SIAM J. SCI. COMPUT.
Vol. 33, No. 2, pp. 849–868

A FAST RANDOMIZED ALGORITHM FOR ORTHOGONAL PROJECTION[*]

E. S. COAKLEY[†], V. ROKHLIN[†], AND M. TYGERT[‡]

Abstract. We describe an algorithm that, given any full-rank matrix A having fewer rows than columns, can rapidly compute the orthogonal projection of any vector onto the null space of A, as well as the orthogonal projection onto the row space of A, provided that both A and its adjoint A^* can be applied rapidly to arbitrary vectors. As an intermediate step, the algorithm solves the overdetermined linear least-squares regression involving A^* and may therefore be used for this purpose as well. In many circumstances, the technique can accelerate interior-point methods for convex optimization, including linear programming (see, for example, Chapter 11 of [S. J. Wright, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, 1997]). The basis of the algorithm is an obvious but numerically unstable scheme (typically known as the method of normal equations); suitable use of a preconditioner yields numerical stability. We generate the preconditioner rapidly via a randomized procedure that succeeds with extremely high probability. We provide numerical examples demonstrating the superior accuracy of the randomized method over direct use of the normal equations.

Key words. least squares, regression, numerical stability, normal equations

AMS subject classifications. 65F20, 65C60, 68W20

DOI. 10.1137/090779656

1. Introduction.

This article introduces an algorithm that, given any full-rank matrix A having fewer rows than columns, such that both A and its adjoint A^* can be applied rapidly to arbitrary vectors, can rapidly compute

- the orthogonal projection of any vector b onto the null space of A,
- the orthogonal projection of b onto the row space of A, or
- the vector x minimizing the Euclidean norm $\|A^* x - b\|$ of the difference between $A^* x$ and the given vector b (thus solving the overdetermined linear least-squares regression $A^* x \approx b$).

For simplicity, we focus on projecting onto the null space, describing extensions in Remark 4.1.

In the present paper, the entries of all matrices and vectors are real-valued, though the algorithm and analysis extend trivially to matrices and vectors whose entries are complex-valued. We use the term "condition number" to refer to the l^2 condition number, i.e., the greatest singular value divided by the least. We denote the condition number of a matrix A by $\kappa(A)$. We denote the Euclidean (l^2) norm of a vector b by $\|b\|$.

The basis of our algorithm is the well-known formula for the orthogonal projection Zb of a vector b onto the null space of a full-rank matrix A having fewer rows than

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section January 11, 2010; accepted for publication (in revised form) January 18, 2011; published electronically April 7, 2011. We are grateful for the grants supporting this research, ONR grant N00014-07-1-0711, AFOSR grant FA9550-09-1-0241, and DARPA/AFOSR grant FA9550-07-1-0541, and an Alfred P. Sloan Research Fellowship.
http://www.siam.org/journals/sisc/33-2/77965.html

[†]Program in Applied Mathematics, Yale University, New Haven, CT 06511 (edouard.coakley@yale.edu, vladimir.rokhlin@yale.edu).
[‡]Department of Mathematics, New York University, New York, NY 10012 (tygert@aya.yale.edu).

columns:

$$Zb = b - A^* \, (A\, A^*)^{-1} \, A\, b. \tag{1}$$

That Z is the orthogonal projection onto the null space of A is verified by checking that $AZ = 0$ (so that Zb is in the null space of A for any vector b), that $ZZ = Z$ (so that Z is a projection), and that $Z^* = Z$ (so that the projection is "orthogonal"). Since we are assuming that A and A^* may be rapidly applied to arbitrary vectors, (1) immediately yields a fast algorithm for computing Zb. Indeed, we may apply A to each column of A^* rapidly, obtaining the smaller square matrix AA^*; we may then apply A to b, solve $(AA^*)x = Ab$ for x, apply A^* to x, and subtract the result from b, obtaining Zb. However, using (1) directly is numerically unstable, as it is analogous to using the normal equations for computing the solution to a linear least-squares regression (see, for example, [7]).

Numerical instability in the use of the normal equations arises in the inversion of AA^*, i.e., the construction of the matrix $(AA^*)^{-1}$ in (1) for projection onto the null space, or in determining the solution x of $(AA^*)x = Ab$ for the overdetermined linear least-squares problem [3]. Direct use of the normal equations entails explicit computation of the matrix AA^*, thereby squaring the condition number of the procedure. Severe numerical instability ensues when AA^* is singular to the machine precision ε, i.e., the condition number $\kappa(AA^*) \gtrsim 1/\varepsilon$. Therefore, use of the normal equations introduces severe numerical instability in the regime where

$$\kappa(A) = \sqrt{\kappa(AA^*)} \gtrsim \sqrt{1/\varepsilon}; \tag{2}$$

in double-precision arithmetic, $\sqrt{1/\varepsilon} \approx 1\text{E}8$. In comparison, pivoted QR methods using Householder or Givens transformations or Gram–Schmidt (with reorthogonalization) need not entail the construction or inversion of a matrix with condition number $(\kappa(A))^2$ (see, for example, Chapter 2 in [2] or Chapter 5 in [7]). Typically, however, direct QR methods are significantly more computationally expensive than use of the normal equations when A and A^* can be applied rapidly to arbitrary vectors. When A and A^* can be applied rapidly to arbitrary vectors, QR methods can be more difficult to parallelize and may require more floating-point operations and more memory (sometimes more than is commonly available).

A simple modification to the method of normal equations maintains the benefit of rapid applicability of A and A^* while avoiding explicit construction and inversion of AA^*. Namely, we replace (1) with a formula that is equivalent in exact arithmetic (but not in floating-point):

$$Zb = b - A^* \, \left(P^*\right)^{-1} \left(P^{-1}\, A\, A^* \,(P^*)^{-1}\right)^{-1} \, P^{-1}\, A\, b, \tag{3}$$

where P is a matrix such that $P^{-1}\, A$ is well conditioned. If A is very short and fat, then P is small and square, and so we may apply P^{-1} and $(P^*)^{-1}$ reasonably quickly to arbitrary vectors. Thus, given a matrix P such that $P^{-1}\, A$ is well conditioned, (3) yields a fast algorithm for computing Zb with enhanced numerical stability (see Remark 5.3 in section 5 for a discussion of numerical stability in this context). We can rapidly produce such a matrix P via the randomized method of [11], which is based on the techniques of [5] and its predecessors. Although the method of the present paper is randomized, the probability of numerical instability is negligible (see Remark 4.2 in section 4). A preconditioned iterative approach similar to that in [11] is also feasible. Related work includes [1] and [8], which develop a randomized approach to the overdetermined least-squares problem for arbitrary dense matrices and a method for computing the null spaces of sparse matrices.

The remaining sections cover the following: section 2 summarizes relevant facts from prior publications. Section 3 details the mathematical basis for the algorithm of the present paper. Section 4 describes the algorithm in detail and estimates its computational costs. Section 5 reports the results of several numerical experiments. (Rather than providing a long rigorous formal proof of the numerical stability of our method, we give a sketch and then illustrate the stability via numerical examples.) Section 6 sums up the results and contains concluding remarks.

2. Preliminaries.

In this section, we summarize several facts about the spectra of random matrices and about techniques for preconditioning.

2.1. Singular values of random matrices.

The following lemma provides a highly probable upper bound on the greatest singular value of a matrix whose entries are independent and identically distributed (i.i.d.) Gaussian random variables of zero mean and unit variance; formula 8.8 in [6] provides an equivalent formulation of the lemma.

LEMMA 2.1. *Suppose that l and m are positive integers with $l \geq m$. Suppose further that H is an $m \times l$ matrix whose entries are i.i.d. Gaussian random variables of zero mean and unit variance, and α is a positive real number, such that $\alpha > 1$ and*

$$\pi_+ = 1 - \frac{1}{4\left(\alpha^2 - 1\right)\sqrt{\pi l \alpha^2}} \left(\frac{2\alpha^2}{e^{\alpha^2 - 1}}\right)^l \tag{4}$$

is nonnegative.

Then, the greatest singular value of H is no greater than $\sqrt{2l}\,\alpha$ with probability no less than π_+ defined in (4).

The following lemma provides a highly probable lower bound on the least singular value of a matrix whose entries are i.i.d. Gaussian random variables of zero mean and unit variance; formula 2.5 in [4] and the proof of Lemma 4.1 in [4] together provide an equivalent formulation of Lemma 2.2.

LEMMA 2.2. *Suppose that l and m are positive integers with $l \geq m$. Suppose further that H is an $m \times l$ matrix whose entries are i.i.d. Gaussian random variables of zero mean and unit variance, and β is a positive real number, such that*

$$\pi_- = 1 - \frac{1}{\sqrt{2\pi\left(l - m + 1\right)}} \left(\frac{e}{\left(l - m + 1\right)\beta}\right)^{l-m+1} \tag{5}$$

is nonnegative.

Then, the least (that is, the mth greatest) singular value of H is no less than $1/(\sqrt{l}\,\beta)$ with probability no less than π_- defined in (5).

The following lemma provides a highly probable upper bound on the condition number of a matrix whose entries are i.i.d. Gaussian random variables of zero mean and unit variance; the lemma follows immediately from Lemmas 2.1 and 2.2. For simpler bounds, see [4].

LEMMA 2.3. *Suppose that l and m are positive integers with $l \geq m$. Suppose further that H is an $m \times l$ matrix whose entries are i.i.d. Gaussian random variables of zero mean and unit variance, and α and β are positive real numbers, such that*

$\alpha > 1$ and

$$\pi_0 = 1 - \frac{1}{4\,(\alpha^2-1)\sqrt{\pi l \alpha^2}} \left(\frac{2\alpha^2}{e^{\alpha^2-1}}\right)^{l} - \frac{1}{\sqrt{2\pi\,(l-m+1)}} \left(\frac{e}{(l-m+1)\,\beta}\right)^{l-m+1} \tag{6}$$

is nonnegative.

Then, the condition number of H is no greater than $\sqrt{2}\,l\alpha\beta$ with probability no less than π_0 defined in (6).

2.2. Preconditioning. The following lemma, proven in a slightly different form as Theorem 1 in [11], states that, given a short, fat $m \times n$ matrix A, the condition number of a certain preconditioned version of A corresponding to an $n \times l$ matrix G is equal to the condition number of $V^*\,G$, where V is an $n \times m$ matrix of orthonormal right singular vectors of A.

LEMMA 2.4. *Suppose that l, m, and n are positive integers such that $m \le l \le n$. Suppose further that A is a full-rank $m \times n$ matrix, and that the singular value decomposition of A is*

$$A_{m\times n} = U_{m\times m}\,\Sigma_{m\times m}\,(V_{n\times m})^*, \tag{7}$$

where the columns of U are orthonormal, the columns of V are orthonormal, and Σ is a diagonal matrix whose entries are all nonnegative. Suppose in addition that G is an $n \times l$ matrix such that the $m \times l$ matrix $A\,G$ has full rank.

Then, there exist an $m \times m$ matrix P and an $l \times m$ matrix Q whose columns are orthonormal, such that

$$A_{m\times n}\,G_{n\times l} = P_{m\times m}\,(Q_{l\times m})^*. \tag{8}$$

Furthermore, the condition numbers of $P^{-1}\,A$ and $V^*\,G$ are equal for any $m \times m$ matrix P and $l \times m$ matrix Q whose columns are orthonormal such that P and Q satisfy (8).

3. Mathematical apparatus. In this section, we describe a randomized method for preconditioning rectangular matrices and prove that it succeeds with very high probability.

The following theorem states that a certain preconditioned version $P^{-1}\,A$ of a short, fat matrix A is well conditioned with very high probability (see Observation 3.2 for explicit numerical bounds).

THEOREM 3.1. *Suppose that l, m, and n are positive integers such that $m \le l \le n$. Suppose further that A is a full-rank $m \times n$ matrix and that the singular value decomposition of A is*

$$A_{m\times n} = U_{m\times m}\,\Sigma_{m\times m}\,(V_{n\times m})^*, \tag{9}$$

where the columns of U are orthonormal, the columns of V are orthonormal, and Σ is a diagonal matrix whose entries are all nonnegative. Suppose in addition that G is an $n \times l$ matrix whose entries are i.i.d. Gaussian random variables of zero mean and unit variance. Suppose finally that α and β are positive real numbers, such that $\alpha > 1$ and π_0 defined in (6) *is nonnegative.*

Then, there exist an $m \times m$ matrix P and an $l \times m$ matrix Q whose columns are orthonormal such that

$$A_{m\times n}\,G_{n\times l} = P_{m\times m}\,(Q_{l\times m})^*. \tag{10}$$

Furthermore, the condition number of $P^{-1}A$ is no greater than $\sqrt{2}\,l\alpha\beta$ with probability no less than π_0 defined in (6) *for any $m \times m$ matrix P and $l \times m$ matrix Q whose columns are orthonormal, such that P and Q satisfy* (10).

Proof. Combining the facts that the columns of V are orthonormal and that the entries of G are i.i.d. Gaussian random variables of zero mean and unit variance yields that the entries of $H_{m\times l} = (V_{n\times m})^* G_{n\times l}$ are also i.i.d. Gaussian random variables of zero mean and unit variance. Combining this latter fact and Lemmas 2.3 and 2.4 completes the proof. ◻

Observation 3.2. To elucidate the bounds given in Theorem 3.1, we look at some special cases. With $m \geq 2$ and $\alpha \geq 2$, we find that π_0 from Theorem 3.1, defined in (6), satisfies

$$\pi_0 \geq 1 - \frac{1}{4\,(\alpha^2-1)\sqrt{\pi(l-m+2)\alpha^2}}\left(\frac{2\alpha^2}{e^{\alpha^2-1}}\right)^{l-m+2} - \frac{1}{\sqrt{2\pi\,(l-m+1)}}\left(\frac{e}{(l-m+1)\,\beta}\right)^{l-m+1}. \tag{11}$$

Combining (11) and Theorem 3.1 yields strong bounds on the condition number of $P^{-1}A$ when $l-m=4$ and $m \geq 2$: With $l-m=4$, $\alpha^2=4$, and $\beta=3$, the condition number is at most $10l$ with probability at least $1-10^{-4}$. With $l-m=4$, $\alpha^2=7$, and $\beta=26$, the condition number is at most $100l$ with probability at least $1-10^{-9}$. With $l-m=4$, $\alpha^2=9$, and $\beta=250$, the condition number is at most $1100l$ with probability at least $1-10^{-14}$. Moreover, if instead of $l-m=4$ we take l to be a few times m, then the condition number of $P^{-1}A$ is no greater than a small constant (that does not depend on l) with very high probability (see [4]).

4. Description and cost analysis of the algorithm.

4.1. Description of the algorithm.

In this section, we describe the algorithm of the present paper in some detail. We tabulate its computational costs in section 4.2.

Suppose that m and n are positive integers with $m \leq n$, and A is a full-rank $m \times n$ matrix. The orthogonal projection of a vector $b_{n\times 1}$ onto the row space of $A_{m\times n}$ is $A^*\,(A\,A^*)^{-1}\,A\,b$. Therefore, the orthogonal projection of $b_{n\times 1}$ onto the null space of $A_{m\times n}$ is $b - A^*\,(A\,A^*)^{-1}\,A\,b$. After constructing a matrix $P_{m\times m}$ such that the condition number of $P^{-1}A$ is not too large, we may compute the orthogonal projection onto the null space of A via the identity

$$b - A^*\,(A\,A^*)^{-1}\,A\,b = b - A^*\,\left(P^*\right)^{-1}\left(P^{-1}\,A\,A^*\,(P^*)^{-1}\right)^{-1}\,P^{-1}\,A\,b. \tag{12}$$

With such a matrix $P_{m\times m}$, (12) provides a numerically stable means for computing the orthogonal projection of b onto the null space of A.

To construct a matrix $P_{m\times m}$ such that the condition number of $P^{-1}A$ is not too large, we first choose a positive integer l such that $m \leq l \leq n$. Typically, $l = m+4$ is a good choice, as explained in Observation 3.2 and illustrated by the numerical results of section 5. Next, we perform the following two steps:

1. Construct the $m \times l$ matrix

$$S_{m\times l} = A_{m\times n}\,G_{n\times l} \tag{13}$$

one column at a time, where G is a matrix whose entries are i.i.d. random variables. Specifically, generate each column of G in succession, applying

A to the column (and saving the result) before generating the next column. (Notice that we do not need to store all nl entries of G simultaneously.)

2. Construct a pivoted QR decomposition of S^*,

$$(S_{m\times l})^* = Q_{l\times m}\, R_{m\times m}\, \Pi_{m\times m}, \tag{14}$$

where the columns of Q are orthonormal, R is upper-triangular, and Π is a permutation matrix. (See, for example, Chapter 5 of [7] for details on computing the pivoted QR decomposition.)

With $P = \Pi^*\, R^*$, Theorem 3.1 and Observation 3.2 show that the condition number of $P^{-1}\, A$ is not too large, with very high probability.

To construct the matrix

$$Y = \left(P^{-1}\, A\, A^*\, (P^*)^{-1}\right)^{-1}, \tag{15}$$

which appears in the right-hand side of (12), we perform the following three steps:

1. Construct the $m \times m$ matrix

$$P^{-}_{m\times m} = \Pi^*_{m\times m}\, R^{-1}_{m\times m} \tag{16}$$

(recalling that Π is a permutation matrix).

2. For each $k = 1, 2, \ldots, m$, construct the kth column $x^{(k)}$ of an $m \times m$ matrix X via the following five steps:
   a. Extract the kth column $c^{(k)}_{m\times 1}$ of P^-.
   b. Construct $d^{(k)}_{n\times 1} = (A_{m\times n})^*\, c^{(k)}_{m\times 1}$.
   c. Construct $e^{(k)}_{m\times 1} = A_{m\times n}\, d^{(k)}_{n\times 1}$.
   d. Construct $f^{(k)}_{m\times 1} = \Pi_{m\times m}\, e^{(k)}_{m\times 1}$ (recalling that Π is a permutation matrix).
   e. Solve $R^*_{m\times m}\, x^{(k)}_{m\times 1} = f^{(k)}_{m\times 1}$ for $x^{(k)}_{m\times 1}$ (recalling that R is an upper-triangular matrix).

   Notice that $X = P^{-1}\, A\, A^*\, (P^*)^{-1}$, where $P = (R\,\Pi)^*$ and $(P^*)^{-1} = P^-$. Also, because we generate X one column at a time, we do not need to store an extra $\mathcal{O}(nm)$ floating-point words of memory at any stage of the algorithm.

3. Construct the $m \times m$ matrix

$$Y_{m\times m} = X^{-1}_{m\times m}. \tag{17}$$

It follows from (12) that the orthogonal projection of b onto the null space of A is

$$b - A^*\, (A\, A^*)^{-1}\, A\, b = b - A^*\, (P^*)^{-1}\, Y\, P^{-1}\, A\, b. \tag{18}$$

In order to apply $A^*\, (P^*)^{-1}\, Y\, P^{-1}\, A$ to a vector $b_{n\times 1}$, where $P = \Pi^*\, R^*$, we perform the following seven steps:

1. Construct $c_{m\times 1} = A_{m\times n}\, b_{n\times 1}$.
2. Construct $d_{m\times 1} = \Pi_{m\times m}\, c_{m\times 1}$ (recalling that Π is a permutation matrix).
3. Solve $R^*_{m\times m}\, e_{m\times 1} = d_{m\times 1}$ for $e_{m\times 1}$ (recalling that R is an upper-triangular matrix).
4. Construct $f_{m\times 1} = Y_{m\times m}\, e_{m\times 1}$.
5. Solve $R_{m\times m}\, g_{m\times 1} = f_{m\times 1}$ for $g_{m\times 1}$ (recalling that R is an upper-triangular matrix).

6. Construct $h_{m\times 1} = \Pi^*_{m\times m}\, g_{m\times 1}$ (recalling that Π is a permutation matrix).
7. Construct $\tilde{b}_{n\times 1} = (A_{m\times n})^*\, h_{m\times 1}$.

The output $\tilde{b} = A^*\,(P^*)^{-1}\,Y\,P^{-1}\,A\,b$ of the above procedure is the orthogonal projection of b onto the row space of A. The orthogonal projection of b onto the null space of A is then $b - \tilde{b}$.

Remark 4.1. The vector $h_{m\times 1}$ constructed in step 6 minimizes the Euclidean norm $\|(A_{m\times n})^*\, h_{m\times 1} - b_{n\times 1}\|$ of $(A_{m\times n})^*\, h_{m\times 1} - b_{n\times 1}$; that is, $h_{m\times 1}$ solves the overdetermined linear least-squares regression $(A_{m\times n})^*\, h_{m\times 1} \approx b_{n\times 1}$. The vector $\tilde{b}_{n\times 1}$ constructed in step 7 is the orthogonal projection of $b_{n\times 1}$ onto the row space of $A_{m\times n}$.

Remark 4.2. The algorithm of the present section should be numerically stable provided that the condition number of $P^{-1}\,A$ is not too large. Theorem 3.1 and Observation 3.2 show that the condition number is not too large, with very high probability.

4.2. Computational costs. In this subsection, we estimate the number of floating-point operations required by the algorithm of section 4.1.

We denote by C_A the cost of applying $A_{m\times n}$ to an $n \times 1$ vector; we denote by C_{A^*} the cost of applying $(A_{m\times n})^*$ to an $m \times 1$ vector. We will be keeping in mind our assumption that $m \le l \le n$.

Constructing a matrix $P_{m\times m}$ such that the condition number of $P^{-1}\,A$ is not too large incurs the following costs:

1. Constructing S defined in (13) costs $l \cdot (C_A + \mathcal{O}(n))$.
2. Constructing R and Π satisfying (14) costs $\mathcal{O}(l \cdot m^2)$.

Given R and Π (whose product is P^*), constructing the matrix $Y_{m\times m}$ defined in (15) incurs the following costs:

1. Constructing P^- defined in (16) costs $\mathcal{O}(m^3)$.
2. Constructing X via the five-step procedure (steps a–e) costs $m \cdot (C_{A^*} + C_A + \mathcal{O}(m^2))$.
3. Constructing Y in (17) costs $\mathcal{O}(m^3)$.

Summing up, we see from $m \le l \le n$ that constructing R, Π, and Y defined in (15) costs

$$C_{\text{pre}} = (l+m)\cdot C_A + m \cdot C_{A^*} + \mathcal{O}(l \cdot (m^2 + n)) \tag{19}$$

floating-point operations overall, where C_A is the cost of applying $A_{m\times n}$ to an $n \times 1$ vector, and C_{A^*} is the cost of applying $(A_{m\times n})^*$ to an $m \times 1$ vector. Typically, $l = m + 4$ is a good choice, as explained in Observation 3.2 and illustrated by the numerical results of section 5. Given R and Π (whose product is P^*) and Y, computing the orthogonal projection $b - \tilde{b}$ of a vector b onto the null space of A via the seven-step procedure of section 4.1 costs

$$C_{\text{pro}} = C_A + \mathcal{O}(m^2) + C_{A^*} \tag{20}$$

floating-point operations, where again C_A is the cost of applying $A_{m\times n}$ to an $n \times 1$ vector, and C_{A^*} is the cost of applying $(A_{m\times n})^*$ to an $m \times 1$ vector.

5. Numerical examples.

In this section, we illustrate the performance of the algorithm of the present paper via several numerical examples.

We compare the algorithm with the standard method of normal equations discussed in section 1. It is not the purpose of the present paper to compare the algorithm with all methods in all possible configurations of the numerical tests and computational environment; instead, we focus on methods that directly take advantage of the ability to efficiently apply to arbitrary vectors the matrix A on whose null space we would like to project. The canonical approaches requiring the computation of a (possibly pivoted or partial) QR decomposition of this matrix A, including the method of seminormal equations, can be appropriate in some circumstances (especially for small-scale problems; see, for example, section 2.4 of [2]), but they are not of the form that the present paper concerns.

All computations were performed using IEEE standard double-precision variables, whose mantissas have approximately one bit of precision less than 16 digits (so that the relative precision of the variables is approximately .2E–15). Computations were performed on a single core of a 2.8 GHz Intel Pentium D microprocessor with 1 MB of L2 cache and 3.5 GB of RAM. All code was written in Fortran 77 and compiled using the Lahey/Fujitsu Linux Express v6.2 compiler, with the optimization flag `--o2` enabled. All QR decompositions were computed via plane (Householder) reflections (see, for example, Chapter 5 of [7]).

5.1. Generation of random numbers and random unit vectors. Except where otherwise noted, all random numbers were generated using (Mitchell–Moore–Brent–Knuth) lagged Fibonacci pseudorandom sequences, uniformly distributed on $[-1, 1]$, constructed solely via floating-point additions and subtractions (with no integer arithmetic); see, for example, section 7.1.5 of [10]. In particular, we used these pseudorandom numbers for the entries of G in (13).

All exceptions concern the use of high-quality (see, for example, [9]) pseudorandom numbers drawn from a Gaussian distribution of zero mean and unit variance in place of a uniform distribution on $[-1, 1]$. Although previous sections of this paper use Gaussian random variables to simplify the theoretical analysis, there appears to be no practical advantage to using high-quality truly Gaussian pseudorandom numbers; cf. Remark 5.4. As demonstrated by the results reported in section 5.4, the very quickly generated lagged Fibonacci sequences perform just as well in our algorithm.

Except where otherwise noted, each random unit vector b was computed as $b = c/\|c\|$, where c is a vector whose entries are i.i.d. random variables, and $\|\cdot\|$ denotes the Euclidean norm.

5.2. Construction and properties of test matrices. Definition of *A*. Tables 1–5 display the results of applying the algorithm to project onto the null space of the sparse matrix $A_{m\times n}$ defined as follows. First, we define the circulant matrix $B_{m\times m}$ via the following formula for each entry:

$$B_{j,k} = \begin{cases} 1, & j = k-2 \bmod m, \\ -4, & j = k-1 \bmod m, \\ 6+d, & j = k, \\ -4, & j = k+1 \bmod m, \\ 1, & j = k+2 \bmod m, \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

for $j, k = 1, 2, \dots, m$, where d is a real number shortly to be specified. Taking n to be a positive integer multiple of m,

$$p = \frac{n}{m}, \tag{22}$$

we define the matrix $A_{m\times n}$ via the formula

$$A_{m\times n} = \frac{\sqrt{m/n}}{16+d}\cdot U_{m\times m}\cdot \left(\, B_{m\times m} \mid B_{m\times m} \mid \cdots \mid B_{m\times m} \mid B_{m\times m} \, \right)_{m\times n}\cdot V_{n\times n}, \tag{23}$$

where $U_{m\times m}$ and $V_{n\times n}$ are drawn uniformly at random from the sets of $m \times m$ and $n \times n$ permutation matrices, and $B_{m\times m}$ is the matrix given by (21).

Definition of $\tilde{A}$. Tables 6–7 display the results of applying the algorithm to project onto the null space of the dense matrix $\tilde{A}_{m\times n}$ defined by

$$\tilde{A}_{m\times n} = F_{m\times m} \cdot A_{m\times n}, \tag{24}$$

where $A_{m\times n}$ is the sparse matrix defined in (23), and $F_{m\times m}$ is the orthogonal matrix corresponding to a real discrete Fourier transform (DFT) of size m. It follows from (24) that the condition numbers of $\tilde{A}$ and A are equal, i.e.,

$$\kappa(\tilde{A}) = \kappa(A). \tag{25}$$

We note that, as with A and A^*, we may rapidly apply $\tilde{A}$ and $\tilde{A}^*$, since the matrices $F_{m\times m}$ and $F^*_{m\times m} = F^{-1}_{m\times m}$ may be rapidly applied via a fast Fourier transform (we used Swarztrauber's FFTPACK for the fast Fourier transform). Unlike A and A^*, the matrices $\tilde{A}$ and $\tilde{A}^*$ are dense.

Condition number and norm. As $B_{m\times m}$ is a real self-adjoint circulant matrix, it is diagonalized by the real DFT matrix $F_{m\times m}$, and the spectrum of B is easily seen to be

$$\lambda_k = d + 6 - 8\cos\left(\frac{2\pi k}{m}\right) + 2\cos\left(\frac{4\pi k}{m}\right) \tag{26}$$

for $k = 1, 2, \ldots, m$. The condition number of B is therefore $\kappa = (16+d)/d$, and the condition number of A or $\tilde{A}$ is consequently also $\kappa = (16+d)/d$. In accordance with this relation, we take $d = 16/(\kappa-1)$.

The relation (26) also shows that the spectral (l^2 operator) norm $\|B_{m\times m}\| = 16 + d$, so that from (23) and (24) it is easily seen that

$$\left\|\tilde{A}\right\| = \|A\| = 1, \tag{27}$$

where $\|\cdot\|$ denotes the spectral (l^2 operator) norm; i.e., $\|A\|$ is the greatest singular value of A.

Null space and row space. Since B is nonsingular and therefore $By = 0$ only if $y = 0$, the relation (23) yields that $Ax = 0$ if and only if

$$x_{m\times 1} = V^*_{n\times n} \cdot \begin{pmatrix} \zeta_1 y_{m\times 1} \\ \zeta_2 y_{m\times 1} \\ \vdots \\ \zeta_p y_{m\times 1} \end{pmatrix}_{n\times 1} \tag{28}$$

with $\sum_{k=1}^{p} \zeta_k = 0$ and/or $y = 0$. Similarly, a column vector $w_{m\times 1}$ is in the column space of A^* if and only if

$$w_{m\times 1} = V^*_{n\times n} \cdot \begin{pmatrix} y_{m\times 1} \\ \vdots \\ y_{m\times 1} \end{pmatrix}_{n\times 1} \tag{29}$$

for some $y_{m\times 1}$. These relations facilitate the computation of random unit vectors in the null space of A (kernel of A) or the column space of A^* (cokernel of A). As the null spaces of A and $\tilde{A}$ are identical due to (24), the same consideration applies to $\tilde{A}$.

Remark 5.1. A random unit vector in the null space of A or $\tilde{A}$ is given by

$$x_{m\times 1} = V^*_{n\times n} \cdot \frac{1}{\sqrt{n/m}} \begin{pmatrix} \zeta_1 y_{m\times 1} \\ \vdots \\ \zeta_p y_{m\times 1} \end{pmatrix}_{n\times 1}, \tag{30}$$

where $(\zeta_1, \zeta_2, \ldots, \zeta_p)$ is a random unit vector of length $p = n/m$ such that $\sum_{k=1}^{p} \zeta_k = 0$, and y is a random unit column-vector of length m. A random unit vector in the column space of A^* or $\tilde{A}^*$ is given by

$$w_{m\times 1} = V^*_{n\times n} \cdot \frac{1}{\sqrt{n/m}} \begin{pmatrix} y_{m\times 1} \\ \vdots \\ y_{m\times 1} \end{pmatrix}_{n\times 1}, \tag{31}$$

where y is a random unit column-vector of length m.

5.3. Notation. *Explanation of notation.* The tables in section 5.4 present numerical results for the matrices A and $\tilde{A}$ corresponding to varying values of n, m, and l. The tables provide three measures, designated with the symbols δ, ϵ, and ρ, characterizing the accuracies of the standard method of normal equations and the randomized method for projection onto the null space. To describe these measures, we will need the following notation: We define Z to be the exact orthogonal projection onto the null space and Z_{norm} to be the orthogonal projection onto the null space calculated via the method of normal equations, as detailed in section 1. We define Z_{rand} to be the orthogonal projection onto the null space calculated via the randomized method, as detailed in section 4.

The first measure δ addresses the extent to which each method, executed in finite-precision arithmetic, produces vectors that are indeed in the null space. Specifically, we define

$$\delta_{\text{norm}} = \|A \cdot Z_{\text{norm}}(b)\| \tag{32}$$

and

$$\delta_{\text{rand}} = \|A \cdot Z_{\text{rand}}(b)\|, \tag{33}$$

where $\|\cdot\|$ denotes the Euclidean norm. The values of δ_{norm} and δ_{rand} presented for each matrix correspond to the largest obtained over 100 realizations of a random unit vector b. Small values correspond to high accuracy.

The second measure ϵ addresses the extent to which each method, executed in finite-precision arithmetic, is a projection, i.e., is idempotent ($ZZ = Z$). Specifically, we define

$$\epsilon_{\text{norm}} = \|Z_{\text{norm}}(Z_{\text{norm}}(b)) - Z_{\text{norm}}(b)\| \tag{34}$$

and

$$\epsilon_{\text{rand}} = \|Z_{\text{rand}}(Z_{\text{rand}}(b)) - Z_{\text{rand}}(b)\|. \tag{35}$$

The values of ϵ_{norm} and ϵ_{rand} presented for each matrix correspond to the largest obtained over 100 realizations of a random unit vector b. Small values correspond to high accuracy.

The third measure ρ quantifies the discrepancy between the size of the exact projection and its approximation produced by each method executed in finite-precision arithmetic. This measure also characterizes the accuracy of the solution to the corresponding least-squares problem. Specifically, we define

$$\rho_{\mathrm{norm}} = \frac{\|r_{\mathrm{norm}}\|^2 - \|r\|^2}{\|r\|^2} \tag{36}$$

and

$$\rho_{\mathrm{rand}} = \frac{\|r_{\mathrm{rand}}\|^2 - \|r\|^2}{\|r\|^2}, \tag{37}$$

where r, r_{norm}, and r_{rand} are the residuals to the corresponding least-squares problem, i.e.,

$$r = Z(b), \tag{38}$$

$$r_{\mathrm{norm}} = Z_{\mathrm{norm}}(b), \tag{39}$$

and

$$r_{\mathrm{rand}} = Z_{\mathrm{rand}}(b). \tag{40}$$

Remark 5.2 below constructs a suitable random unit vector b for any prescribed size $\tau = \|r\| = \|Zb\|$ of the residual r (for our examples, we set $\tau = \sqrt{10\varepsilon\kappa}$, where $\varepsilon \approx .2\text{E–}15$ is the machine precision, and κ is the condition number of A or $\tilde{A}$). The values of ρ_{norm} and ρ_{rand} presented for each matrix correspond to the largest obtained over 100 such realizations of b. Small values correspond to high accuracy.

Remark 5.2. To construct a random unit vector b for use in (38), (39), and (40), we select a real number τ such that $0 < \tau < 1$, and we set

$$b = \sqrt{1-\tau^2}\, w + \tau\, x, \tag{41}$$

where w is a random unit vector in the column space of A^* or $\tilde{A}^*$ obtained via (31), and x is a random unit vector in the null space of A or $\tilde{A}$ obtained via (30). Combining (41) and the facts that $Zw = 0$ and $Zx = x$ yields that $\|r\| = \|Zb\| = \tau$. For the numerical experiments reported in the present section, we set $\tau = \sqrt{10\varepsilon\kappa}$, where $\varepsilon \approx .2\text{E–}15$ is the machine precision, and κ is the condition number of A or $\tilde{A}$.

List of notation. The following list defines the headings used in all tables:

- m is the number of rows in the matrix for which we are computing the orthogonal projection onto the null space.
- n is the number of columns in the matrix for which we are computing the orthogonal projection onto the null space.
- l is the number of columns in the random matrix G from (13).
- $\kappa(A)$ (respectively, $\kappa(\tilde{A})$) is the condition number of the matrix A defined in (23) (respectively, the matrix $\tilde{A}$ defined in (24)).

- $\kappa(P^{-1}A)$ (respectively, $\kappa(P^{-1}\tilde{A})$) is the condition number of the matrix $P^{-1}A$ (respectively, $(P^{-1}\tilde{A})$) from Theorem 3.1 and Remark 4.2.
- δ_{norm} is a measure of the error of the standard method of normal equations for orthogonal projection onto a null space, as defined by relation (32). Specifically, δ_{norm} is the Euclidean norm of the result of applying A or $\tilde{A}$ to the orthogonal projection (onto the null space) of a random unit vector b, computed via the method of normal equations. For a given A or $\tilde{A}$, the presented δ_{norm} is the maximum encountered over 100 independent realizations of the random vector b. The tables report δ_{norm} divided by the condition number, since this is generally more informative (cf. Remark 5.3).
- δ_{rand} is a measure of the error of the algorithm of the present paper, as defined by relation (33). Specifically, δ_{rand} is the Euclidean norm of the result of applying A or $\tilde{A}$ to the orthogonal projection (onto the null space) of a random unit vector b, computed via the method of section 4. For a given A or $\tilde{A}$, the presented δ_{rand} is the maximum encountered over 100 independent realizations of the random vector b. The tables report δ_{rand} divided by the condition number, since this is generally more informative (cf. Remark 5.3).
- ϵ_{norm} is a measure of the error of the standard method of normal equations for orthogonal projection onto a null space, as defined by relation (34). Specifically, ϵ_{norm} is the Euclidean norm of the difference between the orthogonal projection via the method of normal equations (onto the null space) of a random unit vector b and its own orthogonal projection computed via the same method. For a given A or $\tilde{A}$, the presented ϵ_{norm} is the maximum encountered over 100 independent realizations of the random vector b. The tables report ϵ_{norm} divided by the condition number, since this is generally more informative (cf. Remark 5.3).
- ϵ_{rand} is a measure of the error of the algorithm of the present paper, as defined by relation (35). Specifically, ϵ_{rand} is the Euclidean norm of the difference between the orthogonal projection via the method of section 4 (onto the null space) of a random unit vector b and its own orthogonal projection computed via the same method. For a given A or $\tilde{A}$, the presented ϵ_{rand} is the maximum encountered over 100 independent realizations of the random vector b. The tables report ϵ_{rand} divided by the condition number, since this is generally more informative (cf. Remark 5.3).
- ρ_{norm} is a measure of the error of the standard method of normal equations for orthogonal projection onto a null space, as defined by relation (36). Specifically, ρ_{norm} is the relative accuracy of the square of the Euclidean norm of the residual to the least-squares problem produced by the standard method of normal equations. Thus ρ_{norm} is the difference between the squared norm of the computed residual and the squared norm of the ideal residual, divided by the squared norm of the ideal residual. The construction of random vectors b with known residuals is discussed in Remark 5.2. For a given A or $\tilde{A}$, the presented ρ_{norm} is the maximum encountered over 100 independent realizations of the random vector b. The tables report ρ_{norm} divided by the condition number, since this is generally more informative (cf. Remark 5.3).
- ρ_{rand} is a measure of the error of the algorithm of the present paper, as defined by relation (37). Specifically, ρ_{rand} is the relative accuracy of the square of the Euclidean norm of the residual to the least-squares problem produced by the method described in section 4. Thus ρ_{rand} is the difference between the squared norm of the computed residual and the squared norm of the ideal

residual, divided by the squared norm of the ideal residual. The construction of random vectors b with known residuals is discussed in Remark 5.2. For a given A or $\tilde{A}$, the presented ρ_{rand} is the maximum encountered over 100 independent realizations of the random vector b. The tables report ρ_{rand} divided by the condition number, since this is generally more informative (cf. Remark 5.3).

- s_{pre} is the time in seconds required to compute $A\,A^*$ (or $\tilde{A}\,\tilde{A}^*$) and its pivoted QR decomposition.
- s_{pro} is the time in seconds required to compute directly the orthogonal projection $b - A^*\,(A\,A^*)^{-1}\,A\,b$ (or $b - \tilde{A}^*\,(\tilde{A}\,\tilde{A}^*)^{-1}\,\tilde{A}\,b$) for a vector b, having already computed a pivoted QR decomposition of $A\,A^*$ (or $\tilde{A}\,\tilde{A}^*$). Thus, $s_{\text{pre}} + j \cdot s_{\text{pro}}$ is the time required by the standard method of normal equations for the orthogonal projection of j vectors onto the null space.
- t_{pre} is the time in seconds required by the randomized algorithm to construct the matrices R, Π, and Y defined in section 4.
- t_{pro} is the time in seconds required by the randomized algorithm to compute the orthogonal projection of a vector onto the null space, having already computed the matrices R, Π, and Y defined in section 4. Thus, $t_{\text{pre}} + j \cdot t_{\text{pro}}$ is the time required by the algorithm of the present paper for orthogonally projecting j vectors onto the null space.

Remark 5.3. Standard perturbation theory shows that, given a vector b, a perturbation of size Δ to the entries of a short, fat matrix A satisfying (27) may cause the entries of the vector h minimizing $\|A^*h - b\|$ to change by about $\Delta \cdot \kappa^2 \cdot \tau$, where $\kappa = \kappa(A)$ is the condition number of A, and $\tau = \|r\| = \|A^*h - b\|$ is the size of the residual $r = b - A^*h$ (see, for example, formula 1.4.26 in [2]). Thus, it is generally meaningless to compute h more accurately than $\varepsilon \cdot \kappa^2 \cdot \tau$, where ε is the machine precision, $\kappa = \kappa(A)$ is the condition number of A, and $\tau = \|A^*h - b\|$. Similarly, it is generally meaningless to compute the orthogonal projection onto the null space of A more accurately than $\varepsilon \cdot \kappa$. Consequently, any entry in the tables for $\delta_{\text{norm}}/\kappa$, $\delta_{\text{rand}}/\kappa$, $\epsilon_{\text{norm}}/\kappa$, $\epsilon_{\text{rand}}/\kappa$, $\rho_{\text{norm}}/\kappa$, or $\rho_{\text{rand}}/\kappa$ which is less than the machine precision $\varepsilon \approx$ .2E–15 should not be construed as a meaningful improvement over an accuracy of ε.

5.4. Numerical results. The following tables present numerical results for the projection of test vectors onto the null spaces of matrices A and $\tilde{A}$. Tables 1–7 are organized for the purposes of clear exposition. Their content is provided unabridged in Tables 8–10.

Table 1 provides results for comparison of the accuracy of the algorithm using Gaussian random variables for the entries of G in (13) versus using uniformly distributed random variables. No appreciable differences in accuracy are observable.

Table 2 demonstrates that the randomized algorithm runs substantially faster when implemented using uniformly distributed random variables generated via (Mitchell–Moore–Brent–Knuth) lagged Fibonacci pseudorandom sequences in place of high-quality Gaussian pseudorandom variables. For this reason, we present all of the following results for the algorithm using uniformly distributed random variables.

Remark 5.4. Table 1 illustrates that there is no substantive benefit from using high-quality pseudorandom numbers for the entries of G in (13), nor from using random variables with a Gaussian distribution.

Table 3 reports the accuracy of the randomized algorithm with varying values of l and m. Specifically, the values $l = m + 4$, $l = m + 100$, $l = 3m/2$, and $l = 2m$ are shown. Clearly, the condition number $\kappa(P^{-1}A)$ rapidly decreases as l increases.

TABLE 1
Errors for the sparse matrix A defined in (23) *using Gaussian and uniformly distributed random variables, with $l = m + 4$ and $n = 10^5$.*

		Gaussian				Uniformly distributed			
m	$\kappa(A)$	$\kappa(P^{-1}A)$	$\frac{\delta_{\text{rand}}}{\kappa(A)}$	$\frac{\epsilon_{\text{rand}}}{\kappa(A)}$	$\frac{\rho_{\text{rand}}}{\kappa(A)}$	$\kappa(P^{-1}A)$	$\frac{\delta_{\text{rand}}}{\kappa(A)}$	$\frac{\epsilon_{\text{rand}}}{\kappa(A)}$	$\frac{\rho_{\text{rand}}}{\kappa(A)}$
500	1E4	.32E3	.91E–17	.59E–15	.88E–13	.31E3	.82E–17	.58E–15	.69E–13
500	1E5	.32E3	.58E–17	.63E–15	.47E–13	.45E3	.66E–17	.52E–15	.45E–13
500	1E6	.31E3	.49E–17	.41E–15	.57E–13	.34E3	.52E–17	.50E–15	.41E–13
500	1E7	.31E3	.40E–17	.41E–15	.30E–13	.24E3	.47E–17	.39E–15	.24E–13
500	1E8	.33E3	.28E–17	.34E–15	.21E–13	.27E3	.27E–17	.37E–15	.21E–13
1000	1E4	.60E3	.19E–16	.42E–15	.20E–13	.10E4	.20E–16	.44E–15	.22E–13
1000	1E5	.13E4	.13E–16	.40E–15	.14E–13	.15E4	.15E–16	.40E–15	.12E–13
1000	1E6	.97E3	.11E–16	.30E–15	.72E–14	.51E3	.11E–16	.27E–15	.93E–14
1000	1E7	.61E3	.89E–17	.24E–15	.56E–14	.16E4	.77E–17	.24E–15	.44E–14
1000	1E8	.70E3	.62E–17	.22E–15	.41E–14	.12E4	.64E–17	.17E–15	.32E–14
2000	1E4	.16E4	.39E–16	.32E–15	.55E–14	.16E4	.35E–16	.30E–15	.48E–14
2000	1E5	.14E4	.28E–16	.25E–15	.39E–14	.14E4	.26E–16	.23E–15	.31E–14
2000	1E6	.93E3	.20E–16	.21E–15	.19E–14	.10E4	.18E–16	.19E–15	.20E–14
2000	1E7	.16E4	.16E–16	.16E–15	.12E–14	.12E4	.15E–16	.15E–15	.14E–14
2000	1E8	.10E4	.10E–16	.12E–15	.87E–15	.16E4	.11E–16	.14E–15	.93E–15

TABLE 2
Timings for the sparse matrix A defined in (23) *using Gaussian and uniformly distributed random variables, with $n = 10^5$.*

m	l	$\kappa(A)$	$t_{\text{pre}}^{(\text{Gaussian})}$	$t_{\text{pre}}^{(\text{Uniform})}$
500	504	1E6	.85E2	.23E2
500	504	1E7	.86E2	.23E2
500	504	1E8	.85E2	.23E2
1000	1004	1E6	.18E3	.61E2
1000	1004	1E7	.18E3	.62E2
1000	1004	1E8	.18E3	.61E2
2000	1004	1E6	.52E3	.27E3
2000	1004	1E7	.52E3	.27E3
2000	1004	1E8	.51E3	.27E3

TABLE 3
Errors for the sparse matrix A defined in (23) *with varying values of l, with $n = 10^5$.*

m	l	$\kappa(A)$	$\kappa(P^{-1}A)$	$\delta_{\text{rand}}/\kappa(A)$	$\epsilon_{\text{rand}}/\kappa(A)$	$\rho_{\text{rand}}/\kappa(A)$
1000	1004	1E6	.51E3	.11E–16	.27E–15	.93E–14
1000	1100	1E6	.37E2	.76E–17	.19E–15	.34E–14
1000	1500	1E6	.97E1	.73E–17	.15E–15	.20E–14
1000	2000	1E6	.57E1	.65E–17	.14E–15	.16E–14
1000	1004	1E8	.12E4	.64E–17	.17E–15	.32E–14
1000	1100	1E8	.39E2	.57E–17	.12E–15	.20E–14
1000	1500	1E8	.96E1	.44E–17	.10E–15	.96E–15
1000	2000	1E8	.57E1	.41E–17	.84E–16	.11E–14
2000	2004	1E6	.10E4	.18E–16	.19E–15	.20E–14
2000	2100	1E6	.77E2	.14E–16	.16E–15	.11E–14
2000	3000	1E6	.99E1	.12E–16	.11E–15	.56E–15
2000	4000	1E6	.58E1	.11E–16	.97E–16	.36E–15
2000	2004	1E8	.16E4	.11E–16	.14E–15	.93E–15
2000	2100	1E8	.81E2	.89E–17	.10E–15	.51E–15
2000	3000	1E8	.98E1	.88E–17	.60E–16	.19E–15
2000	4000	1E8	.57E1	.86E–17	.55E–16	.13E–15

TABLE 4
Errors for the sparse matrix A defined in (23) *with $l = m + 4$ and $n = 10^6$.*

m	$\kappa(A)$	$\kappa(P^{-1}A)$	$\frac{\delta_{\text{norm}}}{\kappa(A)}$	$\frac{\delta_{\text{rand}}}{\kappa(A)}$	$\frac{\epsilon_{\text{norm}}}{\kappa(A)}$	$\frac{\epsilon_{\text{rand}}}{\kappa(A)}$	$\frac{\rho_{\text{norm}}}{\kappa(A)}$	$\frac{\rho_{\text{rand}}}{\kappa(A)}$
1000	1E4	.11E4	.61E–15	.59E–17	.56E–11	.14E–14	.28E–04	.18E–11
1000	1E5	.11E4	.42E–15	.45E–17	.37E–10	.13E–14	.15E–02	.12E–11
1000	1E6	.50E3	.36E–15	.28E–17	.31E–09	.91E–15	.11E–00	.79E–12
1000	1E7	.50E3	.52E–16	.25E–17	.12E–09	.88E–15	.15E–00	.57E–12
1000	1E8	.51E3	.43E–17	.19E–17	.88E–11	.58E–15	.24E–02	.28E–12
1000	1E9	.51E3	.53E–18	.13E–17	.89E–12	.48E–15	.26E–04	.21E–12
1000	1E10	.15E4	.37E–19	.84E–18	.99E–13	.34E–15	.25E–06	.14E–12
2000	1E4	.10E4	.64E–15	.11E–16	.57E–11	.93E–15	.16E–04	.44E–12
2000	1E5	.14E4	.52E–15	.92E–17	.48E–10	.75E–15	.11E–02	.27E–12
2000	1E6	.12E4	.36E–15	.59E–17	.35E–09	.60E–15	.53E–01	.15E–12
2000	1E7	.99E3	.17E–14	.45E–17	.14E–06	.51E–15	.18E+02	.13E–12
2000	1E8	.99E3	.32E–16	.30E–17	.57E–09	.37E–15	.14E–01	.80E–13
2000	1E9	.16E4	.27E–17	.26E–17	.44E–10	.37E–15	.14E–03	.54E–13
2000	1E10	.91E3	.65E–18	.20E–17	.34E–10	.25E–15	.53E–05	.36E–13
4000	1E4	.27E4	.33E–15	.25E–16	.30E–11	.68E–15	.23E–05	.12E–12
4000	1E5	.33E4	.17E–15	.18E–16	.16E–10	.54E–15	.58E–04	.66E–13
4000	1E6	.37E4	.78E–16	.12E–16	.69E–10	.42E–15	.17E–02	.46E–13
4000	1E7	.30E4	.66E–16	.10E–16	.21E–09	.36E–15	.69E–01	.29E–13
4000	1E8	.31E4	.48E–17	.71E–17	.14E–10	.28E–15	.17E–02	.19E–13
4000	1E9	.19E4	.44E–18	.49E–17	.16E–11	.28E–15	.16E–04	.19E–13
4000	1E10	.19E4	.42E–19	.45E–17	.14E–12	.17E–15	.17E–06	.95E–14

TABLE 5
Timings for the sparse matrix A defined in (23) *with $l = m + 4$ and $n = 10^6$.*

m	$\kappa(A)$	$\kappa(P^{-1}A)$	s_{pre}	t_{pre}	s_{pro}	t_{pro}
1000	1E4	.11E4	.59E3	.88E3	.93E0	.95E0
1000	1E6	.50E3	.56E3	.86E3	.91E0	.92E0
1000	1E8	.51E3	.54E3	.82E3	.86E0	.87E0
1000	1E10	.15E4	.54E3	.81E3	.86E0	.87E0
2000	1E4	.10E4	.11E4	.18E4	.89E0	.94E0
2000	1E6	.12E4	.11E4	.18E4	.89E0	.94E0
2000	1E8	.99E3	.11E4	.18E4	.89E0	.94E0
2000	1E10	.91E3	.11E4	.18E4	.89E0	.94E0
4000	1E4	.27E4	.24E4	.52E4	.13E1	.18E1
4000	1E6	.37E4	.23E4	.54E4	.11E1	.17E1
4000	1E8	.31E4	.23E4	.52E4	.11E1	.17E1
4000	1E10	.19E4	.23E4	.52E4	.12E1	.17E1

However, the accuracies δ_{rand}, ϵ_{rand}, and ρ_{rand} do not improve much when $\kappa(P^{-1}A)$ tends to 1 as l increases beyond $m + 4$. In particular, $l = m + 4$ yields essentially optimal accuracies δ_{rand}, ϵ_{rand}, and ρ_{rand} well before $\kappa(P^{-1}A)$ approaches unity. At the same time, the computational costs of the algorithm increase as l increases. For these reasons, in the following tables we use $l = m + 4$ for comparison against the standard method of normal equations.

Table 4 presents errors for the standard method of normal equations as well as the randomized method of section 4 for projection onto the null space of the matrix A defined by (23). The corresponding timings are provided in Table 5.

Table 6 presents errors for the standard method of normal equations as well as the randomized method of section 4 for projection onto the null space of the matrix $\tilde{A}$ defined by (24). The corresponding timings are provided in Table 7.

Table 6
Errors for the dense matrix $\tilde{A}$ *defined in* (24) *with* $l = m + 4$ *and* $n = 10^6$.

m	$\kappa(\tilde{A})$	$\kappa(P^{-1}\tilde{A})$	$\frac{\delta_{\text{norm}}}{\kappa(\tilde{A})}$	$\frac{\delta_{\text{rand}}}{\kappa(\tilde{A})}$	$\frac{\epsilon_{\text{norm}}}{\kappa(\tilde{A})}$	$\frac{\epsilon_{\text{rand}}}{\kappa(\tilde{A})}$	$\frac{\rho_{\text{norm}}}{\kappa(\tilde{A})}$	$\frac{\rho_{\text{rand}}}{\kappa(\tilde{A})}$
1000	1E4	.11E4	.30E–15	.64E–17	.81E–12	.13E–14	.69E–06	.36E–11
1000	1E5	.11E4	.24E–15	.44E–17	.52E–11	.11E–14	.26E–04	.13E–11
1000	1E6	.50E3	.18E–15	.34E–17	.31E–10	.89E–15	.99E–03	.70E–12
1000	1E7	.50E3	.43E–15	.29E–17	.15E–07	.75E–15	.12E+02	.57E–12
1000	1E8	.51E3	.25E–15	.18E–17	.45E–09	.73E–15	.57E–01	.40E–12
1000	1E9	.51E3	.83E–17	.12E–17	.28E–11	.47E–15	.48E–04	.20E–12
1000	1E10	.15E4	.73E–18	.69E–18	.44E–12	.50E–15	.38E–06	.36E–12
2000	1E4	.10E4	.20E–15	.12E–16	.55E–12	.94E–15	.15E–06	.46E–12
2000	1E5	.14E4	.16E–15	.98E–17	.32E–11	.78E–15	.48E–05	.33E–12
2000	1E6	.12E4	.12E–15	.62E–17	.20E–10	.60E–15	.22E–03	.22E–12
2000	1E7	.99E3	.98E–16	.60E–17	.19E–09	.49E–15	.36E–01	.13E–12
2000	1E8	.99E3	.98E–16	.35E–17	.46E–09	.40E–15	.19E–01	.81E–13
2000	1E9	.16E4	.13E–16	.27E–17	.12E–10	.36E–15	.71E–04	.48E–13
2000	1E10	.91E3	.22E–17	.23E–17	.39E–11	.24E–15	.21E–05	.38E–13
4000	1E4	.27E4	.14E–15	.28E–16	.37E–12	.66E–15	.35E–07	.14E–12
4000	1E5	.33E4	.11E–15	.20E–16	.22E–11	.53E–15	.12E–05	.68E–13
4000	1E6	.37E4	.81E–16	.14E–16	.13E–10	.43E–15	.46E–04	.53E–13
4000	1E7	.30E4	.65E–16	.10E–16	.84E–10	.33E–15	.16E–02	.35E–13
4000	1E8	.31E4	.28E–15	.63E–17	.18E–08	.28E–15	.97E–01	.16E–13
4000	1E9	.19E4	.19E–15	.53E–17	.41E–08	.25E–15	.31E–01	.13E–13
4000	1E10	.19E4	.24E–17	.37E–17	.16E–10	.20E–15	.53E–05	.82E–14

Table 7
Timings for the dense matrix $\tilde{A}$ *defined in* (24) *with* $l = m + 4$ *and* $n = 10^6$.

m	$\kappa(\tilde{A})$	$\kappa(P^{-1}\tilde{A})$	s_{pre}	t_{pre}	s_{pro}	t_{pro}
1000	1E4	.11E4	.54E3	.88E3	.84E0	.99E0
1000	1E6	.50E3	.54E3	.80E3	.86E0	.88E0
1000	1E8	.51E3	.53E3	.81E3	.86E0	.87E0
1000	1E10	.15E4	.53E3	.81E3	.86E0	.87E0
2000	1E4	.10E4	.11E4	.18E4	.89E0	.94E0
2000	1E6	.12E4	.11E4	.18E4	.89E0	.94E0
2000	1E8	.99E3	.11E4	.18E4	.89E0	.94E0
2000	1E10	.91E3	.11E4	.18E4	.89E0	.94E0
4000	1E4	.27E4	.26E4	.51E4	.12E1	.16E1
4000	1E6	.37E4	.23E4	.55E4	.11E1	.16E1
4000	1E8	.31E4	.23E4	.51E4	.11E1	.16E1
4000	1E10	.19E4	.23E4	.51E4	.11E1	.16E1

6. Discussion and conclusions.

The numerical results reported in section 5, as well as our further experiments, all indicate that the timings of the classical scheme and the randomized method are reasonably similar for our implementations, whereas the randomized method produces more accurate projections. For matrices with high condition numbers, the randomized method consistently outperforms the standard method of normal equations for orthogonal projection onto a null space:

- First, with respect to the standard method of normal equations, the randomized method is a projection (i.e., idempotent) to higher accuracy. Section 5 illustrates this via ϵ_{norm} and ϵ_{rand} defined in (34) and (35).
- Second, with respect to the standard method of normal equations, the randomized method produces smaller residuals for the corresponding least-squares problem. Section 5 illustrates this via ρ_{norm} and ρ_{rand} defined in (36) and (37).

TABLE 8
Results for the sparse matrix A defined by (23) with Gaussian random variables.

n	m	l	$\kappa(A)$	$\kappa(P^{-1}A)$	s_{pre}	t_{pre}	s_{pro}	t_{pro}	$\delta_{\text{norm}}/\kappa$	$\delta_{\text{rand}}/\kappa$	$\epsilon_{\text{norm}}/\kappa$	$\epsilon_{\text{rand}}/\kappa$	$\rho_{\text{norm}}/\kappa$	$\rho_{\text{rand}}/\kappa$
100000	500	504	1E4	.32E3	.13E2	.86E2	.50E–01	.54E–01	.42E–15	.91E–17	.39E–11	.59E–15	.28E–05	.88E–13
100000	500	504	1E5	.32E3	.14E2	.86E2	.52E–01	.58E–01	.27E–15	.58E–17	.25E–10	.63E–15	.12E–03	.47E–13
100000	500	504	1E6	.31E3	.12E2	.85E2	.50E–01	.54E–01	.27E–15	.49E–17	.25E–09	.41E–15	.14E–01	.57E–13
100000	500	504	1E7	.31E3	.14E2	.86E2	.51E–01	.57E–01	.59E–15	.40E–17	.15E–07	.41E–15	.28E+01	.30E–13
100000	500	504	1E8	.33E3	.13E2	.85E2	.50E–01	.54E–01	.36E–16	.28E–17	.88E–09	.34E–15	.12E–01	.21E–13
100000	1000	1004	1E4	.60E3	.27E2	.18E3	.58E–01	.70E–01	.17E–15	.19E–16	.15E–11	.42E–15	.18E–06	.20E–13
100000	1000	1004	1E5	.13E4	.30E2	.18E3	.60E–01	.72E–01	.82E–16	.13E–16	.74E–11	.40E–15	.57E–05	.14E–13
100000	1000	1004	1E6	.97E3	.27E2	.18E3	.58E–01	.69E–01	.81E–16	.11E–16	.75E–10	.30E–15	.55E–03	.72E–14
100000	1000	1004	1E7	.61E3	.29E2	.18E3	.60E–01	.72E–01	.32E–16	.89E–17	.24E–09	.24E–15	.10E–01	.56E–14
100000	1000	1004	1E8	.70E3	.27E2	.18E3	.58E–01	.69E–01	.21E–17	.62E–17	.28E–10	.22E–15	.17E–02	.41E–14
100000	2000	2004	1E4	.16E4	.68E2	.51E3	.94E–01	.14E–00	.53E–16	.39E–16	.44E–12	.32E–15	.11E–07	.55E–14
100000	2000	2004	1E5	.14E4	.73E2	.52E3	.94E–01	.14E–00	.20E–16	.28E–16	.46E–12	.25E–15	.11E–07	.39E–14
100000	2000	2004	1E6	.93E3	.68E2	.51E3	.92E–01	.14E–00	.42E–16	.20E–16	.35E–10	.21E–15	.65E–04	.19E–14
100000	2000	2004	1E7	.16E4	.74E2	.52E3	.94E–01	.14E–00	.15E–16	.16E–16	.88E–10	.16E–15	.32E–03	.12E–14
100000	2000	2004	1E8	.10E4	.67E2	.51E3	.92E–01	.14E–00	.30E–17	.10E–16	.35E–10	.12E–15	.12E–02	.87E–15

TABLE 9
Results for the dense matrix $\tilde{A}$ defined by (24) with uniformly distributed random variables.

n	m	l	$\kappa(A)$	$\kappa(P^{-1}A)$	s_{pre}	t_{pre}	s_{pro}	t_{pro}	$\delta_{\text{norm}}/\kappa$	$\delta_{\text{rand}}/\kappa$	$\epsilon_{\text{norm}}/\kappa$	$\epsilon_{\text{rand}}/\kappa$	$\rho_{\text{norm}}/\kappa$	$\rho_{\text{rand}}/\kappa$
1000000	1000	1004	1E4	.11E4	.54E3	.88E3	.84E0	.99E0	.30E–15	.64E–17	.81E–12	.13E–14	.69E–06	.36E–11
1000000	1000	1004	1E5	.11E4	.54E3	.81E3	.87E0	.87E0	.24E–15	.44E–17	.52E–11	.11E–14	.26E–04	.13E–11
1000000	1000	1004	1E6	.50E3	.54E3	.80E3	.86E0	.88E0	.18E–15	.34E–17	.31E–10	.89E–15	.99E–03	.70E–12
1000000	1000	1004	1E7	.50E3	.54E3	.81E3	.87E0	.87E0	.43E–15	.29E–17	.15E–07	.75E–15	.12E+02	.57E–12
1000000	1000	1004	1E8	.51E3	.53E3	.81E3	.86E0	.87E0	.25E–15	.18E–17	.45E–09	.73E–15	.57E–01	.40E–12
1000000	1000	1004	1E9	.51E3	.54E3	.81E3	.86E0	.87E0	.83E–17	.12E–17	.28E–11	.47E–15	.48E–04	.20E–12
1000000	1000	1004	1E10	.15E4	.53E3	.81E3	.86E0	.87E0	.73E–18	.69E–18	.44E–12	.50E–15	.38E–06	.36E–12
1000000	2000	2004	1E4	.10E4	.11E4	.18E4	.89E0	.94E0	.20E–15	.12E–16	.55E–12	.94E–15	.15E–06	.46E–12
1000000	2000	2004	1E5	.14E4	.11E4	.18E4	.90E0	.92E0	.16E–15	.98E–17	.32E–11	.78E–15	.48E–05	.33E–12
1000000	2000	2004	1E6	.12E4	.11E4	.18E4	.89E0	.94E0	.12E–15	.62E–17	.20E–10	.60E–15	.22E–03	.22E–12
1000000	2000	2004	1E7	.99E3	.11E4	.19E4	.91E0	.10E1	.98E–16	.60E–17	.19E–09	.49E–15	.36E–01	.13E–12
1000000	2000	2004	1E8	.99E3	.11E4	.18E4	.89E0	.94E0	.98E–16	.35E–17	.46E–09	.40E–15	.19E–01	.81E–13
1000000	2000	2004	1E9	.16E4	.12E4	.19E4	.91E0	.95E0	.13E–16	.27E–17	.12E–10	.36E–15	.71E–04	.48E–13
1000000	2000	2004	1E10	.91E3	.11E4	.18E4	.89E0	.94E0	.22E–17	.23E–17	.39E–11	.24E–15	.21E–05	.38E–13
1000000	4000	4004	1E4	.27E4	.26E4	.51E4	.12E1	.16E1	.14E–15	.28E–16	.37E–12	.66E–15	.35E–07	.14E–12
1000000	4000	4004	1E5	.33E4	.23E4	.53E3	.11E1	.16E1	.11E–15	.20E–16	.22E–11	.53E–15	.12E–05	.68E–13
1000000	4000	4004	1E6	.37E4	.23E4	.55E4	.11E1	.16E1	.81E–16	.14E–16	.13E–10	.43E–15	.46E–04	.53E–13
1000000	4000	4004	1E7	.30E4	.23E4	.51E3	.11E1	.16E1	.65E–16	.10E–16	.84E–10	.33E–15	.16E–02	.35E–13
1000000	4000	4004	1E8	.31E4	.23E4	.51E4	.11E1	.16E1	.28E–15	.63E–17	.18E–08	.28E–15	.97E–01	.16E–13
1000000	4000	4004	1E9	.19E4	.24E4	.51E3	.11E1	.16E1	.19E–15	.53E–17	.41E–08	.25E–15	.31E–01	.13E–13
1000000	4000	4004	1E10	.19E4	.23E4	.51E4	.11E1	.16E1	.24E–17	.37E–17	.16E–10	.20E–15	.53E–05	.82E–14

Table 10
Results for the sparse matrix A defined by (23) *with uniformly distributed random variables.*

n	m	l	$\kappa(A)$	$\kappa(P^{-1}A)$	s_{pre}	t_{pre}	s_{pro}	t_{pro}	$\delta_{\text{norm}}/\kappa$	$\delta_{\text{rand}}/\kappa$	$\epsilon_{\text{norm}}/\kappa$	$\epsilon_{\text{rand}}/\kappa$	$\rho_{\text{norm}}/\kappa$	$\rho_{\text{rand}}/\kappa$
100000	500	504	1E4	.31E3	.14E2	.23E2	.46E–01	.50E–01	.42E–15	.82E–17	.39E–11	.58E–15	.28E–05	.69E–13
100000	500	504	1E5	.45E3	.14E2	.23E2	.46E–01	.50E–01	.28E–15	.66E–17	.26E–10	.52E–15	.13E–03	.45E–13
100000	500	504	1E6	.34E3	.14E2	.23E2	.46E–01	.50E–01	.28E–15	.52E–17	.26E–09	.50E–15	.12E–01	.41E–13
100000	500	504	1E7	.24E3	.14E2	.23E2	.46E–01	.50E–01	.66E–15	.47E–17	.17E–07	.39E–15	.32E+01	.24E–13
100000	500	504	1E8	.27E3	.14E2	.23E2	.46E–01	.50E–01	.35E–16	.27E–17	.87E–09	.37E–15	.11E–01	.21E–13
100000	1000	1004	1E4	.10E4	.31E2	.61E2	.54E–01	.66E–01	.15E–15	.20E–16	.14E–11	.44E–15	.18E–06	.22E–13
100000	1000	1004	1E5	.15E4	.31E2	.61E2	.54E–01	.66E–01	.83E–16	.15E–16	.76E–11	.40E–15	.57E–05	.12E–13
100000	1000	1004	1E6	.51E3	.30E2	.62E2	.54E–01	.66E–01	.72E–16	.11E–16	.67E–10	.27E–15	.52E–03	.93E–14
100000	1000	1004	1E7	.16E4	.30E2	.61E2	.54E–01	.66E–01	.32E–16	.77E–17	.24E–09	.24E–15	.83E–02	.44E–14
100000	1000	1004	1E8	.12E4	.31E2	.61E2	.54E–01	.66E–01	.23E–17	.64E–17	.26E–10	.17E–15	.18E–02	.32E–14
100000	2000	2004	1E4	.16E4	.75E2	.27E3	.88E–01	.14E–00	.54E–16	.35E–16	.45E–12	.30E–15	.94E–08	.48E–14
100000	2000	2004	1E5	.14E4	.75E2	.27E3	.88E–01	.14E–00	.18E–16	.26E–16	.49E–12	.23E–15	.13E–07	.31E–14
100000	2000	2004	1E6	.10E4	.75E2	.27E3	.88E–01	.14E–00	.44E–16	.18E–16	.37E–10	.19E–15	.64E–04	.20E–14
100000	2000	2004	1E7	.12E4	.75E2	.27E3	.88E–01	.14E–00	.15E–16	.15E–16	.87E–10	.15E–15	.31E–03	.14E–14
100000	2000	2004	1E8	.16E4	.75E2	.27E3	.88E–01	.14E–00	.30E–17	.11E–16	.38E–10	.14E–15	.93E–03	.93E–15
100000	1000	1100	1E6	.37E2	.30E2	.71E2	.57E–01	.68E–01	.76E–16	.76E–17	.70E–10	.19E–15	.43E–03	.34E–14
100000	1000	1500	1E6	.97E1	.30E2	.79E2	.58E–01	.69E–01	.82E–16	.73E–17	.75E–10	.15E–15	.54E–03	.20E–14
100000	1000	2000	1E6	.57E1	.30E2	.90E2	.57E–01	.69E–01	.79E–16	.65E–17	.71E–10	.14E–15	.50E–03	.16E–14
100000	1000	1100	1E7	.37E2	.26E2	.66E2	.52E–01	.66E–01	.32E–16	.70E–17	.24E–09	.18E–15	.71E–02	.33E–14
100000	1000	1500	1E7	.97E1	.26E2	.72E2	.53E–01	.64E–01	.33E–16	.54E–17	.23E–09	.13E–15	.10E–01	.18E–14
100000	1000	2000	1E7	.57E1	.26E2	.84E2	.53E–01	.65E–01	.31E–16	.52E–17	.23E–09	.11E–15	.97E–02	.88E–15
100000	1000	1100	1E8	.39E2	.30E2	.70E2	.58E–01	.69E–01	.23E–17	.57E–17	.30E–10	.12E–15	.19E–02	.20E–14
100000	1000	1500	1E8	.96E1	.30E2	.77E2	.57E–01	.70E–01	.21E–17	.44E–17	.26E–10	.10E–15	.20E–02	.96E–15
100000	1000	2000	1E8	.57E1	.30E2	.89E2	.57E–01	.69E–01	.24E–17	.41E–17	.31E–10	.84E–16	.17E–02	.11E–14
100000	1000	1100	1E9	.39E2	.26E2	.61E2	.52E–01	.65E–01	.19E–18	.47E–17	.24E–11	.13E–15	.25E–04	.13E–14
100000	1000	1500	1E9	.96E1	.26E2	.72E2	.53E–01	.65E–01	.17E–18	.41E–17	.22E–11	.92E–16	.26E–04	.83E–15
100000	1000	2000	1E9	.57E1	.26E2	.82E2	.52E–01	.66E–01	.21E–18	.33E–17	.27E–11	.72E–16	.24E–04	.55E–15
100000	2000	2100	1E6	.77E2	.77E2	.30E3	.96E–01	.14E–00	.42E–16	.14E–16	.36E–10	.16E–15	.61E–04	.11E–14
100000	2000	3000	1E6	.99E1	.77E2	.35E3	.97E–01	.15E–00	.42E–16	.12E–16	.37E–10	.11E–15	.59E–04	.56E–15
100000	2000	4000	1E6	.58E1	.73E2	.39E3	.91E–01	.15E–00	.44E–16	.11E–16	.37E–10	.97E–16	.72E–04	.36E–15
100000	2000	2100	1E7	.77E2	.66E2	.29E3	.88E–01	.14E–00	.15E–16	.11E–16	.94E–10	.12E–15	.34E–03	.66E–15
100000	2000	3000	1E7	.99E1	.65E2	.32E3	.87E–01	.14E–00	.14E–16	.94E–17	.86E–10	.82E–16	.32E–03	.29E–15
100000	2000	4000	1E7	.58E1	.66E2	.36E3	.87E–01	.15E–00	.15E–16	.91E–17	.92E–10	.70E–16	.41E–03	.26E–15
100000	2000	2100	1E8	.81E2	.73E2	.28E3	.91E–01	.14E–00	.31E–17	.89E–17	.38E–10	.10E–15	.12E–02	.51E–15
100000	2000	3000	1E8	.98E1	.73E2	.33E3	.91E–01	.14E–00	.31E–17	.88E–17	.38E–10	.60E–16	.11E–02	.19E–15
100000	2000	4000	1E8	.57E1	.73E2	.37E3	.94E–01	.15E–00	.32E–17	.86E–17	.42E–10	.55E–16	.12E–02	.13E–15
100000	2000	2100	1E9	.81E2	.66E2	.27E3	.87E–01	.14E–00	.23E–18	.73E–17	.29E–11	.10E–15	.18E–04	.29E–15
100000	2000	3000	1E9	.98E1	.65E2	.32E3	.87E–01	.14E–00	.26E–18	.83E–17	.29E–11	.59E–16	.16E–04	.14E–15
100000	2000	4000	1E9	.57E1	.66E2	.36E3	.87E–01	.15E–00	.29E–18	.61E–17	.37E–11	.42E–16	.17E–04	.11E–15
1000000	1000	1004	1E4	.11E4	.59E3	.88E3	.93E–00	.95E–00	.61E–15	.59E–17	.56E–11	.14E–14	.28E–04	.18E–11
1000000	1000	1004	1E5	.11E4	.53E3	.83E3	.85E–00	.88E–00	.42E–15	.45E–17	.37E–10	.13E–14	.15E–02	.12E–11
1000000	1000	1004	1E6	.50E3	.56E3	.86E3	.91E–00	.92E–00	.36E–15	.28E–17	.31E–09	.91E–15	.11E–00	.79E–12
1000000	1000	1004	1E7	.50E3	.57E3	.82E3	.95E–00	.88E–00	.52E–16	.25E–17	.12E–09	.88E–15	.15E–00	.57E–12
1000000	1000	1004	1E8	.51E3	.54E3	.82E3	.86E–00	.87E–00	.43E–17	.19E–17	.88E–11	.58E–15	.24E–02	.28E–12
1000000	1000	1004	1E9	.51E3	.60E3	.84E3	.95E–00	.95E–00	.53E–18	.13E–17	.89E–12	.48E–15	.26E–04	.21E–12
1000000	1000	1004	1E10	.15E4	.54E3	.81E3	.86E–00	.87E–00	.37E–19	.84E–18	.99E–13	.34E–15	.25E–06	.14E–12
1000000	2000	2004	1E4	.10E4	.11E4	.18E4	.89E–00	.94E–00	.64E–15	.11E–16	.57E–11	.93E–15	.16E–04	.44E–12
1000000	2000	2004	1E5	.14E4	.12E4	.20E4	.98E–00	.10E+01	.52E–15	.92E–17	.48E–10	.75E–15	.11E–02	.27E–12
1000000	2000	2004	1E6	.12E4	.11E4	.18E4	.89E–00	.94E–00	.36E–15	.59E–17	.35E–09	.60E–15	.53E–01	.15E–12
1000000	2000	2004	1E7	.99E3	.11E4	.19E4	.90E–00	.94E–00	.17E–14	.45E–17	.14E–06	.51E–15	.18E+02	.13E–12
1000000	2000	2004	1E8	.99E3	.11E4	.18E4	.89E–00	.94E–00	.32E–16	.30E–17	.57E–09	.37E–15	.14E–01	.80E–13
1000000	2000	2004	1E9	.16E4	.11E4	.18E4	.91E–00	.94E–00	.27E–17	.26E–17	.44E–10	.37E–15	.14E–03	.54E–13
1000000	2000	2004	1E10	.91E3	.11E4	.18E4	.89E–00	.94E–00	.65E–18	.20E–17	.34E–10	.25E–15	.53E–05	.36E–13
1000000	4000	4004	1E4	.27E4	.24E4	.12E4	.13E+01	.18E+01	.33E–15	.25E–16	.30E–11	.68E–15	.23E–05	.12E–12
1000000	4000	4004	1E5	.33E4	.23E4	.11E4	.11E+01	.16E+01	.17E–15	.18E–16	.16E–10	.54E–15	.58E–04	.66E–13
1000000	4000	4004	1E6	.37E4	.23E4	.14E4	.11E+01	.17E+01	.78E–16	.12E–16	.69E–10	.42E–15	.17E–02	.46E–13
1000000	4000	4004	1E7	.30E4	.23E4	.11E4	.11E+01	.16E+01	.66E–16	.10E–16	.21E–09	.36E–15	.69E–01	.29E–13
1000000	4000	4004	1E8	.31E4	.23E4	.12E4	.11E+01	.17E+01	.48E–17	.71E–17	.14E–10	.28E–15	.17E–02	.19E–13
1000000	4000	4004	1E9	.19E4	.23E4	.17E4	.11E+01	.16E+01	.44E–18	.49E–17	.16E–11	.28E–15	.16E–04	.19E–13
1000000	4000	4004	1E10	.19E4	.23E4	.12E4	.12E+01	.17E+01	.42E–19	.45E–17	.14E–12	.17E–15	.17E–06	.95E–14

- Both methods appear to perform consistently and equally well in producing vectors which are indeed in the null space of the specified matrix. Section 5 illustrates this via δ_{norm} and δ_{rand} defined in (32) and (33).

Both theoretical and empirical considerations indicate that setting $l = m + 4$ in (13) is generally sufficient. Moreover, the quality of the pseudorandom numbers does not affect the accuracy of the algorithm much, if at all, nor does replacing the Gaussian random variables with uniformly distributed random variables for the entries of G in (13). In practice, one can simply use the fastest pseudorandom number generator available.

Acknowledgments. We would like to thank Professor Ming Gu of UC Berkeley for sharing his insights and particularly for pointing out the application to constrained optimization mentioned in our abstract. We thank the anonymous referees for their useful suggestions.

REFERENCES

[1] H. Avron, P. Maymounkov, and S. Toledo, *Blendenpik: Supercharging LAPACK's least-squares solver*, SIAM J. Sci. Comput., 32 (2010), pp. 1217–1236.
[2] Å. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.
[3] Å. Björck and G. Dahlquist, *Numerical Methods*, Prentice–Hall, Englewood Cliffs, NJ, 1974.
[4] Z. Chen and J. J. Dongarra, *Condition numbers of Gaussian random matrices*, SIAM J. Matrix Anal. Appl., 27 (2005), pp. 603–620.
[5] P. Drineas, M. W. Mahoney, S. Muthukrishnan, and T. Sarlós, *Faster least squares approximation*, Tech. rep., http://arxiv.org/abs/0710.1435, 2007.
[6] H. H. Goldstine and J. von Neumann, *Numerical inverting of matrices of high order*, II, Proc. Amer. Math. Soc., 2 (1951), pp. 188–202.
[7] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.
[8] C. Gotsman and S. Toledo, *On the computation of null spaces of sparse rectangular matrices*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 445–463.
[9] D. E. Knuth, *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*, 3rd ed., Addison–Wesley, Reading, MA, 1997.
[10] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes*, 3rd ed., Cambridge University Press, Cambridge, UK, 2007.
[11] V. Rokhlin and M. Tygert, *A fast randomized algorithm for overdetermined linear least-squares regression*, Proc. Natl. Acad. Sci. USA, 105 (2008), pp. 13212–13217.
[12] S. J. Wright, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, 1997.