-symmetric-Broyden (PSB)
any	I	vector	$XA^{-1} = I$	good Broyden (GB)
s.p.d.	A	vector	$XA^{-1} = I, X = X^T$	Davidon–Fletcher–Powell (DFP)
s.p.d.	A^{-1}	vector	$AX = I, X = X^T$	Broyden–Fletcher–Goldfarb–Shanno (BFGS)
any	$(A^TA)^{-1}$	vector	$AX = I$	column

Next we delve into the details of the *randomized BFGS method* and focus the rest of the paper on developing adaptive sketches for this method and performing numerical experiments on the resulting adaptive method. We focus on the randomized BFGS method because, as we will show, it is remarkably efficient at obtaining approximations to the inverse of positive definite matrices. As such, it can be used as the building block for new iterative preconditioning techniques for positive definite systems and new qN methods.

Developing practical methods based on the remaining instantiations in Table 8.1 should follow the same pattern as we present in the following sections for developing an efficient method based on the randomized BFGS method. That is, first one should design a sketching matrix aimed at improving the convergence rate of the method in question. If the application in mind has a particular structure, such as matrices with a certain sparsity pattern, one should also attempt to incorporate this structure as an explicit constraint within the sketch and project framework.

8.1. Randomized BFGS update. If A is symmetric and positive definite, we can choose $W = A^{-1}$ and apply Algorithm 3 to maintain symmetry of the iterates. The iterates are given by

$$X_{k+1} = S(S^TAS)^{-1}S^T + \left(I - S(S^TAS)^{-1}S^TA\right)X_k\left(I - AS(S^TAS)^{-1}S^T\right). \tag{8.1}$$

This is a block variant (see [15]) of the BFGS update [4, 10, 12, 35]. The constrain-and-approximate viewpoint gives a new interpretation to the block BFGS update. That is, from (5.1), the iterates (8.1) can be equivalently defined by

$$X_{k+1} = \arg_X \min_{X\in\mathbb{R}^{n\times n}, Y\in\mathbb{R}^{n\times q}} \tfrac{1}{2}\left\|X - A^{-1}\right\|^2_{F(A)} \quad \text{subject to} \quad X = X_k + SY^T + YS^T.$$

Thus, the standard and the block BFGS updates obtain an improved estimate of the inverse by calculating the best approximation to the inverse subject to a particular symmetric affine space passing through the current iterate. This is a new way of interpreting BFGS.

If $p_i = \mathbf{Tr}(S_i^T A S_i)/\mathbf{Tr}(\mathbf{S}A\mathbf{S}^T)$, then according to Proposition 7.4, the update (8.1) converges according to

$$\mathbf{E}\left[\left\|X_k - A^{-1}\right\|_{F(A)}^2\right] \leq \left(1 - \tfrac{1}{\kappa_{2,F}^2(A^{1/2}\mathbf{S})}\right)^k \left\|X_0 - A^{-1}\right\|_{F(A)}^2. \tag{8.2}$$

Curiously, this is exactly the same rate at which the coordinate descent method converges when applied to solving linear systems; see [25] and section 3.4 in [16].

One of the most remarkable properties of the update (8.1) is that it preserves positive definiteness of the iterates, as we prove in the following lemma.

LEMMA 8.1. *If S is a complete discrete sampling and $X_0 \in \mathbb{R}^{n\times n}$ and $A \in \mathbb{R}^{n\times n}$ are positive definite matrices, then the iterates* (8.1) *are positive definite matrices.*

Proof. By induction, assume that X_k is positive definite, and let $v \in \mathbb{R}^n$ and $P \overset{\text{def}}{=} S(S^T AS)^{-1}S^T$. Left and right multiplying (8.1) by v^T and v, respectively, gives

$$v^T X_{k+1} v = v^T P v + v^T (I - PA) X_k (I - AP) v \geq 0.$$

Thus $v^T X_{k+1} v = 0$ implies that $Pv = 0$ and $(I - AP)v = 0$, which when combined gives $v = 0$. This proves that X_{k+1} is positive definite. ∎

The above lemma is of particular importance since it shows that the update (8.1) is well suited for estimating the inverse of a positive definite matrix. As such, the randomized BFGS method can be used to precondition positive definite systems, such as those arising in Newton-type optimization methods. Furthermore, the update (8.1) can be used to design new qN-type methods.

In section 9, we detail an update designed to improve the convergence rate in (8.2). The result is a method that is able to estimate the inverse of large scale positive definite matrices orders of magnitude faster than the benchmark methods.

9. AdaRBFGS: Adaptive randomized BFGS. All the updates we have developed thus far use a sketching matrix S that is sampled in an i.i.d. fashion from a fixed distribution $\mathcal{D}$ at each iteration. In this section we assume that A is symmetric positive definite, and propose AdaRBFGS, a variant of the RBFGS update, discussed in section 8.1, which *adaptively* changes the distribution $\mathcal{D}$ throughout the iterative process. Due to this change, Theorems 6.4 and 6.5 and Proposition 7.4 are no longer applicable. Superior numerical efficiency of this update is verified through extensive numerical experiments in section 10.

9.1. Motivation. We now motivate the design of this new update by examining the convergence rate (8.2) of the RBFGS iterates (8.1). Recall that in RBFGS we choose $W = A^{-1}$ and $S = S_i$ with probability

$$p_i = \mathbf{Tr}(S_i^T A S_i)/\mathbf{Tr}(\mathbf{S}A\mathbf{S}^T), \quad i = 1, 2, \ldots, r, \tag{9.1}$$

where S is a complete discrete sampling and $\mathbf{S} = [S_1, \ldots, S_r]$; then

$$\rho = 1 - 1/\kappa_{2,F}^2(A^{1/2}\mathbf{S}) \overset{(7.9)}{=} 1 - \lambda_{\min}(\mathbf{S}^T A\mathbf{S})/\mathbf{Tr}(\mathbf{S}^T A\mathbf{S}).$$

Consider now the question of choosing the matrix $\mathbf{S}$ in such a way that ρ is as small as possible. Note that the optimal choice is any $\mathbf{S}$ such that $\mathbf{S}^T A\mathbf{S} = I$. Indeed, then $\rho = 1 - 1/n$, and the lower bound (7.9) is attained. For instance, the choice $\mathbf{S} = A^{-1/2}$ would be optimal. Hence, in each iteration we would choose S to be a random column

(or random column submatrix) of $A^{-1/2}$. Clearly, this is not a feasible choice, as we do not know the inverse of A. In fact, it is A^{-1} which we are trying to find! However, this leads to the following observation: *the goals of finding the inverse of A and of designing an optimal distribution $\mathcal{D}$ are in synchrony.*

9.2. The algorithm. While we do not know $A^{-1/2}$, we can use the iterates $\{X_k\}$ themselves to construct a good *adaptive* sampling. Indeed, the iterates contain information about the inverse, and hence we can use them to design a better sampling S. In order to do so, it will be useful to maintain a factored form of the iterates,

$$X_k = L_k L_k^T, \tag{9.2}$$

where $L_k \in \mathbb{R}^{n\times n}$ is invertible. With this in place, let us choose S to be a random column submatrix of L_k. In particular, let $C_1, C_2, \ldots, C_r$ be nonempty subsets of $[n] = \{1, 2, \ldots, n\}$ forming a partition of $[n]$, and at iteration k choose

$$S = L_k I_{:C_i} \overset{\text{def}}{=} S_i, \tag{9.3}$$

with probability p_i given by (9.1) for $i = 1, 2, \ldots, r$. For simplicity, assume that $C_1 = \{1, \ldots, c_1\}$, $C_2 = \{c_1 + 1, \ldots, c_2\}$, and so on, so that, by the definition of $\mathbf{S}$,

$$\mathbf{S} = [S_1, \ldots, S_r] = L_k. \tag{9.4}$$

Note that now both $\mathbf{S}$ and p_i depend on k. The method described above satisfies the following recurrence.

THEOREM 9.1. *After one step of the AdaRBFGS method we have*

$$\mathbf{E}\left[\left\|X_{k+1} - A^{-1}\right\|^2_{F(A)} \mid X_k\right] \leq \left(1 - \tfrac{\lambda_{\min}(AX_k)}{\mathbf{Tr}(AX_k)}\right)\left\|X_k - A^{-1}\right\|^2_{F(A)}. \tag{9.5}$$

Proof. Using the same arguments as those in the proof of Theorem 6.5, we obtain

$$\mathbf{E}\left[\left\|X_{k+1} - A^{-1}\right\|^2_{F(A)} \mid X_k\right] \leq (1 - \rho_k)\left\|X_k - A^{-1}\right\|^2_{F(A)}, \tag{9.6}$$

where $\rho_k \overset{\text{def}}{=} \lambda_{\min}\left(A^{-1/2}\mathbf{E}\left[Z \mid X_k\right]A^{-1/2}\right)$ and

$$Z \overset{(9.7)}{=} AS_i(S_i^T AS_i)^{-1}S_i^T A. \tag{9.7}$$

So, we only need to show that $\rho_k \geq \lambda_{\min}(AX_k)/\mathbf{Tr}(AX_k)$. Since S is a complete discrete sampling, Proposition 7.2 applied to our setting says that

$$\mathbf{E}\left[Z \mid X_k\right] = A\mathbf{S}D^2\mathbf{S}^T A, \qquad \text{where} \tag{9.8}$$

$$D \overset{\text{def}}{=} \operatorname{Diag}\left(\sqrt{p_1}(S_1^T AS_1)^{-1/2}, \ldots, \sqrt{p_r}(S_r^T AS_r)^{-1/2}\right). \tag{9.9}$$

We now have

$$\rho_k \overset{(9.8)+(9.4)}{\geq} \lambda_{\min}\left(A^{1/2}L_k L_k^T A^{1/2}\right)\lambda_{\min}(D^2) \overset{(9.2)}{=} \frac{\lambda_{\min}(AX_k)}{\lambda_{\max}(D^{-2})}$$
$$\overset{(9.9)}{=} \frac{\lambda_{\min}(AX_k)}{\max_i \lambda_{\max}(S_i^T AS_i)/p_i} \geq \frac{\lambda_{\min}(AX_k)}{\max_i \mathbf{Tr}(S_i^T AS_i)/p_i} \overset{(9.1)+(9.4)}{=} \frac{\lambda_{\min}(AX_k)}{\mathbf{Tr}(AX_k)}.$$

In the second equality we have used the fact that the largest eigenvalue of a block diagonal matrix is equal to the maximum of the largest eigenvalues of the blocks. ∎

If X_k converges to A^{-1}, then necessarily the one-step rate of AdaRBFGS proved in Theorem 9.1 asymptotically reaches the lower bound

$$\rho_k \overset{\text{def}}{=} 1 - \frac{\lambda_{\min}(AX_k)}{\mathbf{Tr}(AX_k)} \to 1 - \frac{1}{n}.$$

In other words, as long as this method works, the convergence rate gradually improves and becomes asymptotically optimal and independent of the condition number. We leave a deeper analysis of this and other adaptive variants of the methods developed in this paper to future work.

9.3. Implementation. To implement the AdaRBFGS update, we need to maintain the iterates X_k in the factored form (9.2). Fortunately, a factored form of the update (8.1) was introduced in [19] which we shall now describe and adapt to our objective. Assuming that X_k is symmetric positive definite such that $X_k = L_k L_k^T$, we shall describe how to obtain a corresponding factorization of X_{k+1}. Letting $G_k = (S^T L_k^{-T} L_k^{-1} S)^{1/2}$ and $R_k = (S^T A S)^{-1/2}$, it can be verified through direct inspection [19] that $X_{k+1} = L_{k+1} L_{k+1}^T$, where

$$L_{k+1} = L_k + S R_k \left(G_k^{-1} S^T L_k^{-T} - R_k^T S^T A L_k \right). \tag{9.10}$$

If we instead of (9.3) consider the more general update $S = L_k \tilde{S}$, where $\tilde{S}$ is chosen in an i.i.d. fashion from some fixed distribution $\tilde{\mathcal{D}}$, then

$$L_{k+1} = L_k + L_k \tilde{S} R_k \left((\tilde{S}^T \tilde{S})^{-1/2} \tilde{S}^T - R_k^T \tilde{S}^T L_k^T A L_k \right). \tag{9.11}$$

The above can now be implemented efficiently; see Algorithm 4.

Algorithm 4. Adaptive randomized BFGS (AdaRBFGS).

input: symmetric positive definite matrix A
parameter: $\tilde{\mathcal{D}}$ = distribution over random matrices with n rows
initialize: pick invertible $L_0 \in \mathbb{R}^{n \times n}$
for $k = 0, 1, 2, \ldots$ **do**
  Sample an independent copy $\tilde{S} \sim \tilde{\mathcal{D}}$
  Compute $S = L_k \tilde{S}$ ▷ S is sampled adaptively, as it depends on k
  Compute $R_k = (\tilde{S}^T A \tilde{S})^{-1/2}$
  $L_{k+1} = L_k + S R_k \left((\tilde{S}^T \tilde{S})^{-1/2} \tilde{S}^T - R_k^T S^T A L_k \right)$ ▷ Update the factor
output: $X_k = L_k L_k^T$

In section 10 we test two variants based on (9.11). The first is the *AdaRBFGS_gauss* update, in which the entries of $\tilde{S}$ are standard Gaussian. The second is *AdaRBFGS_cols*, where $\tilde{S} = I_{:C_i}$, as described above, and $|C_i| = q$ for all i and for some q.

10. Numerical experiments. Given the demand for approximate inverses of positive definite matrices in preconditioning and in variable metric optimization methods, we restrict our tests to inverting positive definite matrices. The experiments from this section can be replicated by downloading the package `InvRand` from the authors' webpage http://www.di.ens.fr/~rgower/software/ together with scripts that allow for the tests we describe here to be easily reproduced.

We test four iterative methods for inverting matrices and one direct method. For our tests we use two variants of Algorithm 4: AdaRBFGS_gauss, where $\tilde{S} \in \mathbb{R}^{n \times q}$ is

a normal Gaussian matrix, and AdaRBFGS_cols, where $\tilde{S}$ consists of a collection of q distinct coordinate vectors in $\mathbb{R}^n$, selected uniformly at random. At each iteration the AdaRBFGS methods compute the inverse of a small matrix S^TAS of dimension $q \times q$. To invert this matrix we use the MATLAB inbuilt `inv` function, which uses LU decomposition or Gaussian elimination, depending on the input. Either way, `inv` costs $O(q^3)$. For simplicity, we selected $q = \sqrt{n}$ in all our tests.

We compare our method to two iterative methods, the *Newton–Schulz method* [34] and the global self-conditioned *minimal residual* (MR) method [6]. The Newton–Schulz method arises from applying the Newton–Raphson method to solve the equation $X^{-1} = A$, which gives

$$X_{k+1} = 2X_k - X_kAX_k. \tag{10.1}$$

The MR method was designed to calculate approximate inverses, and it does so by minimizing the norm of the residual along the preconditioned residual direction:

$$\|I - AX_{k+1}\|_F^2 = \min_{\alpha\in\mathbb{R}} \left\{ \|I - AX\|_F^2 \quad \text{s.t.} \quad X = X_k + \alpha X_k(I - AX_k) \right\}. \tag{10.2}$$

See [33, Chapter 10.5] for a didactic introduction to MR methods. Letting $R_k = I - AX_k$, the resulting iterates of the MR method are given by

$$X_{k+1} = X_k + \frac{\mathbf{Tr}(R_k^T AX_kR_k)}{\mathbf{Tr}((AX_kR_k)^T AX_kR_k)} X_kR_k. \tag{10.3}$$

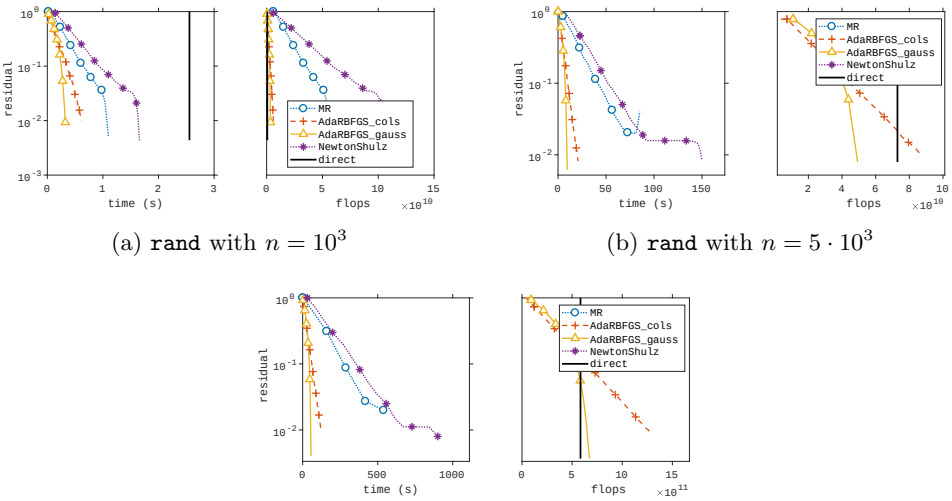
We also compare our methods to a *direct* inversion method by using Cholesky decomposition together with forward substitution triangular inversion; see Method 2 in [7].

We perform two sets of tests. For the first set, we choose a different starting matrix for each method which is optimized, in some sense, for that method. We then compare the empirical convergence of each method, including the time taken to calculate X_0. In particular, the Newton–Schulz method is only guaranteed to converge for an initial matrix X_0 such that $\rho(I - X_0A) < 1$. Indeed, the Newton–Schulz method did not converge in most of our experiments when X_0 was not carefully chosen according to this criteria. To remedy this, we choose $X_0 = 0.99 \cdot A^T/\rho^2(A)$ for the Newton–Schulz method, so that $\rho(I - X_0A) < 1$ is satisfied. To compute $\rho(A)$ we used the inbuilt MATLAB function `normest` which is coded in C++. For MR we followed the suggestion in [33] and used the projected identity for the initial matrix $X_0 = (\mathbf{Tr}(A)/\mathbf{Tr}(AA^T)) \cdot I$. For our AdaRBFGS methods we simply used $X_0 = I$, as this worked well in practice.

Further tests where we compare the empirical convergence of the methods starting from the same matrix, namely the identity matrix $X_0 = I$, can be found in [18].

We run each method until the relative error $\|I - AX_k\|_F / \|I - AX_0\|_F$ is below 10^{-2}. All experiments were performed and run in MATLAB R2014b. To appraise the performance of each method we plot the relative error against time taken and against the number of *floating point operations* (flops).

10.1. Experiment 1: Synthetic matrices. First we compare the four methods on synthetic matrices generated using the matrix function `rand`. To appraise the difference in performance of the methods as the dimension of the problem grows, we tested for $n = 1000$, 5000, and $10'000$. As the dimension grows, only the two variants of the AdaRBFGS method are able to reach the 10^{-2} desired tolerance in a reasonable amount of time and number of flops (see Figure 10.1). In particular, in



(a) `rand` with $n = 10^3$ (b) `rand` with $n = 5 \cdot 10^3$

(c) `rand` with $n = 10^4$

FIG. 10.1. *Synthetic problems with a uniform random matrix* $A = \bar{A}^T \bar{A}$, *where* $\bar{A} =$`rand`(n).

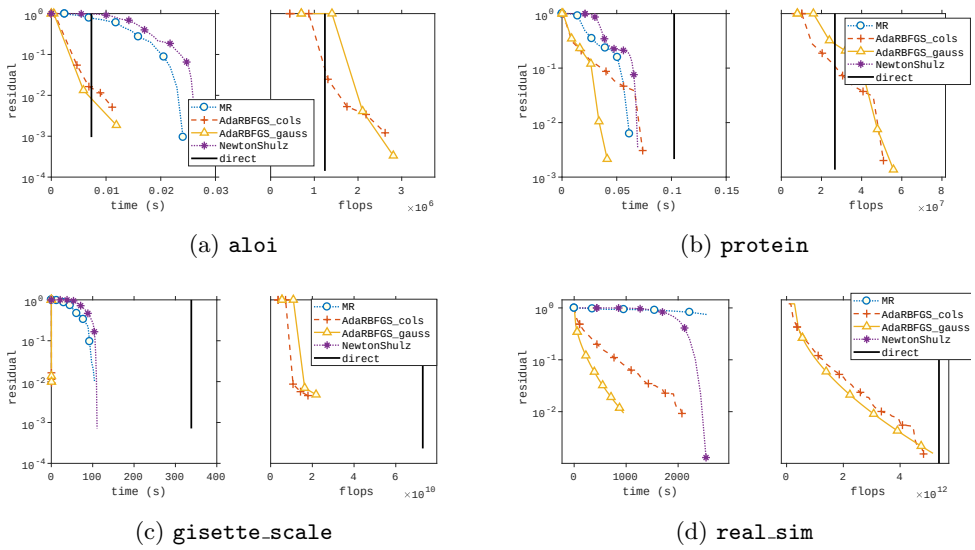
Figure 10.1(a) we see that the direct method, represented by a black vertical line, was the slowest in time yet the fastest in the number of flops. In Figures 10.1(b) and (c) we removed the direct method from the time plot since it took 327.29s and 2526.1s, respectively, which was considerably slower than the iterative methods, thus making the figures hard to read. Furthermore, in Figures 10.1(a), (b), and (c) we see that the gap in time and flops taken between the MR and the Newton–Shulz methods and the AdaRBFGS methods grows as the dimension of the problem grows. Indeed, we have had to remove MR and Newton–Shulz from the flops plot in Figures 10.1(b) and (c) since they were orders of magnitude greater.

10.2. Experiment 2: LIBSVM matrices. Next we invert the Hessian matrix $\nabla^2 f(x)$ of four ridge-regression problems of the form

$$\min_{x \in \mathbb{R}^n} f(x) \stackrel{\text{def}}{=} \tfrac{1}{2} \|Ax - b\|_2^2 + \tfrac{\lambda}{2} \|x\|_2^2, \qquad \nabla^2 f(x) = A^T A + \lambda I, \tag{10.4}$$

using data from LIBSVM [5]; see Figure 10.2. We use $\lambda = 1$ as the regularization parameter. On the two problems of smaller dimension, `aloi` and `protein`, the five methods have a similar performance and encounter the inverse in less than one second. On the two larger problems, `gisette-scale` and `real-sim`, the two variants of AdaRBFGS significantly outperform the MR method, the Newton–Schulz method, and the direct method. Note that we have once again removed the MR and the Newton–Schulz methods from the flops plots since they were distorting the plot with their exorbitant number of flops taken. We have also omitted the 23460s (6.51 hours) taken for the direct method in the `real_sim` problem since it dwarfed the other time plots.

10.3. Experiment 3: UF sparse matrices. For our final batch of tests, we invert several sparse matrices from the University of Florida sparse matrix collection [9]. We have selected six problems from six different applications, so that the set of matrices display a varied sparsity pattern and structure; see Figure 10.3.



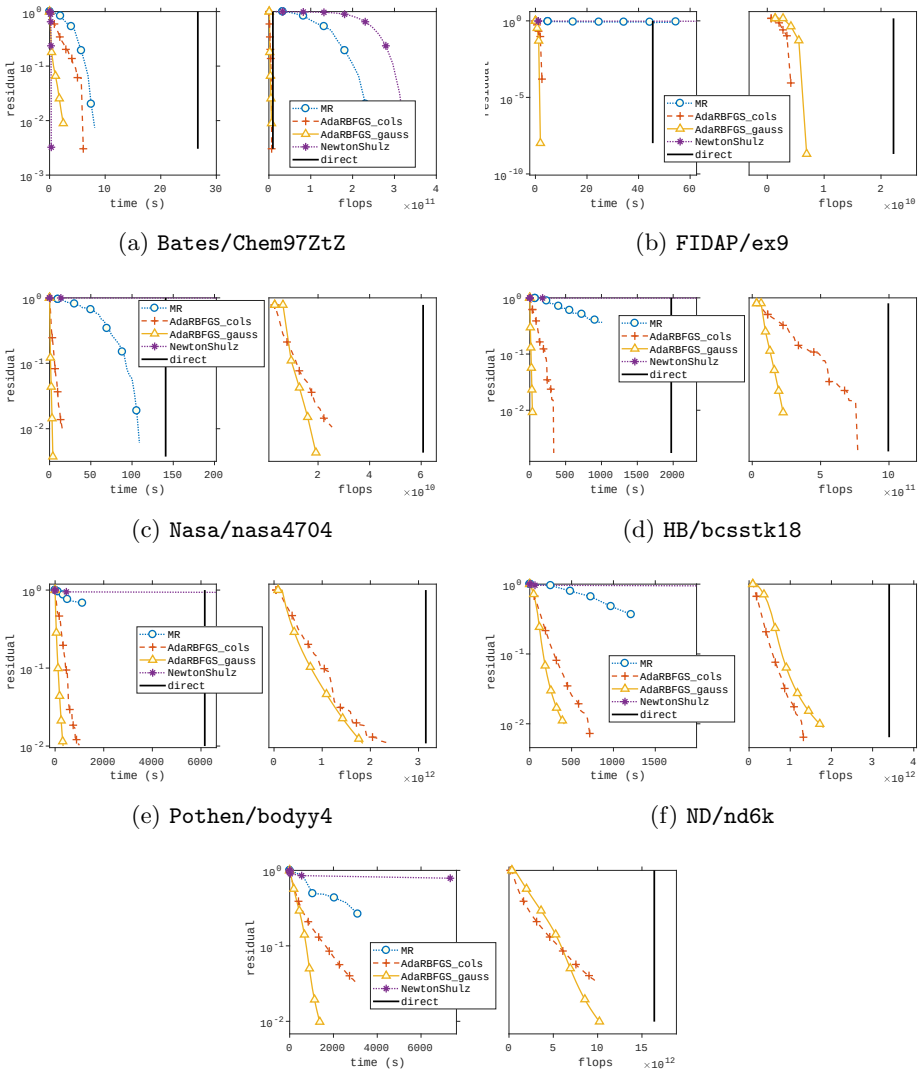
(a) `aloi` (b) `protein`

(c) `gisette_scale` (d) `real_sim`

FIG. 10.2. *The performance of Newton–Schulz, MR, AdaRBFGS_gauss, and AdaRBFGS_cols methods on the Hessian matrix of four LIBSVM test problems:* (a) `aloi`: $(m;n) = (108,000;128)$, (b) `protein`: $(m;n) = (17,766;357)$, (c) `gisette_scale`: $(m;n) = (6000;5000)$, (d) `real-sim`: $(m;n) = (72,309;20,958)$.

On the matrix `Bates/Chem97ZtZ` of moderate size, the four iterative methods perform comparably well, with the Newton–Schulz method converging first in time and AdaRBFGS_cols first in flops. The direct method took just over double the amount of time of the slowest iterative method to calculate an exact inverse. On the matrices of larger dimensions, the two variants of AdaRBFGS converge often orders of magnitude faster, and the gap between the direct method and the iterative methods also grows. Indeed, on the two largest problems `ND/nd6k` and `GHS_psdef/wathen100` the time taken by the direct method to terminate was 7.14 hours and 9.09 hours, which was 24 and 50 times slower, respectively, than the AdaRBFGS_gauss method. We have also removed the flop counts of the MR and Newton–Schulz methods from Figures 10.3(b), (c), (d), (e), (f), and (g) since they took orders of magnitude more flops as compared to the other methods. Finally, only the AdaRBFGS_gauss method reached the desired residual tolerance on the `GHS_psdef/wathen100` problem within the 2000 seconds schedule allocated to each method. This is in stark contrast to the Newton–Schulz and MR methods, which made negligible progress, even after 7334.9 seconds (2.03 hours) of time.

The significant difference between the performance of the iterative methods on large scale problems can be explained in part by their iteration cost. The iterates of the Newton–Schulz and MR methods compute $n \times n$ matrix-matrix products. Meanwhile the cost of an iteration of the AdaRBFGS methods is dominated by the cost of an $n \times n$ matrix by an $n \times q$ matrix product. As a result, and because we set $q = \sqrt{n}$, this is a difference of n^3 to $n^{2+1/2}$ in iteration cost, which clearly shows on the larger dimensional instances. On the other hand, both the Newton–Schulz and MR methods are quadratically locally convergent; thus when the iterates are close to the solution, these methods enjoy a notable speed-up. This can be seen in Figures 10.2(c) and (d), where the Newton–Schulz method is clearly converging at a superlinear/quadratic



(a) Bates/Chem97ZtZ (b) FIDAP/ex9

(c) Nasa/nasa4704 (d) HB/bcsstk18

(e) Pothen/bodyy4 (f) ND/nd6k

(g) GHS_psdef/wathen100

FIG. 10.3. *The performance of Newton–Schulz, MR, AdaRBFGS_gauss, and AdaRBFGS_cols on* (a) Bates-Chem97ZtZ: $n = 2,541$, (b) FIDAP/ex9: $n = 3,363$, (c) Nasa/nasa4704: $n = 4,704$, (d) HB/bcsstk18: $n = 11,948$, (e) Pothen/bodyy4: $n = 17,546$, (f) ND/nd6k: $n = 18,000$, (g) GHS_psdef/wathen100: $n = 30,401$.

rate. Thus, if we required a high precision inverse, the Newton–Schulz and MR methods would most likely converge before the AdaRBFGS methods.

11. Conclusion. We developed a family of stochastic methods for iteratively inverting matrices, with specialized variants for nonsymmetric, symmetric, and positive definite matrices. The methods have dual viewpoints: a sketch-and-project viewpoint (which is an extension of the least-change formulation of the qN methods) and a constrain-and-approximate viewpoint (which is related to the approximate inverse preconditioning (API) methods). The equivalence between these viewpoints reveals

a deep connection between the qN and the API methods, which were previously considered to be unrelated.

Under mild conditions, we establish convergence of the expected norm of the error and the norm of the expected error. Our convergence theorems are general enough to accommodate discrete samplings and continuous samplings, though we only explore discrete samplings here in more detail. For discrete samplings, we determine a probability distribution for which the convergence rates are equal to a scaled condition number and thus are easily interpretable. Furthermore, for discrete sampling, we determine a practical optimized sampling distribution, which is obtained by minimizing an upper bound on the convergence rate. We develop new randomized block variants of the qN updates, including the BFGS update, complete with convergence rates, and provide new insights into these methods using our dual viewpoint.

For positive definite matrices, we develop an adaptive randomized BFGS method (AdaRBFGS), which in large scale numerical experiments can be orders of magnitude faster (in time and flops) than the self-conditioned minimal residual (MR) method and the Newton–Schulz method. In particular, only the AdaRBFGS methods are able to approximately invert the $20{,}958 \times 20{,}958$ ridge regression matrix based on the `real-sim` data set in reasonable time and flops.

This paper opens up many avenues for future work, such as developing methods that use continuous random sampling and implementing a limited memory approach akin to the LBFGS (limited memory BFGS) [29] method, which could maintain an operator that serves as an approximation to the inverse. The same method can also be used to calculate the pseudoinverse. As recently shown in [16], an analogous method can be applied to linear systems, converging with virtually no assumptions on the system matrix.

REFERENCES

[1] M. Benzi and M. Tůma, *Comparative study of sparse approximate inverse preconditioners*, Appl. Numer. Math., 30 (1999), pp. 305–340.

[2] R. Bhatia, *Positive Definite Matrices*, Princeton Ser. Appl. Math., Princeton University Press, Princeton, NJ, 2008, p. 264.

[3] M. D. Bingham, *A new method for obtaining the inverse matrix*, J. Amer. Statist. Assoc., 36 (1941), pp. 530–534.

[4] C. G. Broyden, *A class of methods for solving nonlinear simultaneous equations*, Math. Comp., 19 (1965), pp. 577–593.

[5] C. C. Chang and C. J. Lin, *LIBSVM: A library for support vector machines*, ACM Trans. Intel. Syst. Tech., 2 (2011), pp. 1–27.

[6] E. Chow and Y. Saad, *Approximate inverse preconditioners via sparse-sparse iterations*, SIAM J. Sci. Comput., 19 (1998), pp. 995–1023, https://doi.org/10.1137/S1064827594270415.

[7] J. J. D. Croz and N. J. Higham, *Stability of methods for matrix inversion*, IMA J. Numer. Anal., 12 (1992), pp. 1–19.

[8] W. C. Davidon, *Variable Metric Method for Minimization*, Tech. rep., A.E.C. Research and Development Report, ANL-5990, Argonne National Laboratory, Lemont, IL, 1959.

[9] T. A. Davis and Y. Hu, *The University of Florida sparse matrix collection*, ACM Trans. Math. Software, 38 (2011), 1.

[10] B. R. Fletcher and M. J. D. Powell, *A rapidly convergent descent method for minimization*, Comput. J., 6 (1963), pp. 163–168.

[11] D. Goldfarb, *Modification methods for inverting matrices and solving systems of linear algebraic equations*, Math. Comp., 26 (1972), pp. 829–829.

[12] D. Goldfarb, *A family of variable-metric methods derived by variational means*, Math. Comp., 24 (1970), pp. 23–26.

[13] N. I. M. Gould and J. A. Scott, *Sparse approximate-inverse preconditioners using norm-minimization techniques*, SIAM J. Sci. Comput., 19 (1998), pp. 605–625, https://doi.org/

10.1137/S1064827595288425.
[14] R. M. Gower, *Sketch and Project: Randomized Iterative Methods for Linear Systems and Inverting Matrices*, Ph.D. thesis, University of Edinburgh, Edinburgh, UK, 2016.
[15] R. M. Gower and J. Gondzio, *Action Constrained Quasi-Newton Methods*, preprint, https://arxiv.org/abs/1412.8045v1, 2014.
[16] R. M. Gower and P. Richtárik, *Stochastic Dual Ascent for Solving Linear Systems*, preprint, https://arxiv.org/abs/1512.06890, 2015.
[17] R. M. Gower and P. Richtárik, *Randomized iterative methods for linear systems*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 1660–1690, https://doi.org/10.1137/15M1025487.
[18] R. M. Gower and P. Richtárik, *Randomized Quasi-Newton Updates Are Linearly Convergent Matrix Inversion Algorithms*, preprint, https://arxiv.org/abs/1602.01768, 2016.
[19] S. Gratton, A. Sartenaer, and J. Tshimanga, *On a class of limited memory preconditioners for large scale linear systems with multiple right-hand sides*, SIAM J. Optim., 21 (2011), pp. 912–935, https://doi.org/10.1137/08074008.
[20] B. J. Greenstadt, *Variations on variable-metric methods*, Math. Comp., 24 (1969), pp. 1–22.
[21] A. Griewank, *Broyden updating, the good and the bad!*, Doc. Math., Extra Vol.: Optimization Stories (2012), pp. 301–315.
[22] P. Hennig, *Probabilistic interpretation of linear solvers*, SIAM J. Optim., 25 (2015), pp. 234–260, https://doi.org/10.1137/140955501.
[23] T. Huckle and A. Kallischko, *Frobenius norm minimization and probing for preconditioning*, Internat. J. Comput. Math., 84 (2007), pp. 1225–1248.
[24] L. Y. Kolotilina and A. Y. Yeremin, *Factorized sparse approximate inverse preconditionings I. Theory*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 45–58, https://doi.org/10.1137/0614004.
[25] D. Leventhal and A. S. Lewis, *Randomized methods for linear constraints: Convergence rates and conditioning*, Math. Oper. Res., 35 (2010), pp. 641–654.
[26] D. Leventhal and A. Lewis, *Randomized Hessian estimation and directional search*, Optimization, 60 (2011), pp. 329–345.
[27] W. Li and Z. Li, *A family of iterative methods for computing the approximate inverse of a square matrix and inner inverse of a non-square matrix*, Appl. Math. Comput., 215 (2010), pp. 3433–3442.
[28] Y. Lu, P. Dhillon, D. P. Foster, and L. Ungar, *Faster ridge regression via the subsampled randomized Hadamard transform*, in Proceedings of NIPS 2013, Advances in Neural Information Processing Systems 26, 2013, pp. 369–377.
[29] J. Nocedal, *Updating quasi-Newton matrices with limited storage*, Math. Comp., 35 (1980), pp. 773–782.
[30] M. Pilanci and M. Wainwright, *Randomized sketches of convex programs with sharp guarantees*, IEEE Trans. Inform. Theory, 61 (2015), pp. 5096–5115.
[31] M. Pilanci and M. J. Wainwright, *Iterative Hessian sketch: Fast and accurate solution approximation for constrained least-squares*, J. Mach. Learn. Res., 17 (2016), pp. 1–33.
[32] M. Pilanci and M. J. Wainwright, *Newton Sketch: A Linear-Time Optimization Algorithm with Linear-Quadratic Convergence*, preprint, https://arxiv.org/abs/1505.02250, 2015.
[33] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003, https://doi.org/10.1137/1.9780898718003.
[34] G. Schulz, *Iterative berechung der reziproken matrix*, ZAMM Z. Angew. Math. Mech., 13 (1933), pp. 57–59.
[35] D. F. Shanno, *Conditioning of quasi-Newton methods for function minimization*, Math. Comp., 24 (1971), pp. 647–656.
[36] S. U. Stich, C. L. Müller, and B. Gärtner, *Variable metric random pursuit*, Math. Program., 156 (2015), pp. 549–579.
[37] T. Strohmer and R. Vershynin, *A randomized Kaczmarz algorithm with exponential convergence*, J. Fourier Anal. Appl., 15 (2009), pp. 262–278.