SIAM/ASA J. Uncertainty Quantification
Vol. 11, No. 3, pp. 996–1024

Towards Practical Large-Scale Randomized Iterative Least Squares Solvers through Uncertainty Quantification[*]

Nathaniel Pritchard[†] and Vivak Patel[†]

Abstract. As the scale of problems and data used for experimental design, signal processing, and data assimilation grow, the oft-occurring least squares subproblems are correspondingly growing in size. As the scale of these least squares problems creates prohibitive memory movement costs for the usual incremental QR and Krylov-based algorithms, randomized least squares problems are garnering more attention. However, these randomized least squares solvers are difficult to integrate into application algorithms as their uncertainty limits practical tracking of algorithmic progress and reliable stopping. Accordingly, in this work, we develop theoretically rigorous, practical tools for quantifying the uncertainty of an important class of iterative randomized least squares algorithms, which we then use to track algorithmic progress and create a stopping condition. We demonstrate the effectiveness of our algorithm by solving a 0.78 TB least squares subproblem from the inner loop of incremental 4D-Var using only 195 MB of memory.

Key words. random sketching, linear systems, iterative methods, gradient estimation, stopping criterion, least-squares, coordinate descent, 4D-Var

MSC codes. 65F10, 65F25, 60F10, 62L12

DOI. 10.1137/22M1515057

1. Introduction.

Least squares problems are regularly solved as core subproblems in a variety of important algorithms for experimental design [13, 3], signal processing [26, 28], data assimilation [29, 9], and uncertainty quantification [30, 27]. Moreover, these least squares subproblems are growing in both the number of equations and the dimension of the unknown variables for two reasons: (1) improvements in technology have increased the permeation of higher-frequency sensors, which grows the volume of data being used and which, in turn, (usually) increases the number of equations in the least squares subproblem; and (2) the growing desire for more accurately simulating models (e.g., using finer meshes for partial differential equation models) increases the number of unknown variables in the least squares problems.

Unfortunately, the growth of least squares subproblems is a challenge for commonly used solvers. For instance, solving a least squares problem with many observations can be addressed in a memory-efficient manner using an incremental QR algorithm [17], so long as the resulting upper triangular term can be fit in memory. Unfortunately, if the number of unknowns is sufficiently larger, this least squares incremental QR algorithm will be unable to store and

[*] Received by the editors August 10, 2022; accepted for publication (in revised form) April 10, 2023; published electronically August 31, 2023.
https://doi.org/10.1137/22M1515057
Funding: This work was supported by UW-Madison WARF award AAD5914.

[†] Department of Statistics, University of Wisconsin - Madison, Madison, WI 53706 USA (npritchard@wisc.edu, vivak.patel@wisc.edu, vivakpatel.org).

manipulate the resulting upper triangular matrix without substantial slowdowns induced by memory movement costs. As another example, Krylov-based least squares solvers can also be efficiently deployed [10], so long as matrix-vector and matrix-transpose-vector products can be efficiently computed. Unfortunately, if the system is sufficiently large that it cannot be stored in memory, then Krylov-based least squares solvers are substantially slowed down also because of the memory movement costs needed to read in the matrix multiple times per iteration [15].

As these challenges to standard solvers are driven by size, randomized least squares solvers (e.g., iterative Hessian sketch [22] and generalized column subspace descent [20, 19, 21]) seem to be promising alternatives as they are able to compress the information in the original linear system to more manageable dimensions. However, such iterative randomized least squares solvers must first overcome a key practical challenge: as such solvers would be called repeatedly within an iterative algorithm, their solution accuracy must be controlled so as to ensure algorithmic efficiency. For example, in incremental 4D-Var [4], a least squares subproblem occurs at every iteration of the algorithm. Indeed, in the initial few iterations, the least squares subproblem only needs to be solved to low accuracy as this is usually enough to generate progress quickly, while later iterations will demand that the subproblem be solved to higher accuracy. Thus, achieving such control over the least squares subproblem solver's accuracy ensures the efficiency of the overall algorithm.

When it comes to solving least squares problems, controlling the solver's accuracy depends on tracking the progress of the iterations and defining clear stopping conditions, which are typically achieved by using the norm of the gradient of the least squares subproblem.[1] Unfortunately, the gradient of a large least squares problem is calculated by applying a very large matrix in both its original and transposed orientation to a vector—a procedure that is very costly because of its guaranteed violation of the principal of spatial locality for memory accesses [15] (excepting the case in which the matrix is symmetric). This issue is further exacerbated for a randomized solver: the gradient at an iterate of a randomized solver *is never explicitly calculated,* and, even if it were calculated occasionally for monitoring progress, it would be less reliable, as we now explain. As the iterates of the randomized solver are random, the gradient evaluated at these iterates inherits this randomness; thus, a wide range of gradient norm values would correspond to the same residual norm squared in the iterates (see the blue boxes in Figure 1), which results in the norm of the gradient being a poor reflection of the residual norm squared. To reiterate, the gradient norm is widely used for tracking and stopping least squares problems, but it is infeasible to calculate for large-scale problems, and is unreliable for randomized solvers.

In the class of sketching-based randomized solvers that we consider in this work, the infeasibility of calculating the entire gradient can be addressed by using the sketch of the gradient,[2] which *is efficiently and regularly calculated by this class of randomized solvers.*[3]

[1] In the case of this manuscript, the least squares problem we are considering is $\min_{x \in \mathbb{R}^n} \|Ax - b\|_B^2$, and thus the gradient is $g_k = A^\top B(Ax_k - b)$.

[2] We can mathematically represent the sketched gradient as $\tilde{g}_k = S_{k+1}^\top g_k$, where S_{k+1} is a random matrix satisfying properties to be discussed in section 2.

[3] While there are cases where this is not true, we generally accept the premise that randomly sketching a matrix can be efficiently calculated. For instance, the Fast Johnson–Lindenstrauss Transform leverages the fast Fourier transform to efficiently sketch a matrix [2]. As another example, a Gaussian sketch can be efficiently applied using emerging photonic hardware, e.g., lighton.ai.

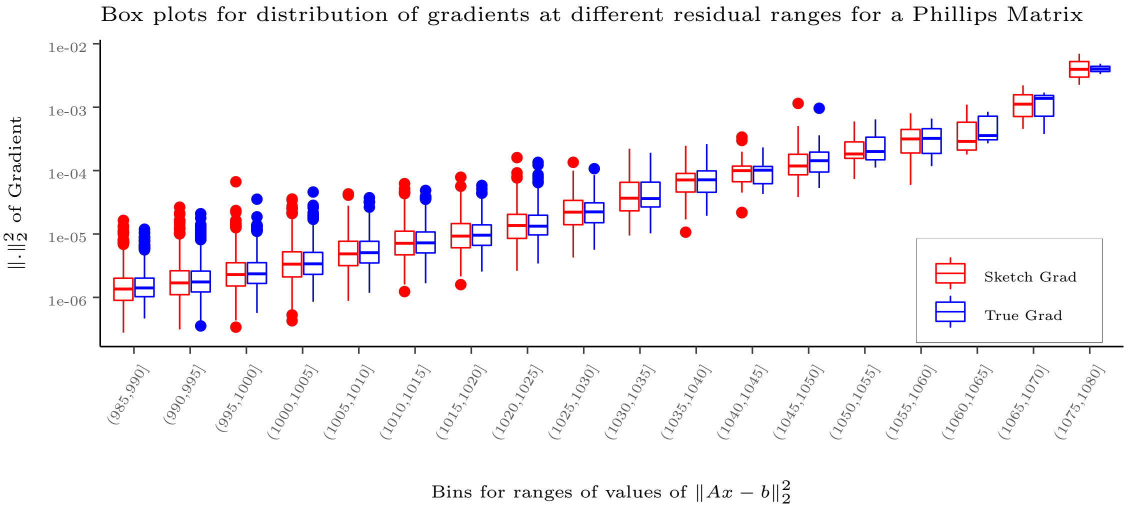

Figure 1. *We solve a Phillips linear system, which has a condition number of* $\mathcal{O}(10^9)$*, from Matrix Depot* [32] *using an iterative random sketching method. We compute the norm squared of the true and sketched gradients of the iterates as well as the norm squared of the residual of the iterates. The box plots show the distribution of gradient values for the norms squared of the sketched and true gradients at different intervals of residual norm squared values. For instance, the red box plot and blue box plot over* $(985, 990]$ *represent the distribution of the norms squared of the sketched and true gradients that correspond to residual norm squared values between* 985 *and* 990.

However, the sketched gradient norm inherits not only the randomness of the gradient at an iterate, but also the randomness from the sketching procedure. To see this, as shown by the red boxes in Figure 1, the sketched gradient norm has an even wider range for the same residual norm squared relative to the gradient norm. Thus, the sketched gradient norm, though feasibly calculated, is even less reliable for tracking and stopping an iterative randomized least squares solver.

While the sketched gradient norm alone is insufficient to reliably track and stop the underlying randomized least squares solver, if the sketched gradient norm's uncertainty could be quantified, then we could use this uncertainty set to create risk-informed[4] metrics for tracking and stopping the corresponding underlying algorithm. In this work, we develop a practical, computationally efficient method for quantifying the uncertainty set of the norm squared of the sketched gradient, and use it to develop risk-informed methods for tracking and stopping the underlying algorithm. In fact, we take this a step further by generalizing our method to a moving average of the sketched gradients, which turns out to be more reliable. We emphasize that our method, which requires only a small additional computational and memory cost over the solver, will accurately reflect the algorithm's progress based on a user-defined threshold for risk, and will stop the algorithm based on a user-defined threshold for risk. We demonstrate the power of our methodology by solving a 0.78 TB least squares subproblem arising from the incremental 4D-Var algorithm using only 195 MB of memory, for which LSQR is infeasible

[4] By risk-informed, we mean that the user can specify probabilities for which the tracking metrics and stopping conditions can fail.

(see subsection 4.4). As a result of our methodology, we are enabling the practical integration of an important class of randomized least squares solvers into algorithms that are widely used in science and engineering, which will support solving larger problems in these fields.

Note that in our previous work [23] we developed an analogous procedure for consistent linear systems. While at first glance these procedures seem identical owing to our effort to maintain notational consistency, the procedures and their analysis differ in two fundamental ways. First, the procedure and analysis in [23] relies on consistency, which is not available for the least squares problem. Because of consistency, the procedure in [23] can use left-sketching techniques, which are well studied [1, 2, 8, 25, 14, 20, 22]. Without consistency the procedure in [23] would fail to reflect the progress of the algorithm because left-sketching fails to adequately solve least squares problems [22, 24]. Hence, in this work, the procedure uses the less familiar right-sketching approach.

Second, as the procedure herein uses right-sketching, we must analyze the procedure using arguments about the column space, rather than arguments about the row space as in [23]. Thus, while we follow a similar sequence of steps to, and replicate notation from, [23], the underlying concepts in the analysis of the two procedures are rather distinct.

The remainder of this paper is organized as follows. In section 2, we specify the problem that we are solving, the algorithm used to solve this problem, our moving average of the norm squared of the gradient estimator, our estimate of its uncertainty set, and our stopping condition. In section 3, we rigorously establish the foundations of our estimators. In section 4, we numerically demonstrate the effectiveness of our estimators and compare our algorithm to a state-of-the-art solver. In section 5, we conclude.

2. Problem formulation and algorithm.

We are interested in solving the minimization problem

$$\min_{x\in\mathbb{R}^n} \|Ax-b\|_B^2, \tag{2.1}$$

where $A\in\mathbb{R}^{m\times n}$ is a coefficient matrix; $B\in\mathbb{R}^{m\times m}$ is any symmetric positive definite matrix; $b\in\mathbb{R}^m$ is a constant vector; and both m and n are large. Note that we allow m and n to be arbitrary, so our methodology applies to overdetermined, underdetermined, and rank-deficient systems. Owing to the size of A, we can only access A through matrix-vector multiplications; similarly, *though we will not need it in our algorithm,* we can access $A^\top$ through matrix-vector multiplications, though this would be substantially more expensive owing to the needed memory access pattern [15]. For all other operations, we make use of efficiently computed (see footnote 3) sketches of A, which we individually denote by (possibly with a subscript)

$$\tilde{A} = AS \in \mathbb{R}^{m\times p}, \tag{2.2}$$

where p is generally significantly smaller than n (see Remark 3.6), and $S\in\mathbb{R}^{n\times p}$ is a random matrix that satisfies the Johnson–Lindenstrauss property [12] defined in the following manner.

Definition 2.1. *A matrix $S\in\mathbb{R}^{n\times p}$ satisfies the Johnson–Lindenstrauss property if there exist constants $C,\omega>0$ such that for all $\delta\geq 0$ and for any $x\in\mathbb{R}^n$,*

$$\mathbb{P}\left(|\|Sx\|_2^2 - \|x\|_2^2| > \delta\|x\|_2^2\right) < 2e^{-\min\left\{\frac{Cp\delta^2}{2},\frac{\delta}{2\omega}\right\}}. \tag{2.3}$$

Table 1
Values of C and ω in Definition 2.1 for common sampling methods.

	C	ω
Gaussian Matrix [5]	1.1	0.47
Achlioptas [1]	1.16	0.46
FJLT [2]	0.83	0.70

Remark 2.2. In Definition 2.1, the constants C and ω are determined by the method used to generate S. There are many choices of these methods, such as sparse Rademacher matrices [1], the Fast Johnson–Lindenstrauss Transform (FJLT) [2], and Gaussian matrices [5, 11, 16]. Estimates for these constants based on numerical experiments are supplied in Table 1.

Remark 2.3. By Definition 2.1, $\frac{\|Sx\|_2^2}{\|x\|_2^2}$ is a subexponential (defined in Definition 3.8) random variable with parameters $(1/(Cp), \omega)$ [31].

To solve this problem we will employ an important subclass of generalized column-space descent methods (see [21]), which begins with an iterate $x_0 \in \mathbb{R}^n$ and generates a sequence of iterates, $\{x_k : k \in \mathbb{N}\}$, according to the recursive equation

$$x_k = x_{k-1} - S_k u_k, \tag{2.4}$$

$$\text{where } u_k = \text{argmin}_{u \in \mathbb{R}^p} \|\tilde{A}_k u - (Ax_{k-1} - b)\|_B^2, \tag{2.5}$$

and $\tilde{A}_k = S_k A$, which can be computed efficiently (see footnote 3). This update is explicitly given by

$$x_k = x_{k-1} - S_k(\tilde{A}_k^\top B \tilde{A}_k)^\dagger \tilde{A}_k^\top B(Ax_{k-1} - b), \tag{2.6}$$

where † is the Moore–Penrose pseudoinverse, and $\{S_k : k \in \mathbb{N}\}$ are independent, identically distributed matrices satisfying Definition 2.1. This form is mathematically equivalent to

$$x_k = x_{k-1} - S_k(S_k^\top A^\top BAS_k)^\dagger S_k^\top A^\top B(Ax_{k-1} - b), \tag{2.7}$$

which is a form that will be useful for proving theory relating to the convergence of (2.6), but which *we do not explicitly use for the algorithm as $A^\top$ is unfavorable to access for large matrices.*

Under this formulation, Algorithm 2.1 presents our methodology for practically tracking and stopping the progress of least squares solvers of the form (2.7) for matrices $\{S_k\}$ that satisfy Definition 2.1. Algorithm 2.1 has several key components that we explain presently.[5]

1. At each iteration, we compute estimators of two key quantities to determine the progress and uncertainty of the algorithm. One quantity we wish to estimate is the moving average of the norms squared of the gradients, ρ_k^λ, which we define as

$$\rho_k^\lambda = \sum_{i=k-\lambda+1}^{k} \frac{\|g_i\|_2^2}{\lambda}, \tag{2.8}$$

[5] In Algorithm 2.1, we use $\tilde{A}$ to denote $B^{1/2}AS$, possibly with a subscript. This is done to write Algorithm 2.1 in terms of 2-norms.

Algorithm 2.1 Tracking and Stopping for Least Squares.

```
Require: $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $B^{1/2} \in \mathbb{R}^{m\times m}$, $x_0 \in \mathbb{R}^n$,$\{S_k\}$ satisfying Definition 2.1.
Require: Moving average window widths $\lambda_1 \le \lambda_2 \in \mathbb{N}$.
Require: $\alpha > 0, \xi_I \in (0,1), \xi_{II} \in (0,1), \delta_I \in (0,1)$, $\delta_{II} > 1$, $\eta \ge 1$, $\upsilon > 0$.
$k \leftarrow 0$, $k' \leftarrow \infty$, $\tilde\rho_0^* \leftarrow 0$, $\tilde\iota_0^* \leftarrow 0$, $\lambda \leftarrow 1$, $FLAG \leftarrow$ false.
while $k == 0$ or $\tilde\rho_{k-1}^\lambda \ge \upsilon$ or
    $$\sqrt{\tilde\iota_{k-1}^\lambda} \ge \min\left\{ \frac{\lambda(1-\delta_I)^2\upsilon^2 Cp}{(1+\log(\lambda))2\log(1/\xi_I)\sqrt{\tilde\iota_{k-1}^\lambda}}, \frac{\lambda\upsilon(1-\delta_I)}{2\log(1/\xi_I)\omega}, \frac{\lambda(\delta_{II}-1)^2\upsilon^2 Cp}{(1+\log(\lambda))2\log(1/\xi_{II})\sqrt{\tilde\iota_{k-1}^\lambda}}, \frac{\lambda\upsilon(\delta_{II}-1)}{2\log(1/\xi_{II})\omega} \right\}$$
do
    # Iteration $k$ #
    $r_k \leftarrow B^{1/2}(Ax_k - b)$
    $\tilde A_{k+1} \leftarrow B^{1/2} A S_{k+1}$
    $\tilde g_k \leftarrow \tilde A_{k+1}^\top r_k$
    if $k == 0$ then
        $\lambda \leftarrow 1$
        $\tilde\rho_0, \tilde\iota_0 \leftarrow \|\tilde g_0\|_2^2, \|\tilde g_0\|_2^4$
    else
        if (not $FLAG$) and $\|\tilde g_k\|_2^2 > \|\tilde g_{k-1}\|_2^2$ then
            $FLAG \leftarrow$ true
        end if
        if (not $FLAG$) and $k < \lambda_1$ then
            $\lambda \leftarrow k+1$
            $\tilde\rho_k, \tilde\iota_k \leftarrow (k\tilde\rho_{k-1} + \|\tilde g_k\|_2^2)/\lambda, (k\tilde\iota_{k-1} + \|\tilde g_k\|_2^4)/\lambda$
        else if (not $FLAG$) and $k \ge \lambda_1$ then
            $\lambda \leftarrow \lambda_1$
            $\tilde\rho_k, \tilde\iota_k \leftarrow (\lambda_1\tilde\rho_{k-1} + \|\tilde g_k\|_2^2 - \|\tilde g_{k-\lambda_1}\|_2^2)/\lambda, (\lambda_1\tilde\iota_{k-1} + \|\tilde g_k\|_2^4 - \|\tilde g_{k-\lambda_1}\|_2^4)/\lambda$
        else if $FLAG$ and $\lambda < \lambda_2$ then
            $\lambda \leftarrow \lambda + 1$
            $\tilde\rho_k, \tilde\iota_k \leftarrow ((\lambda-1)\tilde\rho_{k-1} + \|\tilde g_k\|_2^2)/\lambda, ((\lambda-1)\tilde\iota_{k-1} + \|\tilde g_k\|_2^4)/\lambda$
        else
            $\lambda \leftarrow \lambda_2$
            $\tilde\rho_k, \tilde\iota_k \leftarrow (\lambda_2\tilde\rho_{k-1} + \|\tilde g_k\|_2^2 - \|\tilde g_{k-\lambda_2}\|_2^2)/\lambda, (\lambda_2\tilde\iota_{k-1} + \|\tilde g_k\|_2^4 - \|\tilde g_{k-\lambda_2}\|_2^4)/\lambda$
        end if
    end if
    Update the estimated $(1-\alpha)$-interval by computing:
```

Algorithm 2.1 (cont.).

```
        $\tilde{\rho}_k^\lambda \pm \max\left(\sqrt{\frac{2\log(2/\alpha)\tilde{\iota}_k^\lambda(1+\log(\lambda))}{Cp\lambda\eta}}, \frac{2\log(2/\alpha)\sqrt{\tilde{\iota}_k^\lambda}\omega}{\lambda\eta}\right)$
        $u_{k+1} \leftarrow \text{argmin}_u \|\tilde{A}_{k+1}u - r_k\|_2^2$        # See [17] and [10]
        $x_{k+1} \leftarrow x_k - S_k u_{k+1}$
        $k \leftarrow k+1$
    end while
    return $x_k$ and estimated $(1-\alpha)$-interval
```

where $g_k = A^\top B(Ax_k - b)$ is the gradient at iterate x_k, and where λ is the width of the moving window. When $\lambda = 1$, we recover just the gradient at iteration x_{k-1}, and when $\lambda > 1$, we have a moving average of the gradients. As it is infeasible to calculate ρ_k^λ, we estimate ρ_k^λ with the norms squared of the sketched gradients *that have already been computed in the updates of our algorithm* (see (2.6)),

$$\tilde{\rho}_k^\lambda = \sum_{i=k-\lambda+1}^{k} \frac{\|\tilde{g}_i\|_2^2}{\lambda}, \tag{2.9}$$

where $\tilde{g}_k = \tilde{A}_{k+1}^\top B(Ax_k - b)$. When $\lambda = 1$, we recover the sketched gradient norm at iterate x_k, and when $\lambda > 1$, we have a moving average of the sketched gradient norms, which turns out to be more reliable.

2. We derive a distribution for $\tilde{\rho}_k^\lambda$ in subsection 3.2. This distribution relies on an unknown quantity that we estimate using

$$\tilde{\iota}_k^\lambda = \sum_{i=k-\lambda+1}^{k} \frac{\|\tilde{g}_i\|_2^4}{\lambda}. \tag{2.10}$$

3. The matrix $B^{1/2}$ is the square root of the positive definite matrix, $B \in \mathbb{R}^{m\times m}$, used in the general norm. In practice, $B^{1/2}$ can be computed using the Cholesky decomposition, if B is not too dense or large. Fortunately, in many problems that we consider, such as 4D-Var, B has an underlying structure that can be exploited to efficiently compute $B^{1/2}$.

4. The constants C, ω, and p play an important role in the algorithm owing to their relationship with Definition 2.1. The parameters C and ω are constants relating to the size of the tail bound described in Definition 2.1, which depend on the chosen sketching method; see Table 1. The constant p is the embedding dimension of the random matrix S_k, and also appears in the tail bound of Definition 2.1. A small lower bound on the size of p is necessary for convergence (see Lemma 3.5 and Remark 3.6).

5. Line 2 contains the conditions for stopping the algorithm. If ρ_k^λ could be practically calculated, then the algorithm could be stopped when ρ_k^λ falls below a user-specified threshold, υ. However, since we must instead use the estimator of ρ_k^λ, $\tilde{\rho}_k^\lambda$, stopping when $\tilde{\rho}_k^\lambda \le \upsilon$ leads to two possible sources of error.

5a. One type of error is associated with stopping the algorithm later than desired. Algorithmically, this scenario arises when $\rho_k^\lambda \le \upsilon$ while $\tilde{\rho}_k^\lambda > \upsilon$. To control this error, we need two user-specified quantities. The first quantity specifies how far ρ_k^λ is below υ. In particular, we let the user choose $\delta_I \in (0,1)$, and we control the probability that $\rho_k^\lambda \le \delta_I \upsilon$ while $\tilde{\rho}_k^\lambda > \upsilon$. The second quantity is a user-specified bound on this probability, ξ_I, that indicates the user's level of risk tolerance for possibly stopping too late.

5b. The second type of error is associated with stopping too early. Algorithmically, this scenario occurs when $\rho_k^\lambda > \upsilon$, while $\tilde{\rho}_k^\lambda \le \upsilon$. Similar to the first scenario, we will let the user choose $\delta_{II} > 1$ to quantify how much larger ρ_k^λ is in comparison to υ, when $\tilde{\rho}_k^\lambda < \upsilon$. Then we control this probability with a user-specified value ξ_{II}, which reflects the user's level of risk tolerance for potentially stopping too early.

6. The user-specified parameter η is an optional parameter to adjust for the conservativeness of the theoretical confidence interval and stopping condition. If the user specifies $\eta = 1$, then there is no adjustment. Reasonable, yet still conservative, choices for η can be found in Table 2, which are based on numerical simulations.

7. Lines 15, 18, 21, and 24 adaptively change the window width of the moving average. This procedure is necessary as there are two distinct phases of convergence in the algorithm. In the first phase, the iterates converge rapidly towards the solution, which necessitates a smaller moving average window width to reduce the impact of earlier iterates. In the second phase, the iterates begin to make less progress and the randomness of the algorithm is more pronounced in their behavior, which necessitates a larger moving average window width to smooth out this randomness. We identify the change point between the two phases to be the iteration where the norm of the sketched gradients are no longer monotonically decreasing, i.e., $\|\tilde{g}_k\|_2^2 > \|\tilde{g}_{k-1}\|_2^2$. At this point we slowly increase the width of the window from the narrow window width, λ_1, by one at each iteration until it reaches that of the wide window width, λ_2. While we choose the monotonic condition because of its simplicity and effectiveness, other conditions that attempt to estimate the change point between phases could also be used.

8. Lines 16, 19, 22, and 25 inexpensively update the estimators $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$, requiring only four floating point operations to calculate. However, this update can suffer from issues of numerical stability, especially for $\tilde{\iota}_k^\lambda$. If this is a concern, then $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$ can be computed in $\mathcal{O}(\lambda_2)$ time simply by taking the mean of the nonzero entries in its storage vector, ρ or ι.

9. Line 28 describes a $1-\alpha$ credible interval designed to contain ρ_k^λ using the estimators $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$ computed at iteration k. As with the stopping condition, this credible interval is derived in subsection 3.3 from the tail bounding distribution described in subsection 3.2. The parameter α is selected by the user.

3. Validity of the credible interval and stopping condition.

With an understanding of the parts of Algorithm 2.1, we must now demonstrate the validity of Algorithm 2.1. In

Table 2

Table of conservative η values for three sampling methods.

Method	Gaussian	FJLT	Achlioptas
η	3	4	3

particular, we must show that line 28 is a valid, credible interval for ρ_k^λ, and we must show that line 2 controls the probabilities of the two aforementioned errors at ξ_I and ξ_{II}. As both the credible interval and stopping condition depend on $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$, we will need to establish the validity of these two estimators (i.e., their consistency) in order to establish the validity of the credible interval and stopping condition. In turn, as the consistency of $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$ depends on the convergence of the iterates, $\{x_k\}$, we show the convergence of the iterates in subsection 3.1 (specifically, see Theorem 3.7). Then we show that $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$ are consistent estimators for their respective quantities ρ_k^λ and ι_k^λ[6] by deriving a tail bound for both quantities (see subsection 3.2 and Theorems 3.9 and 3.10). Now that we have established the validity of $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$, we derive the credible interval (see subsection 3.2 and Corollary 3.11) and stopping condition (see subsection 3.2 and Corollary 3.12). Both the credible interval and stopping condition require a quantity that is impractical to compute, so we establish that using $\tilde{\iota}_k^\lambda$ as a plug-in estimator for the impractical quantity controls the relative error between the theoretical values for the credible interval and stopping condition and the versions that use the plug-in estimator (see subsection 3.3 and Lemma 3.13).

3.1. Convergence of the iterates.

To show that the iterates converge to a solution, it is equivalent to show that the gradient of the least squares problem goes to zero. In turn, if B is the identity matrix, it is equivalent to show that the component of the residual of the linear system in the column space of A goes to zero. For general B, an analogous equivalence is established in the following lemma.

Lemma 3.1. *Let $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{m\times m}$ be positive definite, and $x \in \mathbb{R}^n$. Let $\mathcal{P}$ be the orthogonal projection onto $\mathrm{col}(B^{1/2}A)$. Then the gradient of the least squares problem at x, $A^\top B(Ax-b)=0$ if and only if $\mathcal{P}B^{1/2}(Ax-b)=0$.*

Proof. Let $r = Ax-b$. Suppose $A^\top Br = 0$,

$$0 = A^\top Br = A^\top B^{1/2}(\mathcal{P}B^{1/2}r + (I-\mathcal{P})B^{1/2}r) = A^\top B^{1/2}\mathcal{P}B^{1/2}r, \tag{3.1}$$

where the last equality comes from $I-\mathcal{P}$ being an orthogonal projector onto the null space of $A^\top B^{1/2}$. Since $\mathcal{P}B^{1/2}r$ is in the range of $B^{1/2}A$ we know that $A^\top B^{1/2}\mathcal{P}B^{1/2}r$ will only be zero when $\mathcal{P}B^{1/2}r = 0$.

Now suppose $\mathcal{P}B^{1/2}r = 0$. Then

[6] This quantity has not yet been defined, but will be defined in subsection 3.2.

$$A^\top B r = A^\top B^{1/2}(\mathcal{P}B^{1/2}r + (I-\mathcal{P})B^{1/2}r) = A^\top B^{1/2}(I-\mathcal{P})B^{1/2}r = 0, \tag{3.2}$$

where the last equality follows from $I-\mathcal{P}$ being an orthogonal projector onto the null space of $A^\top B^{1/2}$. ■

As the preceding lemma establishes, showing $\{\mathcal{P}B^{1/2}r_k\} \to 0$ is equivalent to showing that the iterates converge to a solution. Thus, we establish a recursive relationship between $\mathcal{P}B^{1/2}r_k$ and $\mathcal{P}B^{1/2}r_{k-1}$. From (2.7),

$$r_k = (I - AS_k(S_k^\top A^\top BAS_k)^\dagger S_k^\top A^\top B)r_{k-1}. \tag{3.3}$$

Multiplying both sides by $B^{1/2}$,

$$B^{1/2}r_k = (I - B^{1/2}AS_k(S_k^\top A^\top BAS_k)^\dagger S_k^\top A^\top B^{1/2})B^{1/2}r_{k-1}. \tag{3.4}$$

From here, let $\psi_k = B^{1/2}r_k$. Since $\text{col}(B^{1/2}A) \supset \text{col}(B^{1/2}AS_k)$, multiplying both sides by $\mathcal{P}$ produces

$$\mathcal{P}\psi_k = (I - B^{1/2}AS_k(S_k^\top A^\top BAS_k)^\dagger S_k^\top A^\top B^{1/2})\mathcal{P}\psi_{k-1}. \tag{3.5}$$

Finally, since $B^{1/2}AS_k(S_k^\top A^\top BAS_k)^\dagger S_k^\top A^\top B^{1/2}$ is an orthogonal projection matrix, we can define a matrix Q_k to be the matrix with orthonormal columns that span $\text{col}(B^{1/2}AS_k)$. Then we can write (3.5) as

$$\mathcal{P}\psi_k = (I - Q_kQ_k^\top)\mathcal{P}\psi_{k-1}. \tag{3.6}$$

With these relationships and notation established, we now turn to establishing convergence.

Geometric reduction in residual components that lie in the column space of $B^{1/2}A$. Let $\tau_0 = 0$ and τ_1 being the first iteration, where

$$\text{col}(Q_1) + \text{col}(Q_2) + \cdots + \text{col}(Q_{\tau_1}) = \text{col}(B^{1/2}A) \tag{3.7}$$

is satisfied. Noting that if (3.7) is not satisfied, then τ_1 is infinite; otherwise, τ_1 is finite and the following lemma holds.

Lemma 3.2. *Let $\psi_0 \in \mathbb{R}^m$ and let $\{\mathcal{P}\psi_k\}$ be generated according to (3.5) for $\{S_k : k \in \mathbb{N}\}$, which are independent and identically distributed random matrices satisfying Definition 2.1. On the event, $\{\tau_1 < \infty\}$ there exists a $\gamma_1 \in (0,1)$ such that*

$$\|\mathcal{P}\psi_{\tau_1}\|_2 \le \gamma_1 \|\mathcal{P}\psi_0\|_2. \tag{3.8}$$

Proof. To prove this, it is only necessary to show that γ_1 exists. First, let $q_{k,1}, \ldots, q_{k,p}$ denote the columns of Q_k. Then we can write ψ_{τ_1} by (3.6) as

$$\mathcal{P}\psi_{\tau_1} = \left[\prod_{k=1}^{\tau_1}\left(I - q_{k,j}q_{k,j}^\top\right)\right]\mathcal{P}\psi_0. \tag{3.9}$$

Since $\mathcal{P}\psi_0 \in \text{col}(B^{1/2}A)$, [20, Theorem 4.1] implies that there exist $\gamma_1 \in (0,1)$ that is a function of $\{q_{1,1}, q_{1,2}, \dots, q_{\tau_1,p}\}$ such that $\|\mathcal{P}\psi_{\tau_1}\|_2 \le \gamma_1 \|\mathcal{P}\psi_{\tau_0}\|_2$. ■

We can easily repeat this argument for more than just τ_1; in fact when $\{\tau_\ell < \infty\}$, define $\tau_{\ell+1}$ to be the first iteration after τ_ℓ, where

$$\text{col}(Q_{\tau_\ell+1}) + \text{col}(Q_{\tau_\ell+2}) + \cdots + \text{col}(Q_{\tau_{\ell+1}}) = \text{col}(B^{1/2}A); \tag{3.10}$$

otherwise let $\tau_{\ell+1}$ be infinite. The preceding argument for the existence of $\gamma_1 \in (0,1)$ will then result in the following corollary.

Corollary 3.3. *Let $\psi_0 \in \mathbb{R}^m$ and let $\{\mathcal{P}\psi_k\}$ be generated according to (3.5) for $\{S_k : k \in \mathbb{N}\}$, which are independent and identically distributed random matrices satisfying Definition 2.1. On the event $\cap_{\ell=1}^{L} \{\tau_\ell < \infty\}$ there exist $\gamma_\ell \in (0,1)$ for $\ell = 1, \dots, L$, such that*

$$\|\mathcal{P}\psi_{\tau_L}\|_2 \le \left(\prod_{\ell=1}^{L} \gamma_\ell \right) \|\mathcal{P}\psi_0\|_2. \tag{3.11}$$

Control of random rate and random iteration. While appearing to indicate the convergence of the $\mathcal{P}\psi_k$, Corollary 3.3 does not guarantee that the portion of the ψ_k in the range of $B^{1/2}A$ converges to 0. This lack of guarantee for convergence arises from two possible points of failure, one being the case where $\gamma_\ell \to 1$ as $\ell \to \infty$ and the other being the case where τ_ℓ is infinite. The following result addresses the former issue using the independence of $\{S_k\}$.

Lemma 3.4. *Let $\{S_k : k \in \mathbb{N}\}$ be independent and identically distributed random variables. If, for any $\ell \in \mathbb{N}$, τ_ℓ is finite, then $\{\tau_j - \tau_{j-1} : j \le \ell\}$ exist and are independent and identically distributed; and $\{\gamma_j : j \le \ell\}$ are independent and identically distributed.*

Proof. When τ_ℓ is finite, [7, Theorem 4.1.3] states that $\{Q_{\tau_\ell+1}, \dots, Q_{\tau_\ell+k}\}$ given τ_ℓ are independent of $\{Q_1, \dots, Q_{\tau_\ell}\}$ and are identically distributed to $\{Q_1, \dots, Q_k\}$ for all k. Therefore, $\tau_{\ell+1} - \tau_\ell$ and τ_1 are independent and identically distributed. It follows that γ_ℓ are independent and identically distributed. ■

So far, we only know that $\tau_0 = 0$ is finite. Hence, we only know that the random variable $\tau_1 - \tau_0$ exists, but we do not know anything about its finiteness. The next result provides the appropriate remedy.

Lemma 3.5. *Let $\{S_k : k \in \mathbb{N}\}$ be independent and identically distributed random variables satisfying Definition 2.1. If*

$$p > \frac{2\log(2)}{C\delta^2} \tag{3.12}$$

for some $\delta \in (2\omega\log(2), 1)$,[7] *then there exists $\pi \in (0,1]$ such that for all $\ell \in \mathbb{N}$ and $k \ge \text{rank}(A)$,*

$$\mathbb{P}(\tau_\ell - \tau_{\ell-1} = k) \le \binom{k-1}{\text{rank}(A)-1} (1-\pi)^{k-\text{rank}(A)} \pi^{\text{rank}(A)}. \tag{3.13}$$

[7] The implicit restriction on $\omega \le \frac{1}{2\log(2)}$ poses no real concerns in practice; see Table 1.

Proof. We begin by verifying that for $z \in \operatorname{col}(B^{1/2}A)$ and $z \neq 0$, then $S^\top A^\top B^{1/2} z \neq 0$ with some nonzero probability. Definition 2.1 implies that for any $\delta \in (0,1)$,

$$\begin{aligned} &(3.14) && \mathbb{P}(\|S^\top A^\top B^{1/2} z\|_2^2 > 0) \geq \mathbb{P}\left(\left| \|S^\top A^\top B^{1/2} z\|_2^2 - \|A^\top B^{1/2} z\|_2^2 \right| \leq \delta \|A^\top B^{1/2} z\|_2^2 \right) \\ &(3.15) && \geq 1 - 2e^{-\min\left\{\frac{Cp\delta^2}{2}, \frac{\delta}{2\omega}\right\}}. \end{aligned}$$

When $\delta \in (2\omega \log(2), 1)$ is chosen such that (3.12) holds, then $1 - 2e^{-\min\left\{\frac{Cp\delta^2}{2}, \frac{\delta}{2\omega}\right\}} > 0$. Moreover, as this bound is independent of $z \in \operatorname{col}(B^{1/2}A)$, we will refer to the lower bound of $\mathbb{P}(\|S^\top A^\top B^{1/2} z\|_2^2 > 0)$ by $\pi \in (0,1]$ for any $z \neq 0$. Thus, owing to the relationship between Q_k and $\operatorname{col}(B^{1/2}AS_k), \mathbb{P}(\|Q_k^\top z\|_2 > 0) \geq \pi$ for all $z \neq 0$.

Given that $\{\operatorname{col}(Q_k) : k \in \mathbb{N}\}$ are independent and identically distributed, we conclude that the probability that $\operatorname{col}(Q_1) + \cdots + \operatorname{col}(Q_{k+1})$ increases in dimension from $\operatorname{col}(Q_1) + \cdots + \operatorname{col}(Q_k)$, when $\dim(\operatorname{col}(Q_1) + \cdots + \operatorname{col}(Q_{k+1})) < \operatorname{rank}(A)$, is at least π. This implies that the probability that the dimension increases $\operatorname{rank}(A)$ times in the first k iterations with $k > \operatorname{rank}(A)$ is dominated by a negative binomial distribution, i.e., for $k \geq \operatorname{rank}(A)$,

$$(3.16) \qquad \mathbb{P}(\tau_1 = k) \leq \binom{k-1}{\operatorname{rank}(A) - 1} (1-\pi)^{k - \operatorname{rank}(A)} \pi^{\operatorname{rank}(A)}.$$

As a result, τ_1 is finite with probability one. The result follows by Lemma 3.4. ■

Remark 3.6. If $\delta = .7$, for the Gaussian, Achlioptas, and FJLT sampling methods one should choose $p \geq 2$ to satisfy the hypothesis of Lemma 3.5.

Convergence of the moments. With the establishment of the previous lemmas we can now conclude the following theorem.

Theorem 3.7. *Let $x_0 \in \mathbb{R}^n$ and let $\mathcal{P}$ be the orthogonal projection onto* $\operatorname{col}(B^{1/2}A)$. *Suppose that $\{S_k : k \in \mathbb{N}\}$ are independent and identically distributed random variables satisfying Definition* 2.1 *and* (3.12) *for some $\delta \in (0,1)$. Let $\{x_k : k \in \mathbb{N}\}$ be generated according to* (2.6). *Define $\psi_k = B^{1/2}(Ax_k - b)$. Then for any $d \in \mathbb{N}$, $\mathbb{E}\left[\|\mathcal{P}\psi_k\|_2^d\right] \to 0$ and $\mathbb{E}\left[\|\tilde{g}_k\|_2^d\right] = \mathbb{E}\left[\|\tilde{A}_{k+1}^\top B(Ax_k - b)\|_2^d\right] \to 0$ as $k \to \infty$. Furthermore, for any particular ℓ we have*

$$(3.17) \qquad \mathbb{E}\left[\|\mathcal{P}\psi_{\tau_\ell}\|_2^d\right] \leq \mathbb{E}[\gamma_1^d]^\ell \|\mathcal{P}\psi_0\|_2^d.$$

Proof. It is enough to show that $\mathbb{E}\left[\|\mathcal{P}\psi_{\tau_\ell}\|_2^d\right] \to 0$ as $k \to \infty$. By Lemma 3.2, $\|\mathcal{P}\psi_k\|_2$ is a nonincreasing sequence. Thus, we only need to show a subsequence converges to zero. By Corollary 3.3 and Lemmas 3.4 and 3.5,

$$(3.18) \qquad \mathbb{E}\left[\|\mathcal{P}\psi_{\tau_\ell}\|_2^d\right] \leq \mathbb{E}[\gamma_1^d]^\ell \|\mathcal{P}\psi_0\|_2^d$$

for all $\ell \in \mathbb{N}$, where $\mathbb{E}[\gamma_1^d] < 1$. Therefore, as $\ell \to \infty$, the conclusion follows. ■

3.2. Theoretical values for the credible interval and stopping condition.

With convergence in all moments established, we now turn to understanding the distributions of $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$, in order to validate the estimators as well as derive the stopping condition and credible interval. We begin with an examination of the distribution of $\tilde{\rho}_k^\lambda$. To perform this examination, it is first important to present the definition of a subexponential distribution for it will be used throughout this subsection.

Definition 3.8. *For a random variable Y, with $\mathbb{E}[Y]=\mu$, $Y-\mu$ follows a subexponential, SE(σ^2,ω), distribution with parameters σ^2 and ω if for all $\delta \geq 0$*

$$\mathbb{P}\left(|Y-\mu|>\delta\right) \leq 2e^{-\min\{\delta^2/(2\sigma^2),\delta/(2\omega)\}}. \tag{3.19}$$

Equivalently, a random variable $Y-\mu$ is subexponential, SE(σ^2,ω), if

$$\mathbb{E}[e^{t(Y-\mu)}] \leq e^{\frac{t^2\sigma^2}{2}}, \tag{3.20}$$

when $|t|<1/\omega$ [31].

With this definition established, we can note intuitively that if the terms of $\tilde{\rho}_k^\lambda$ were independent, we would trivially have that $\tilde{\rho}_k^\lambda$ satisfies Definition 3.8. Unfortunately, *they are not independent.* Thus, we innovate the following method to derive the distribution of $\tilde{\rho}_k^\lambda$ to handle the dependencies, which results in only an additional logarithmic term relative to what would have been the case, if the terms had been independent.

Theorem 3.9. *Suppose the setting of Theorem 3.7 holds. Define $\mathcal{F}_{k-\lambda+1}$ to be the σ-algebra generated by $S_1,\ldots,S_{k-\lambda+1}$. Then*

$$\tilde{\rho}_k^\lambda - \rho_k^\lambda \Big| \mathcal{F}_{k-\lambda+1} \sim \mathbf{SE}\left(\frac{M_{k-\lambda+1}^4(1+\log(\lambda))}{Cp\lambda}, \frac{\omega M_{k-\lambda+1}^2}{\lambda}\right), \tag{3.21}$$

where $M_{k-\lambda+1} = \|A^\top B^{1/2}\|_2 \|\mathcal{P}B^{1/2}r_{k-\lambda+1}\|_2$ and $r_{k-\lambda+1}=Ax_{k-\lambda}-b$.

Proof. By induction, we prove, for $|t| \leq \lambda/(\omega M_{k-\lambda+1}^2)$,

$$\mathbb{E}\left[\prod_{i=k-\lambda+1}^{k} \exp\left\{\frac{t}{\lambda}\left(\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2\right)\right\}\Bigg| \mathcal{F}_{k-\lambda+1}\right] \leq \exp\left(\frac{t^2 M_{k-\lambda+1}^4}{2Cp\lambda}\sum_{j=1}^{\lambda}\frac{1}{j}\right), \tag{3.22}$$

where $M_{k-\lambda+1} = \|A^\top B^{1/2}\|_2 \|\mathcal{P}B^{1/2}r_{k-\lambda+1}\|_2$ and the bound on t comes from Lemma SM3.1 in the supplementary material. We can then use a logarithm to bound the summation. As a result, the subexponential distribution of $\tilde{\rho}_k^\lambda - \rho_k^\lambda$ follows by Definition 3.8.

The base case of $\lambda=1$ follows trivially from $\|\tilde{g}_{k-\lambda+1}\|_2^2$ being subexponential. Now assume that the result holds for $k-\lambda+1$ to $k-1$. Then

$$\mathbb{E}\left[\prod_{i=k-\lambda+1}^{k} \exp\left\{\frac{t\left(\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2\right)}{\lambda}\right\}\middle|\mathcal{F}_{k-\lambda+1}\right] \tag{3.23}$$

$$= \mathbb{E}\left[\mathbb{E}\left[\prod_{i=k-\lambda+1}^{k} \exp\left\{\frac{t\left(\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2\right)}{\lambda}\right\}\middle|\mathcal{F}_k\right]\middle|\mathcal{F}_{k-\lambda+1}\right] \tag{3.24}$$

$$= \mathbb{E}\left[\mathbb{E}\left[\exp\left\{\frac{t\left(\|\tilde{g}_k\|_2^2 - \|g_k\|_2^2\right)}{\lambda}\right\}\middle|\mathcal{F}_k\right]\prod_{i=k-\lambda+1}^{k-1} \exp\left\{\frac{t\left(\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2\right)}{\lambda}\right\}\middle|\mathcal{F}_{k-\lambda+1}\right] \tag{3.25}$$

$$\leq \mathbb{E}\left[\exp\left\{\frac{t^2\|g_k\|_2^4}{2\lambda^2 Cp}\right\}\prod_{i=k-\lambda+1}^{k-1} \exp\left\{\frac{t\left(\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2\right)}{\lambda}\right\}\middle|\mathcal{F}_{k-\lambda+1}\right], \tag{3.26}$$

where we have made use of $\|\tilde{g}_k\|_2^2$ being subexponential in the ultimate line. Now, applying Hölder's inequality and the induction hypothesis,

$$\mathbb{E}\left[\exp\left\{\frac{t^2\|g_k\|_2^4}{2\lambda^2 Cp}\right\}\prod_{i=k-\lambda+1}^{k-1} \exp\left\{\frac{t\left(\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2\right)}{\lambda}\right\}\middle|\mathcal{F}_{k-\lambda+1}\right] \tag{3.27}$$

$$\leq \mathbb{E}\left[\exp\left\{\frac{t^2\|g_k\|_2^4}{2\lambda Cp}\right\}\middle|\mathcal{F}_{k-\lambda+1}\right]^{\frac{1}{\lambda}}\mathbb{E}\left[\prod_{i=k-\lambda+1}^{k-1} \exp\left\{\frac{t\left(\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2\right)}{\lambda-1}\right\}\middle|\mathcal{F}_{k-\lambda+1}\right]^{\frac{\lambda-1}{\lambda}} \tag{3.28}$$

$$\leq \mathbb{E}\left[\exp\left\{\frac{t^2\|g_k\|_2^4}{2\lambda Cp}\right\}\middle|\mathcal{F}_{k-\lambda+1}\right]^{\frac{1}{\lambda}}\exp\left\{\frac{t^2 M_{k-\lambda+1}^4}{2Cp(\lambda-1)}\sum_{j=1}^{\lambda-1}\frac{1}{j}\right\}^{\frac{\lambda-1}{\lambda}}. \tag{3.29}$$

Now, Lemmas 3.1 and 3.4 and Corollary 3.3 imply, with probability one,

$$\|g_k\|_2^4 \leq \|A^\top B^{1/2}\|_2^4\|\mathcal{P}B^{1/2}r_k\|_2^4 \leq \|A^\top B^{1/2}\|_2^4\|\mathcal{P}B^{1/2}r_{k-\lambda+1}\|_2^4 = M_{k-\lambda+1}^4. \tag{3.30}$$

Since $M_{k-\lambda+1}$ is measurable with respect to $\mathcal{F}_{k-\lambda+1}$, we apply the inequality of (3.30) to (3.29) to conclude the proof by induction. ■

With the establishment of the distribution around the difference between $\tilde{\rho}_k^\lambda$ and ρ_k^λ, we also obtain the consistency of $\tilde{\rho}_k^\lambda$ for ρ_k^λ from Theorem 3.9 by allowing $k \to \infty$, taking the expectation of the subexponential tail bound Definition 3.8, and using the dominated convergence theorem to switch the limit and the integral. With this consistency result, *we conclude that $\tilde{\rho}_k^\lambda$ is a valid estimator for ρ_k^λ.*

Just as $\tilde{\rho}_k^\lambda$ is an estimator for ρ_k^λ, $\tilde{\iota}_k^\lambda$ is an estimator for the quantity

$$\iota_k^\lambda = \sum_{i=k-\lambda+1}^{k}\frac{\|A^\top(Ax_i - b)\|_2^4}{\lambda}, \tag{3.31}$$

which is impractical to compute. We now turn to showing the validity of $\tilde{\iota}_k^\lambda$ as an estimator for ι_k^λ. To show the validity of $\tilde{\iota}_k^\lambda$ we transform $\tilde{\iota}_k^\lambda - \iota_k^\lambda$ into a form where we can make repeated applications of (3.22). After making these applications, we get the consistency result for $\tilde{\iota}_k^\lambda$ presented in the following theorem.

Theorem 3.10. *Under the conditions of Theorem 3.7, we have for $\epsilon > 0$*

$$\begin{aligned}\mathbb{P}\left(\left|\tilde{\iota}_k^\lambda - \iota_k^\lambda\right| > \epsilon|\mathcal{F}_{k-\lambda+1}\right) \\ \le 2(1+\lambda)\exp\Bigg(-\min\Bigg(&\frac{\epsilon^2 C p\lambda}{2(2M_{k-\lambda+1}^2+\sqrt{\lambda\epsilon})^2 M_{k-\lambda+1}^4(1+\log(\lambda))}, \\ &\frac{\lambda\epsilon}{2(2M_{k-\lambda+1}^2+\sqrt{\lambda\epsilon})\omega M_{k-\lambda+1}^2}\Bigg)\Bigg),\end{aligned} \tag{3.32}$$

where $M_{k-\lambda+1} = \|A^\top B^{1/2}\|_2\|\mathcal{P}B^{1/2}r_{k-\lambda+1}\|_2$ and $r_{k-\lambda+1} = Ax_{k-\lambda+1} - b$. Thus, as $k\to\infty$, $\tilde{\iota}_k^\lambda$ is a consistent estimator for ι_k^λ.

Proof. Using the definitions of ι_k^λ and $\tilde{\iota}_k^\lambda$ we have

$$\mathbb{P}\left(|\tilde{\iota}_k^\lambda - \iota_k^\lambda| > \epsilon\middle|\mathcal{F}_{k-\lambda+1}\right) \tag{3.33}$$

$$= \mathbb{P}\left(\left|\sum_{i=k-\lambda+1}^{k}\frac{\|\tilde{g}_i\|_2^4 - \|g_i\|_2^4}{\lambda}\right| > \epsilon\middle|\mathcal{F}_{k-\lambda+1}\right) \tag{3.34}$$

$$\le \mathbb{P}\left(\sum_{i=k-\lambda+1}^{k}\left|\frac{\|\tilde{g}_i\|_2^4 - \|g_i\|_2^4}{\lambda}\right| > \epsilon\middle|\mathcal{F}_{k-\lambda+1}\right) \tag{3.35}$$

$$\le \mathbb{P}\left(\sum_{i=k-\lambda+1}^{k}\left|\frac{\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2}{\lambda}\right| \left|\|\tilde{g}_i\|_2^2 + \|g_i\|_2^2\right| > \epsilon\middle|\mathcal{F}_{k-\lambda+1}\right). \tag{3.36}$$

Then, using any constant $G > 2M_{k-\lambda+1}^2$, we partition (3.36) into disjoint sets. Thus,

$$\mathbb{P}\left(\sum_{i=k-\lambda+1}^{k}\left|\frac{\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2}{\lambda}\right| \left|\|\tilde{g}_i\|_2^2 + \|g_i\|_2^2\right| > \epsilon\middle|\mathcal{F}_{k-\lambda+1}\right) \tag{3.37}$$

$$\begin{aligned}&= \mathbb{P}\Bigg(\sum_{i=k-\lambda+1}^{k}\left|\frac{\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2}{\lambda}\right| \left|\|\tilde{g}_i\|_2^2 + \|g_i\|_2^2\right| > \epsilon, \bigcap_{i=k-\lambda+1}^{k}\left\{\left|\|\tilde{g}_i\|_2^2 + \|g_i\|_2^2\right| \le G\right\}\Bigg|\mathcal{F}_{k-\lambda+1}\Bigg) \\ &+ \mathbb{P}\Bigg(\sum_{i=k-\lambda+1}^{k}\left|\frac{\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2}{\lambda}\right| \left|\|\tilde{g}_i\|_2^2 + \|g_i\|_2^2\right| > \epsilon, \bigcup_{i=k-\lambda+1}^{k}\left\{\left|\|\tilde{g}_i\|_2^2 + \|g_i\|_2^2\right| > G\right\}\Bigg|\mathcal{F}_{k-\lambda+1}\Bigg)\end{aligned} \tag{3.38}$$

$$\le \mathbb{P}\Bigg(\sum_{i=k-\lambda+1}^{k}\left|\frac{\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2}{\lambda}\right| > \frac{\epsilon}{G}\Bigg|\mathcal{F}_{k-\lambda+1}\Bigg) + \mathbb{P}\Bigg(\bigcup_{i=k-\lambda+1}^{k}\left\{\left|\|\tilde{g}_i\|_2^2 + \|g_i\|_2^2\right| > G\right\}\Bigg|\mathcal{F}_{k-\lambda+1}\Bigg). \tag{3.39}$$

From here we will present the bounds for the left and right terms of (3.39) separately. For the left-hand term of (3.39) we use a Chernoff bound and (3.22), resulting in

$$\mathbb{P}\Bigg(\sum_{i=k-\lambda+1}^{k} \left| \frac{\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2}{\lambda} \right| > \frac{\epsilon}{G} \Bigg| \mathcal{F}_{k-\lambda+1} \Bigg) \leq 2 \exp\left(\frac{t^2 M_{k-\lambda+1}^4 (1+\log(\lambda))}{2Cp\lambda} - \frac{\epsilon t}{G} \right). \tag{3.40}$$

We next wish to minimize this bound. First note that if unconstrained, this minimization would be achieved by setting $t = \frac{\epsilon C p \lambda}{G M_{k-\lambda+1}^4 (1+\log(\lambda))}$. However, from Definition 3.8 we know this Chernoff bound only holds when $0 \leq t \leq \frac{\lambda}{\omega M_{k-\lambda+1}^2}$, thus minimizing this bound requires the consideration of two cases. In the first case we consider when $\frac{\epsilon C p \lambda}{G M_{k-\lambda+1}^4 (1+\log(\lambda))} < \frac{\lambda}{\omega M_{k-\lambda+1}^2}$, resulting in the minimum of the Chernoff bound of the left-hand term of (3.39) being

$$2 \exp\left(- \frac{\epsilon^2 C p \lambda}{2 G^2 M_{k-\lambda+1}^4 (1+\log(\lambda))} \right). \tag{3.41}$$

In the second case we consider when $\frac{\epsilon C p \lambda}{G M_{k-\lambda+1}^4 (1+\log(\lambda))} > \frac{\lambda}{\omega M_{k-\lambda+1}^2}$, and in this case we set $t = \frac{\lambda}{\omega M_{k-\lambda+1}^2}$, resulting in the minimum of the Chernoff bound of the left-hand term of (3.39) being

$$2 \exp\left(- \frac{\epsilon \lambda}{2 G M_{k-\lambda+1}^2 \omega} \right). \tag{3.42}$$

Combining these two cases we get that

$$\mathbb{P}\Bigg(\sum_{i=k-\lambda+1}^{k} \left| \frac{\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2}{\lambda} \right| > \frac{\epsilon}{G} \Bigg| \mathcal{F}_{k-\lambda+1} \Bigg) \tag{3.43}$$

$$\leq 2 \exp\left(- \min\left(\frac{\epsilon^2 C p \lambda}{2 G^2 M_{k-\lambda+1}^4 (1+\log(\lambda))}, \frac{\lambda \epsilon}{2 G \omega M_{k-\lambda+1}^2} \right) \right). \tag{3.44}$$

We next address the right-hand term of (3.39), for which we have

$$\mathbb{P}\Bigg(\bigcup_{i=k-\lambda+1}^{k} \left\{ \left| \|\tilde{g}_i\|_2^2 + \|g_i\|_2^2 \right| > G \right\} \Bigg| \mathcal{F}_{k-\lambda+1} \Bigg) \tag{3.45}$$

$$= \mathbb{P}\Bigg(\bigcup_{i=k-\lambda+1}^{k} \left\{ \left| \|\tilde{g}_i\|_2^2 - \|g_i\|_2^2 + 2\|g_i\|_2^2 \right| > G \right\} \Bigg| \mathcal{F}_{k-\lambda+1} \Bigg) \tag{3.46}$$

$$\leq \mathbb{P}\Bigg(\bigcup_{i=k-\lambda+1}^{k} \left\{ \left| \frac{\|\tilde{g}_i\|_2^2 - \|g_i\|_2^2}{M_{k-\lambda+1}^2} \right| + 2M_{k-\lambda+1}^2 > G \right\} \Bigg| \mathcal{F}_{k-\lambda+1} \Bigg) \tag{3.47}$$

$$\leq \sum_{i=k-\lambda+1}^{k} \mathbb{P}\Big(\left| \|\tilde{g}_i\|_2^2 - \|g_i\|_2^2 \right| > G - 2M_{k-\lambda+1}^2 \Big| \mathcal{F}_{k-\lambda+1} \Big) \tag{3.48}$$

$$\leq 2\lambda \exp\left(\frac{t^2 M_{k-\lambda+1}^4}{2Cp} - t\left(G - 2M_{k-\lambda+1}^2\right) \right), \tag{3.49}$$

where (3.47) comes from (3.30), and (3.49) comes from the Chernoff bound and (3.22). We next wish to minimize this bound. First note that if unconstrained, this minimization would be achieved by setting $t = \frac{Cp(G-2M_{k-\lambda+1}^2)}{M_{k-\lambda+1}^4}$. However, from Definition 3.8 we know this bound only holds when $0 \leq t \leq \frac{1}{\omega M_{k-\lambda+1}^2}$, and thus minimizing this Chernoff bound requires the consideration of two cases. In the first case, $\frac{Cp(G-2M_{k-\lambda+1}^2)}{M_{k-\lambda+1}^4} < \frac{1}{\omega M_{k-\lambda+1}^2}$, which results in the minimum of the Chernoff bound of the right-hand term of (3.39) being

$$2\lambda \exp\left(-\frac{Cp(G-2M_{k-\lambda+1}^2)^2}{2M_{k-\lambda+1}^4}\right). \tag{3.50}$$

In the second case, $\frac{Cp(G-2M_{k-\lambda+1}^2)}{M_{k-\lambda+1}^2} \geq \frac{1}{\omega M_{k-\lambda+1}^2}$, and in this case we set $t = \frac{\lambda}{\omega M_{k-\lambda+1}^2}$, resulting in the minimum of the Chernoff bound of the right-hand term of (3.39) being

$$2\lambda \exp\left(-\frac{(G-2M_{k-\lambda+1}^2)}{2\omega M_{k-\lambda+1}^2}\right). \tag{3.51}$$

Combining these two cases gives us that

$$\begin{align}
&\mathbb{P}\left(\bigcup_{i=k-\lambda+1}^{k} \left\{\left|\|\tilde{g}_i\|_2^2 + \|g_i\|_2^2\right| > G\right\} \middle| \mathcal{F}_{k-\lambda+1}\right) \tag{3.52}\\
&\leq 2\lambda \exp\left(-\min\left(\frac{Cp(G-2M_{k-\lambda+1}^2)^2}{2M_{k-\lambda+1}^4}, \frac{(G-2M_{k-\lambda+1}^2)}{2\omega M_{k-\lambda+1}^2}\right)\right). \tag{3.53}
\end{align}$$

By combining the left-hand and right-hand terms of (3.39) we get

$$\begin{align}
&\mathbb{P}\left(|\tilde{\iota}_k^\lambda - \iota_k^\lambda| > \epsilon \middle| \mathcal{F}_{k-\lambda+1}\right) \tag{3.54}\\
&\leq 2\exp\left(-\min\left(\frac{\epsilon^2 Cp\lambda}{2G^2 M_{k-\lambda+1}^4(1+\log(\lambda))}, \frac{\lambda\epsilon}{2G\omega M_{k-\lambda+1}^2}\right)\right) \tag{3.55}\\
&\quad + 2\lambda \exp\left(-\min\left(\frac{\left(G-2M_{k-\lambda+1}^2\right)^2 Cp}{2M_{k-\lambda+1}^4}, \frac{G-2M_{k-\lambda+1}^2}{2M_{k-\lambda+1}^2\omega}\right)\right). \nonumber
\end{align}$$

This bound can be tightened by minimizing the bound with respect to G. To do this minimization we first note that when $G \geq 2M_{k-\lambda+1}^2 + \sqrt{\lambda\epsilon} > 2M_{k-\lambda+1}^2$ it is the case that

$$\begin{split}
&\exp\left(-\min\left(\frac{\epsilon^2 Cp\lambda}{2G^2 M_{k-\lambda+1}^4(1+\log(\lambda))}, \frac{\lambda\epsilon}{2G\omega M_{k-\lambda+1}^2}\right)\right)\\
&\quad \geq \exp\left(-\min\left(\frac{\left(G-2M_{k-\lambda+1}^2\right)^2 Cp}{2M_{k-\lambda+1}^4}, \frac{G-2M_{k-\lambda+1}^2}{2M_{k-\lambda+1}^2\omega}\right)\right).
\end{split} \tag{3.56}$$

We can then upper bound the right side of (3.32) in the following manner:

$$\begin{aligned}
&\inf_{G > 2M_{k-\lambda+1}^2} 2\exp\left(-\min\left(\frac{\epsilon^2 Cp\lambda}{2G^2 M_{k-\lambda+1}^4(1+\log(\lambda))}, \frac{\lambda\epsilon}{2G\omega M_{k-\lambda+1}^2}\right)\right) \\
&\qquad + 2\lambda\exp\left(-\min\left(\frac{\left(G-2M_{k-\lambda+1}^2\right)^2 Cp}{2M_{k-\lambda+1}^4}, \frac{G-2M_{k-\lambda+1}^2}{2M_{k-\lambda+1}^2\omega}\right)\right) \qquad (3.57)\\
&\le \inf_{G>2M_{k-\lambda+1}^2+\sqrt{\lambda\epsilon}} 2(1+\lambda)\exp\left(-\min\left(\frac{\epsilon^2 Cp\lambda}{2G^2M_{k-\lambda+1}^4(1+\log(\lambda))}, \frac{\lambda\epsilon}{2G\omega M_{k-\lambda+1}^2}\right)\right) \qquad (3.58)\\
&= 2(1+\lambda)\exp\left(-\min\left(\frac{\epsilon^2 Cp\lambda}{2(2M_{k-\lambda+1}^2+\sqrt{\lambda\epsilon})^2 M_{k-\lambda+1}^4(1+\log(\lambda))},\right.\right. \\
&\qquad \left.\left.\frac{\lambda\epsilon}{2(2M_{k-\lambda+1}^2+\sqrt{\lambda\epsilon})\omega M_{k-\lambda+1}^2}\right)\right), \qquad (3.59)
\end{aligned}$$

where the last line comes from recognizing that (3.58) is monotonically increasing when $G > 0$. We then conclude consistency by taking the expectation and the limit as $k \to \infty$ of both sides. Then by using the dominated convergence theorem to switch the expectation and the limit we can then use the fact that Theorem 3.7 implies that $M_{k-\lambda+1} \to 0$ as $k \to \infty$ to get that the bound converges to zero. This implies the desired consistency result. ■

With the consistency and distributional results now established, we conclude that our estimators are valid; thus, we are now able to derive the credible interval[8] corresponding to Line 28 and the stopping condition[9] corresponding to Line 2 of Algorithm 2.1.

Corollary 3.11. *Under the conditions of Theorem 3.7, a credible interval of level $1-\alpha$ for $\tilde{\rho}_k^\lambda$, corresponding to line 28 in Algorithm 2.1, is*

$$\tilde{\rho}_k^\lambda \pm \max\left(\sqrt{2\log(2/\alpha)\frac{M_{k-\lambda+1}^4(1+\log(\lambda))}{Cp\lambda}}, 2\log(2/\alpha)\frac{M_{k-\lambda+1}^2\omega}{\lambda}\right). \qquad (3.60)$$

Corollary 3.12. *Let $\xi_I, \xi_{II}, \delta_I \in (0,1)$, $\delta_{II} > 1$, and $\upsilon > 0$. Under the conditions of Theorem 3.7, the following statements are true:*

$$\begin{aligned}
&M_{k-\lambda+1}^2 \le \min\left\{\frac{\lambda(1-\delta_I)^2\upsilon^2 Cp}{(1+\log(\lambda))2\log(1/\xi_I)M_{k-\lambda+1}^2}, \frac{\lambda\upsilon(1-\delta_I)}{2\log(1/\xi_I)\omega}\right\} \\
&\Rightarrow \mathbb{P}\left[\tilde{\rho}_{k+1}^\lambda > \upsilon, \rho_k^\lambda \le \delta_I\upsilon \,\middle|\, \mathcal{F}_{k-\lambda+1}\right] < \xi_I
\end{aligned} \qquad (3.61)$$

and

$$\begin{aligned}
&M_{k-\lambda+1}^2 \le \min\left\{\frac{\lambda(\delta_{II}-1)^2\upsilon^2 Cp}{(1+\log(\lambda))2\log(1/\xi_{II})M_{k-\lambda+1}^2}, \frac{\lambda\upsilon(\delta_{II}-1)}{2\log(1/\xi_{II})\omega}\right\} \\
&\Rightarrow \mathbb{P}\left[\tilde{\rho}_{k+1}^\lambda \le \upsilon, \rho_k > \delta_{II}\upsilon \,\middle|\, \mathcal{F}_{k-\lambda+1}\right] < \xi_{II}.
\end{aligned} \qquad (3.62)$$

[8] The proof to this corollary can be found in section SM4 of the supplementary material.
[9] The proof to this corollary can be found in section SM5 of the supplementary material.

3.3. Estimating the credible interval and stopping condition.

Corollaries 3.11 and 3.12 provide a well-controlled uncertainty set and stopping condition, yet require knowing $M_{k-\lambda+1}$, which is usually not available. As stated before, Corollaries 3.11 and 3.12 can be operationalized by replacing $M^4_{k-\lambda+1}$ with $\tilde{\iota}^\lambda_k$. Of course, $M^4_{k-\lambda+1}$ and $\tilde{\iota}^\lambda_k$ must coincide in some sense in order for this estimation to be valid. Indeed, by Theorems 3.7 and 3.10, both $M^4_{k-\lambda+1}$ and $\tilde{\iota}^\lambda_k$ converge to zero as $k \to \infty$, which would allow us to estimate $M^4_{k-\lambda+1}$ with $\tilde{\iota}^\lambda_k$ to generate consistent estimators. However, we could also estimate $M^4_{k-\lambda+1}$ by 0 to generate consistent estimators, but these would be uninformative during finite time. Therefore, we must establish that estimating $M^4_{k-\lambda+1}$ by $\tilde{\iota}^\lambda_k$ is also appropriate within some finite time. In the next result, we establish that the relative error between $M^4_{k-\lambda+1}$ and $\tilde{\iota}^\lambda_k$ is controlled by a constant (in probability).[10]

Lemma 3.13. *Under the conditions of Theorem 3.7, for $\epsilon > 0$, $M_{k-\lambda+1}$ as described in Theorem 3.9, and $\tilde{\iota}^\lambda_k$ as defined in (2.10),*

$$\begin{aligned}(3.63)\quad &\mathbb{P}\left(\left|\frac{M^4_{k-\lambda+1}-\tilde{\iota}^\lambda_k}{M^4_{k-\lambda+1}}\right| > 1+\epsilon, M^4_{k-\lambda+1}\neq 0 \middle| \mathcal{F}_{k-\lambda+1}\right)\\ &\leq 2(1+\lambda)\exp\left(-\min\left(\frac{\epsilon^2 Cp\lambda}{2(2+\sqrt{\lambda\epsilon})^2(1+\log(\lambda))}, \frac{\lambda\epsilon}{2(2+\sqrt{\lambda\epsilon})\omega}\right)\right).\end{aligned}$$

Owing to Lemma 3.13, the relative error between $\tilde{\iota}^\lambda_k$ and $M_{k-\lambda+1}$ is reasonably well controlled for practical purposes. As a result, we can use $\tilde{\iota}^\lambda_k$ as a plug-in estimator for $M_{k-\lambda+1}$ in the credible interval, (3.60), to produce the estimated credible interval suggested in line 28 of Algorithm 2.1, and we do the same for the stopping condition controls in (3.61) and (3.62) to produce the estimated stopping condition in line 2 of Algorithm 2.1.

4. Experimental results.

Here, we have two goals. First, we demonstrate the correctness of our theory using numerical simulations. Specifically, we verify the consistency of $\tilde{\rho}^\lambda_k$ and $\tilde{\iota}^\lambda_k$ (see subsection 4.1); we verify the coverage probabilities of the credible intervals (see subsection 4.2); and we verify the effectiveness and error control for the stopping condition (see subsection 4.3). Second, we compare our method to state-of-the-art methods on an inner loop of incremental 4D-Var at very large scales (see subsection 4.4). A summary of these experiments can be found in Table 3.[11]

4.1. Consistency of estimators.

To verify the consistency of our estimators, we solve 44 least squares problems (512 unknowns, 1024 equations) with coefficient matrices generated from *Matrix Depot* [32]. Each of these least squares problems is solved three times, once for each of the FJLT, Gaussian, and Achlioptas sketching methods, using an embedding dimension of $p = 20$, a narrow moving average window width of $\lambda_1 = 1$, and a wide moving average window width of $\lambda_2 = 100$ for 10,000 iterations. At each iteration, for each of the three different sketching methods and 44 matrix systems, the values of $\tilde{\rho}^\lambda_k$, $\tilde{\iota}^\lambda_k$, ρ^λ_k, and ι^λ_k are recorded. Using these values, we compute the relative error for both estimators, $\tilde{\rho}^\lambda_k$ and $\tilde{\iota}^\lambda_k$, by taking the absolute value of the difference between the value of the estimator and the value of

[10] The proof of Lemma 3.13 can be found in section SM7 of the supplementary material.

[11] We are using × here to denote the Cartesian product between sets.

Table 3
Summary of experiments.

Section	Question being addressed	Matrices used	Dimensions
subsection 4.1	Are $\tilde{\rho}_k^\lambda$ and ι_k^λ consistent estimators?	44 matrices from *Matrix Depot*	1024 by 512
subsection 4.2	Are the $(1-\alpha)$ uncertainty sets of $\tilde{\rho}_k^\lambda$ actually capturing $(1-\alpha)\%$ of ρ_k^λ?	Wilkinson, Rohess, and Golub matrices from *Matrix Depot*	512 by 256
subsection 4.3	Are we stopping the algorithm in accordance with user defined risks?	44 matrices from *Matrix Depot*	1024 by 512
subsection 4.4	How does this method work at scale?	Subproblem from Incremental 4-D Var for the Shallow Water Equation	$2N_cN_t$ by $2N_c$, using $N_c \times N_t = \{20,\dots,1280\} \times \{20\dots640\}$. Additionally, we consider a 5120000 by 20480 system.

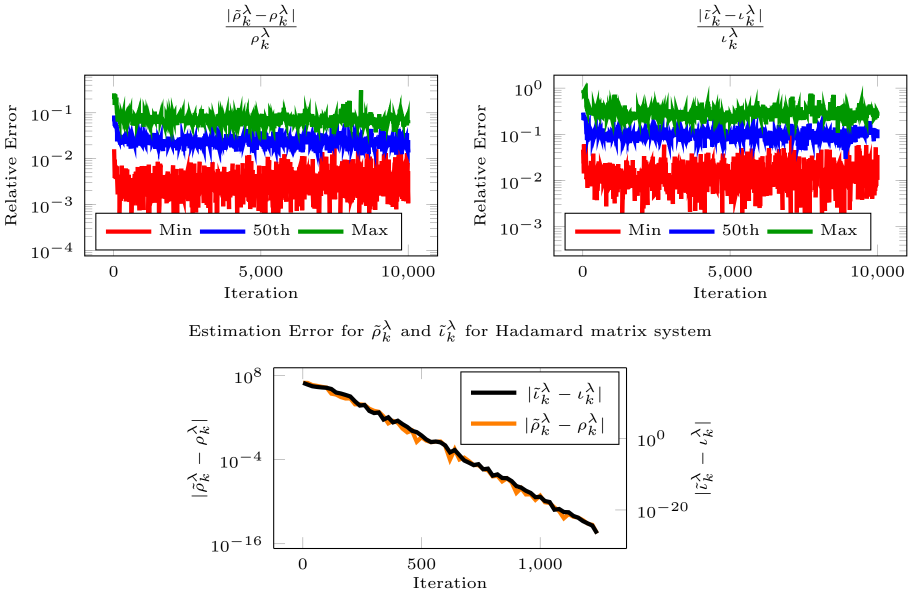

Figure 2. *The top two plots are relative error plots of the min (red), 50th percentile (blue), and max (green) of the relative error between the estimator and actual value. The top left plot features the relative error of* $\tilde{\rho}_k^\lambda$ *across 44 least squares problems solved three times, once using each of the Gaussian, Achlioptas, and FJLT sketching methods. The top right plot features the relative error of* $\tilde{\iota}_k^\lambda$ *for those same problems. The bottom plot features the absolute error for* $\tilde{\rho}_k^\lambda$ *and* $\tilde{\iota}_k^\lambda$ *when applied to the Hadamard matrix system from Matrix Depot.*

the quantity being estimated divided by the value of the quantity being estimated. We then summarize the distribution of these relative errors by computing the min, 50th percentile, and max for both estimator types. In the top left graph of Figure 2, we plot these statistics for the relative error of $\tilde{\rho}_k^\lambda$, $\frac{|\tilde{\rho}_k^\lambda - \rho_k^\lambda|}{\rho_k^\lambda}$; in the top right graph, we do the same for the relative error of $\tilde{\iota}_k^\lambda$, $\frac{|\tilde{\iota}_k^\lambda - \iota_k^\lambda|}{\iota_k^\lambda}$. In the bottom graph we show a specific example of the absolute error $|\tilde{\rho}_k^\lambda - \rho_k^\lambda|$ (orange line), and the absolute error $|\tilde{\iota}_k^\lambda - \iota_k^\lambda|$ (black line) for the solver applied to a Hadamard matrix system from [32]. More detailed results for the max and min of these relative errors across all sampling types, for each of the 44 systems tested, can be found in Table 4.

As Theorems 3.9 and 3.10 show that $\tilde{\rho}_k^\lambda$ and $\tilde{\iota}_k^\lambda$ are consistent estimators, it should be the case that we see constant relative error at all percentiles of the distribution. This is exactly what we obtain when we look at the top two plots in Figure 2 with all the percentiles corresponding to relative errors that fluctuate around a particular constant. This is confirmed in more detail when looking at Table 4 and observing that, aside from the Foxgood and Ursell matrices, all 44 systems see roughly the same minimum and maximum relative errors for each estimator. Looking at the bottom graph, which uses the Hadamard matrix system as an illustrative example, we can see that when ρ_k^λ converges, the absolute error of $\tilde{\rho}_k^\lambda$ converges as well. The same is true when ι_k^λ converges. Overall we can see that our estimators for ρ_k^λ and ι_k^λ are quite good, performing similarly in terms of relative error of estimators across all systems, and clearly consistent when the value being estimated converges.

4.2. Coverage probability. To verify that our credible intervals have the correct coverage probabilities, we perform a two-phase experiment where we solve three linear systems (256 unknowns, 512 equations) with coefficient matrices generated from the Golub, Rohess, and Wilkinson matrices found in *Matrix Depot* [32]. These matrices are chosen owing to the range of their condition numbers, with the Golub, Rohess, and Wilkinson systems having condition numbers of $(81575, 1, 603)$, respectively, which should help reveal the interplay between the coverage of our intervals and the conditioning of the system. In the first phase of the experiment, we solve each system once for 500 iterations using a Gaussian sketching matrix with an embedding dimension of $p = 25$ and a constant moving average window width of $\lambda_1 = \lambda_2 = 15$. At each iteration during this phase, we save the iterate, $\{x_k\}$, $\tilde{\rho}_k^\lambda$, and the 95% credible interval. Once the first phase is complete we move onto the second phase. The goal of the second phase is to approximate the possible variation in ρ_{k+15}^λ given the first phase's iterate, x_k, as starting points as dictated by the conditioning in Theorem 3.9. To accomplish this goal for each saved iterate x_k from the first phase, the second phase starts at that x_k and runs Algorithm 2.1 for 15 iterations. At each of those 15 iterations, the second phase saves the true norm squared of the gradient, $\|g_k\|_2^2$. At the end of those 15 iterations the second phase uses the fifteen $\|g_k\|_2^2$'s to compute ρ_{k+15}^λ. This process is repeated 1000 independent times for each iterate saved in the first phase. This process results in 1000 observations of ρ_k^λ for each iteration greater than 14. Upon the completion of the second phase, we test the coverage of the credible intervals by examining across all iterations how many times the second phase's ρ_k^λ's exceeded the estimated credible interval from the first phase for its corresponding iteration.

In Figure 3, we display for each iteration, k, the credible interval bound shifted by subtracting the first phase's $\tilde{\rho}_k^\lambda$, resulting in an interval centered at zero (see the black lines). Additionally, for each iteration, we display $\rho_k^\lambda - \tilde{\rho}_k^\lambda$ for each of the second phase's 1000 different ρ_k^λ's. If this difference is within the credible interval bound, the observation is colored green; otherwise it is colored red. In the left-hand plots of Figure 3 we display the results

Table 4

Max and min relative errors (RE) for $\tilde{\iota}_k^\lambda$ and $\tilde{\rho}_k^\lambda$ and condition number for each of the 44 systems across each of the three sampling methods.

Matrix	Condition	Max RE $\tilde{\rho}_k^\lambda$	Min RE $\tilde{\rho}_k^\lambda$	Max RE $\tilde{\iota}_k^\lambda$	Min RE $\tilde{\iota}_k^\lambda$
rohess	1	0.63	2.6e-06	0.93	3.6e-05
hadamard	1	0.43	1.2e-05	0.82	1.7e-05
grcar	3.6	0.61	2.6e-05	1.3	3.4e-05
rosser	3.8	0.96	1.3e-06	2.8	7.9e-07
dingdong	4	0.63	1.7e-06	1.6	3.5e-05
parter	4	0.69	8.4e-06	1.6	1.7e-05
randcorr	4.8	0.69	7.8e-06	0.9	2.8e-05
kms	9	0.6	3.3e-06	1.1	4.7e-06
gilbert	10	0.8	2e-06	2.3	3.6e-06
oscillate	12	0.78	5.8e-07	1.6	4.5e-06
smallworld	34	0.59	1.3e-06	1.3	7.8e-06
rando	76	0.71	2.1e-06	1.2	1.1e-05
circul	5.1e+02	0.5	4.4e-06	1.2	5.6e-06
pei	5.1e+02	0.61	2.6e-05	1.6	9.2e-05
hankel	5.2e+02	0.5	1.9e-07	1.2	5.9e-06
wilkinson	1.2e+03	0.47	3.8e-08	1.2	1e-05
randsvd	4.1e+04	0.74	6.7e-07	2	2.2e-05
tridiag	1.1e+05	1.1	7e-07	3.3	3.3e-06
prolate	1.1e+05	0.61	1.7e-06	1.2	5.5e-07
golub	1.1e+05	0.52	2e-07	1.3	1.4e-05
fiedler	1.8e+05	0.76	1.4e-06	2.1	2.8e-07
toeplitz	1.8e+05	0.76	4.9e-07	2.1	1.9e-06
lehmer	2.8e+05	0.7	3.6e-07	1.9	1.4e-05
deriv2	3.2e+05	0.52	7.1e-07	1.3	4.6e-06
minij	4.3e+05	0.65	3.5e-07	1.7	6e-06
phillips	1.8e+09	0.68	9.9e-07	1.8	1.6e-06
chebspec	2.2e+14	0.68	1.6e-06	1.8	1e-05
ursell	1.1e+15	0.47	0.002	1.2	0.011
chow	1.2e+16	0.6	8.6e-07	1.5	3.5e-06
sampling	2e+16	0.8	1.1e-06	2.2	2.5e-05
moler	3.2e+17	0.57	2.4e-07	1.5	4.5e-06
kahan	3.8e+17	0.42	6.6e-07	0.92	4.3e-06
baart	4.1e+17	0.55	5.2e-07	1.7	3.3e-06
cauchy	4.6e+18	0.8	6.2e-06	2.2	7.8e-06
hilb	5.8e+18	0.88	1.6e-06	2.5	3.5e-05
spikes	1.4e+19	0.79	3.6e-06	2.2	8.8e-06
frank	1.6e+19	0.67	2.5e-06	1.8	1.4e-06
lotkin	4.2e+19	0.7	8.1e-06	2.2	6.3e-06
shaw	1.7e+20	1.1	2.1e-06	3.4	5.1e-06
triw	2.6e+20	0.75	3.9e-06	1	4.6e-06
gravity	3e+20	0.68	1.2e-05	1.8	0.00025
magic	5.2e+20	0.75	6.6e-06	2.9	5.1e-07
foxgood	1e+21	0.54	0.074	0.79	0.15
heat	8.7e+124	0.31	4.8e-06	0.8	5.3e-06

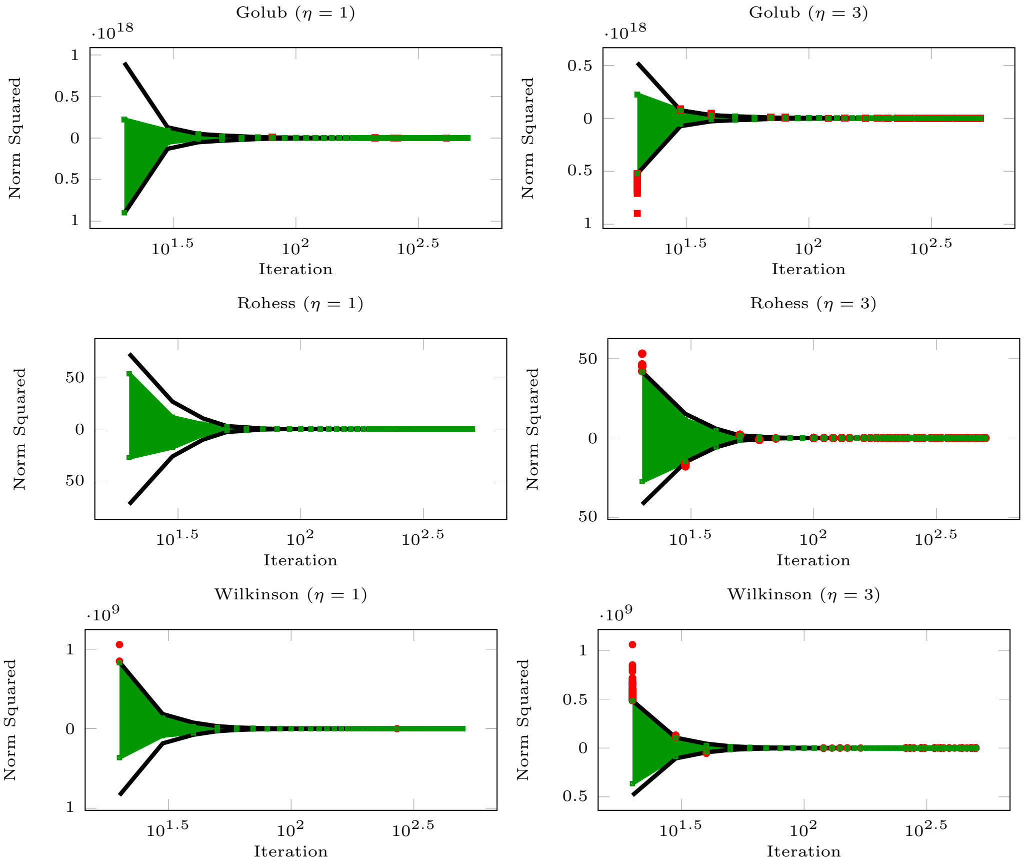

Figure 3. *Coverage results for credible intervals with $\alpha = 0.05$. The plots on the left display the coverage when the credible interval is calculated with $\eta = 1$, while those on the right are computed with η chosen according to Table 2. The green points display all the values of ρ_k^λ that remain within the interval, while the red points are the values of ρ_k^λ that fall outside the interval. The failure rates when $\eta = 1$ for the Golub, Rohess, and Wilkinson matrices are $(0.00548, 0.00000617, 0.00121)$ respectively, while when the η parameter is set according to Table 2 these values change to $(0.125, 0.0162, 0.0428)$.*

for when the credible interval is computed with $\eta = 1$, while the right-hand plots display the results for a credible interval computed with η according to Table 2.

From Figure 3, we can observe that with $\eta = 1$ the credible intervals are conservative, with the coverage failure rates of the Golub, Rohess, and Wilkinson systems being $(0.00548, 0.00000617, 0.00121)$, respectively—all less than the 0.05 failure rate for which the intervals were designed. With the η parameter chosen according to Table 2, we observe failure rates across the different matrix systems that are substantially more aligned with the 0.05 failure rate for which the intervals are designed. The coverage failure rates for the Golub, Rohess, and Wilkinson matrices become $(0.0765, 0.00565, 0.0202)$, respectively. Considering that the condition number for the Golub, Rohess, and Wilkinson matrices are $(81575, 1, 603)$, these results seem to suggest that the choice in η value can be made more or less severe depending on the conditioning of the system, with poorer conditioned systems requiring an η value closer to 1, while better conditioned systems probably require higher η values than what is suggested by Table 2 in order for the intervals to have appropriate coverage rates. Overall, these results demonstrate that, while somewhat conservative, these intervals perform as designed.

4.3. Stopping condition.

To determine the effectiveness of the stopping condition we again consider 44 least squares problems (512 unknowns, 1024 equations) with coefficient matrices generated from *Matrix Depot* [32]. Each of these least squares problems is solved three times for each of the FJLT, Gaussian, and Achlioptas sketching methods with an embedding dimension of $p = 20$, a narrow moving average window width of $\lambda_1 = 1$, and a wide moving average window width of $\lambda_2 = 100$ for 10,000 iterations. After solving these systems, we then consider the frequency at which stopping errors of the form of (3.61) and (3.62) occur when the condition

$$\sqrt{\tilde{\iota}_k^\lambda} \leq \min \left\{ \frac{\lambda(1-\delta_I)^2 \upsilon^2 C p}{(1+\log(\lambda)) 2\log(1/\xi_I)\sqrt{\tilde{\iota}_k^\lambda}}, \frac{\lambda \upsilon (1-\delta_I)}{2\log(1/\xi_I)\omega}, \right. \tag{4.1}$$
$$\left. \frac{\lambda(\delta_{II}-1)^2 \upsilon^2 C p}{(1+\log(\lambda)) 2\log(1/\xi_{II})\sqrt{\tilde{\iota}_k^\lambda}}, \frac{\lambda \upsilon (\delta_{II}-1)}{2\log(1/\xi_{II})\omega} \right\}$$

is satisfied. We do this by considering all iterations where (4.1) is satisfied, then determining the frequency that (3.61)—stopping too late—occurs in these iterations, as well as how often (3.62)—stopping too early—occurs in this set of iterations. The parameters $(\upsilon, \delta_I, \delta_{II}, \xi_I, \xi_{II})$ are set to be $(100, 0.9, 1.1, 0.01, 0.01)$.

Looking at Figure 4, we observe that when (4.1) is satisfied, no error of the form (3.61) or (3.62) occurs, and this continues to be the case even with η set according to Table 2. This low failure rate indicates that overall (4.1) accurately stops the algorithm. Thus, if we stop when both $\tilde{\rho}_k^\lambda \leq \upsilon$ and (4.1) occur, we will make a stopping decision with a magnitude and error rate acceptable to the user.

4.4. 4D-Variational Data Assimilation.

To demonstrate the utility of Algorithm 2.1 at scale, we consider the Incremental 4D-Variational Data Assimilation problem, 4D-Var [4]. This problem is solved by iteratively updating an initial estimate by minimizing the distance between noisy observations at different time points and predictions of these observations made by evolving an estimate of the initial state to the same points in time as the observations. To

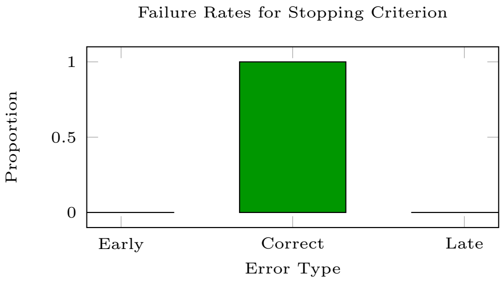

Figure 4. *Graph depicting the stopping decision results by error type. The late category describes an error of the form* (3.61)*, while early describes an error of the form* (3.62)*. These results are displayed with $\eta = 1$; however, they remain unchanged even if η is chosen according to Table* 2*.*

evolve the initial state for our experiment, we use the dynamics defined by the one-dimensional Shallow Water Equations, which are

$$\frac{\partial \phi(x,t)}{\partial t} = -\frac{\partial}{\partial x}\phi(x,t)u(x,t) \quad \text{and} \tag{4.2}$$

$$\frac{\partial u(x,t)}{\partial t} = -\frac{\partial}{\partial x}\left(\phi(x,t) + \frac{u(x,t)^2}{2}\right), \tag{4.3}$$

where x is the spatial coordinate, t is the time point; $\phi(x,t)$ is the potential energy, and $u(x,t)$ is the velocity [6].

To solve the 4D-Var problem with these shallow water dynamics, rather than directly considering Algorithm 2.1, we consider a modified version of Algorithm 2.1, Algorithm SM8.1 in the supplementary material, specifically tailored to the 4D-Var problem in a way that minimizes memory usage, and we compare it to the LSQR solver [18] applied to the same system. For this comparison, we first demonstrate on small problems, i.e., those less than 32 GB in size, how Algorithm SM8.1 produces the same quality of solution as LSQR, has the same runtime scaling as LSQR, and uses substantially less memory than LSQR, when we vary either the number of time points or the number of coordinate points and keep the other at a constant size. We then show that the capabilities of Algorithm SM8.1 exceed those of LSQR by solving a 4D-Var problem where the system size at 0.78 TB far exceeds the 32 GB memory constraint.

To perform both experiments, we generate a set of observations of the potential energy and velocity states for the shallow water equations with the desired number of time and coordinate points. This is done using Euler's method with the initial condition of potential energy being set to $\frac{(i-100)^2}{10000}$, where i is the index of the location, and the initial condition on velocity being set to 0.5 for all coordinates. Each time point is set to be 10^{-11} units apart, and each coordinate point is 100 units apart to ensure that the system can be stably simulated when the number of coordinates and time points is large. Since in most practical instances one would only observe either the potential energy or velocity at a particular location, we set all velocity components of the observations to zero. We then add a vector with mean zero, variance one, and Gaussian entries to the potential energy states at each time point, which results in our noisy observations.[12]

With these observations, we then solve a single inner iteration of the Incremental 4D-Var problem with an initial state estimate of $\frac{(j-100)^4}{10000}$, where j is the entry index of the state vector, once with LSQR and once with Algorithm SM8.1. For Algorithm SM8.1, we use the Achlioptas sketching method with an embedding dimension of $p = 20$, a narrow moving average window width of $\lambda_1 = 1$, and a wide moving average window width of $\lambda_2 = 100$. In order to account for the floating point errors associated with solving large matrix systems, the threshold for stopping is set to be $\upsilon = 10^{-9}(N_c(N_t+1))$, where N_c is the number of coordinates and N_t is the number of time points. The other stopping parameters, $(\delta_I, \delta_{II}, \xi_I, \xi_{II})$, are set to be $(.9, 1.1, .95, .95)$. For both solvers we use a single thread of an Intel Xeon E5-2680 v3 @ 2.50 GHz with a memory constraint of 32 GB. We consider systems with the number of time

[12] Precise formulations of the equations used for Euler's method can be found in Algorithm SM8.1 of the supplementary material.

points varying from 20 to 640 by way of doubling, as well as with the number of coordinate points varying from 20 to 1280 by way of doubling. This results in matrix systems that range in size from 250 KB to 31.25 GB. The LSQR algorithm is stopped once a norm of the gradient of $\sqrt{\upsilon}$ is achieved, and Algorithm SM8.1 is stopped according to line 2 of Algorithm 2.1. Once stopped, we compare the runtime scaling, memory usage, and norm squared of the residual of final solution for both methods in the cases where the number of coordinates changes, but the number of time points stays constant, and vice versa.

The results for keeping the number of time points constant at 640 and varying the number of coordinates are displayed on the left of Figure 5, and the results for keeping the number of coordinates constant at 1280 and varying the number of time points are displayed on the right of Figure 5. In all instances, the minimum residual found by both methods is the same. When considering runtime, the runtime for the LSQR method is faster than Algorithm SM8.1, as long as the matrix system size is less than the memory constraint, and if the system size is greater than the memory constraint, the LSQR method fails. Since we care most about

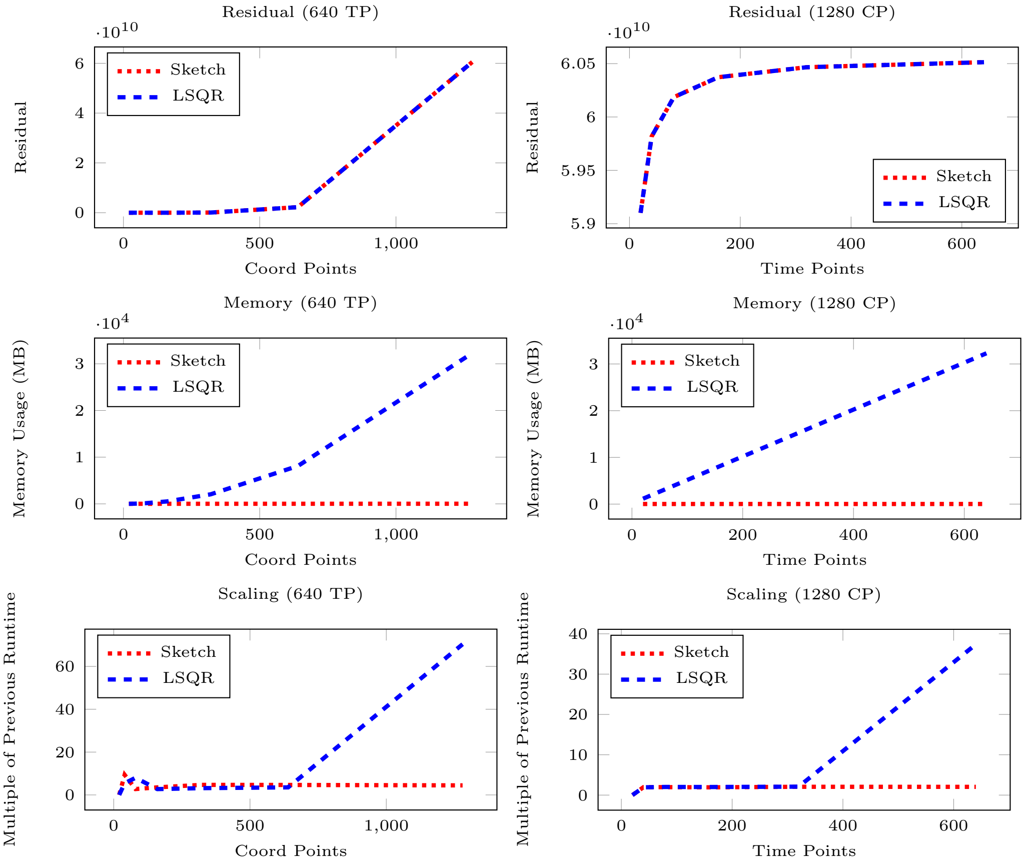

Figure 5. *Displays how residual (top), memory (middle), and slowdown (bottom) compare between LSQR and Algorithm* SM8.1*. The left graphs show scaling when the time points (TP) are set at* 640 *and the number of coordinates are allowed to vary. The right graphs show scaling when the coordinate points (CP) are set at* 1280 *and number of time points are allowed to vary. The blue curve shows the results for the LSQR solver, while the red line shows the results for our solver implemented with Algorithm* SM8.1*.*

how the methods scale with changes in the number of coordinates or number of time points, we present how many times longer the runtime of the solver is at a particular system size, compared to the runtime of the same solver applied to a system with half as many coordinate or time points.

Looking at both sets of plots, we see that for a fixed time point, the LSQR method and Algorithm SM8.1 increase at close to the same rate, with LSQR taking on average 4.57 times longer to solve a problem with twice as many coordinates, and Algorithm SM8.1 taking 5 times longer to solve a problem with twice as many coordinates. This trend continues until we reach the system with 1280 coordinate points, at which point the LSQR runtime is 70 times longer than it was at 640 coordinate points, while Algorithm SM8.1 only takes 4.46 times longer. Evidence for why Algorithm SM8.1 does not experience the same scaling issues as LSQR is found in the memory frame of Figure 5, where we observe the memory usage for Algorithm SM8.1 remains relatively constant at every value of the number of coordinate points, while the memory usage for LSQR grows quadratically over the same span, reaching a maximum of 31.5 GB of memory used. A similar story can be observed if we vary the number of time points, with Algorithm SM8.1 and the LSQR algorithm both doubling in runtime for every doubling in the number of time points, until 640 time points are reached, at which point the scaling for LSQR becomes 37 times that of the previous system size, but remains constant for Algorithm SM8.1. Overall, we can conclude that to generate the same solution quality, Algorithm SM8.1 scales as well as LSQR, but with a longer overall runtime. Further, we can say that Algorithm SM8.1 is significantly more memory efficient than LSQR and is therefore able to avoid the poor scaling effects from memory usage for substantially longer than LSQR.

We finally consider the sketched residual and credible interval for a Shallow Water problem with 250 time points and 10,240 spatial coordinates, which equates to a system with a storage requirement of 0.78 TB. We use Achlioptas sketching with an embedding dimension of $p = 20$, a narrow moving average window width of $\lambda_1 = 1$, and a wide moving average window width of $\lambda_2 = 100$. As with the previous problem we solve this system using a single thread of an Intel Xeon E5-2680 v3 @ 2.50 GHz with a memory constraint of 32 GB of which Algorithm SM8.1 uses 194.68 MB.

We observe in Figure 6 that most of the progress is made within the first 100,000 iterations progressing from a $\tilde{\rho}_k^\lambda$ value of 1.466019×10^{32} to a value of 6520.793. The likely cause for this

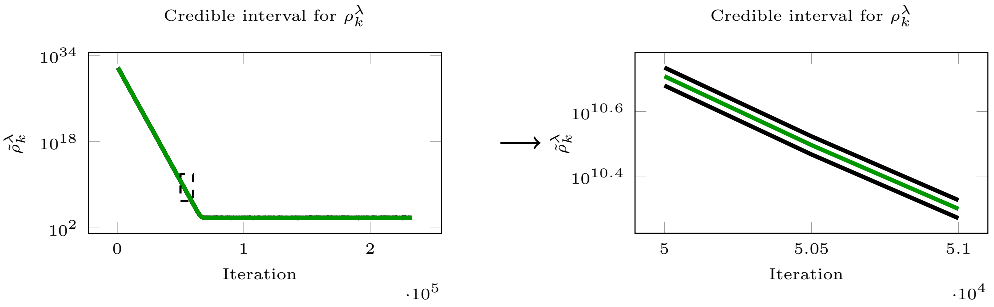

Figure 6. *Displays $\tilde{\rho}_k^\lambda$ and the credible interval for the single inner iteration solve of the large 4D-Var system which has 10,240 spatial coordinates and 250 time points. For better viewing of the interval, the left plot represents all iterations; the right plot is simply iterations 50,000 through 51,000.*

stalled progress is the conditioning of the system, since even at larger sample sizes, $\tilde{\rho}_k^\lambda$ does not improve beyond 6520.793. This leads us to claim we have solved the system sufficiently, and have done so under constraints for which LSQR fails to work.

5. Conclusions.

To efficiently solve the large-scale least squares subproblems that arise in uncertainty quantification, such as 4D-Var, we have proposed an iterative method that leverages random sketching to solve these least squares problems with minimal memory load. The iterative nature of our solution leads to a need to track and stop our method with minimal computational cost, a goal we achieve by utilizing the moving average of the sketched gradients. Through our rigorous proofs, we are then able to verify not only that our algorithm converges, but also that our estimators are consistent and have a quantifiable uncertainty despite their dependent structure. We perform numerous numerical experiments to verify that this theory holds in practice. In addition to the practical verification of our theory, we make clear the advantages of our method over one like LSQR by comparing both solvers on a 0.78 TB system. Through this comparison, we find that while the LSQR method fails because it reaches the 32 GB memory bound, our method can solve the system utilizing only 195 MB of memory. Our future work will involve improving the practicality of our methodology for solving large-scale scientific problems by examining the effects of the choice of embedding dimension on convergence rate and considering parallelization opportunities to reduce runtime.

REFERENCES

[1] D. Achlioptas, *Database-friendly random projections: Johnson-Lindenstrauss with binary coins*, J. Comput. Syst. Sci., 66 (2003), pp. 671–687, https://doi.org/10.1016/S0022-0000(03)00025-4.

[2] N. Ailon and B. Chazelle, *The fast Johnson–Lindenstrauss transform and approximate nearest neighbors*, SIAM J. Comput., 39 (2009), pp. 302–322, https://doi.org/10.1137/060673096.

[3] S. Chen, X. Hong, and C. J. Harris, *Sparse kernel regression modeling using combined locally regularized orthogonal least squares and D-optimality experimental design*, IEEE Trans. Automat. Control, 48 (2003), pp. 1029–1036.

[4] P. Courtier, J.-N. Thépaut, and A. Hollingsworth, *A strategy for operational implementation of 4D-Var, using an incremental approach*, Q. J. Roy. Meteorol. Soc., 120 (1994), pp. 1367–1387.

[5] S. Dasgupta and A. Gupta, *An elementary proof of a theorem of Johnson and Lindenstrauss*, Random Structures Algorithms, 22 (2003), pp. 60–65.

[6] F.-X. L. DIMET and O. TALAGRAND, *Variational algorithms for analysis and assimilation of meteorological observations: Theoretical aspects*, Tellus A, 38A (1986), pp. 97–110, https://doi.org/10.1111/j.1600-0870.1986.tb00459.x.

[7] R. Durrett, *Probability: Theory and Examples*, Cambridge University Press, 2013.

[8] 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.

[9] S. Gurol, A. Weaver, A. Moore, A. Piaccentini, H. Arango, and S. Gratton, *B-preconditioned minimization algorithms for variational data assimilation with the dual formulation*, Q. J. Roy. Meteorol. Soc., 140 (2013), pp. 539–556, https://doi.org/10.1002/qj.2150.

[10] K. Hayami, J.-F. Yin, and T. Ito, *GMRES methods for least squares problems*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2400–2430, https://doi.org/10.1137/070696313.

[11] P. Indyk and R. Motwani, *Approximate nearest neighbors: Towards removing the curse of dimensionality*, in STOC '98: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, 1998, pp. 604–613, https://doi.org/10.1145/276698.276876.

[12] W. Johnson and J. Lindenstrauss, *Extensions of Lipschitz maps into a Hilbert space*, in Conference on Modern Analysis and Probability, Contemp. Math. 26, AMS, 1984, pp. 189–206, https://doi.org/10.1090/conm/026/737400.

[13] A. Krishnan, L. J. Williams, A. R. McIntosh, and H. Abdi, *Partial least squares (PLS) methods for neuroimaging: A tutorial and review*, NeuroImage, 56 (2011), pp. 455–475.

[14] J. Lacotte and M. Pilanci, *Faster Least Squares Optimization*, https://arxiv.org/abs/1911.02675, 2021.

[15] M. S. Lam, E. E. Rothberg, and M. E. Wolf, *The cache performance and optimizations of blocked algorithms*, ACM SIGPLAN Not., 26 (1991), pp. 63–74.

[16] J. Matoušek, *On variants of the Johnson-Lindenstrauss lemma*, Random Structures Algorithms, 33 (2008), pp. 142–156.

[17] A. J. Miller, *Algorithm as* 274*: Least squares routines to supplement those of Gentleman*, J. R. Stat. Soc. Ser. C (Appl. Stat.), 41 (1992), pp. 458–478, https://doi.org/10.2307/2347583.

[18] C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Softw., 8 (1982), pp. 43–71, https://doi.org/10.1145/355984.355989.

[19] V. Patel, M. Jahangoshahi, and D. A. Maldonado, *Convergence of Adaptive, Randomized, Iterative Linear Solvers*, preprint, https://arxiv.org/abs/2104.04816, 2021.

[20] V. Patel, M. Jahangoshahi, and D. A. Maldonado, *An implicit representation and iterative solution of randomly sketched linear systems*, SIAM J. Matrix Anal. Appl., 42 (2021), pp. 800–831, https://doi.org/10.1137/19M1259481.

[21] V. Patel, M. Jahangoshahi, and D. A. Maldonado, *Randomized Block Adaptive Linear System Solvers*, preprint, https://arxiv.org/abs/2204.01653, 2022.

[22] 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. 1842–1879.

[23] N. Pritchard and V. Patel, *Residual Tracking and Stopping for Solving Consistent Linear Inverse Problems with Finite Domains*, https://arxiv.org/abs/2201.05741, 2022.

[24] G. Raskutti and M. Mahoney, *A statistical perspective on randomized sketching for ordinary least-squares*, J. Mach. Learn. Res., 17 (2014), pp. 1–31.

[25] P. Richtárik and M. Takác, *Stochastic reformulations of linear systems: Algorithms and convergence theory*, SIAM J. Matrix Anal. Appl., 41 (2020), pp. 487–524, https://doi.org/10.1137/18M1179249.

[26] I. Selesnick, *Least squares with examples in signal processing*, Connexions, 4 (2013), pp. 1–25.

[27] K. Singh, A. Sandu, M. Jardak, K. W. Bowman, and M. Lee, *A practical method to estimate information content in the context of* 4*D-Var data assimilation*, SIAM/ASA J. Uncertain. Quantif., 1 (2013), pp. 106–138, https://doi.org/10.1137/120884523.

[28] H. C. So and L. Lin, *Linear least squares approach for accurate received signal strength based source localization*, IEEE Trans. Signal Process., 59 (2011), pp. 4035–4040.

[29] O. Talagrand and P. Courtier, *Variational assimilation of meteorological observations with the adjoint vorticity equation.* I*: Theory*, Q. J. R. Meteorol. Soc., 113 (1987), pp. 1311–1328.

[30] A. Tarakanov and A. H. Elsheikh, *Regression-based sparse polynomial chaos for uncertainty quantification of subsurface flow models*, J. Comput. Phys., 399 (2019), 108909, https://doi.org/10.1016/j.jcp.2019.108909.

[31] M. J. Wainwright, *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*, Cambridge Ser. Statist. Probab. Math., Cambridge University Press, 2019, https://doi.org/10.1017/9781108627771.

[32] W. Zhang and N. Higham, *Matrix Depot: An extensible test matrix collection for Julia*, PeerJ. Comput. Sci., 2 (2016), e58, https://doi.org/10.7717/peerj-cs.58.